

**ADAPTIVE FIREFLY ALGORITHM FOR HIERARCHICAL  
TEXT CLUSTERING**



**ATHRAA JASIM MOHAMMED**

**UUM**  
**Universiti Utara Malaysia**

**DOCTOR OF PHILOSOPHY  
UNIVERSITI UTARA MALAYSIA  
2016**

## **Permission to Use**

In presenting this thesis in fulfilment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this thesis in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor(s) or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my thesis.

Requests for permission to copy or to make other use of materials in this thesis, in whole or in part, should be addressed to:

Dean of Awang Had Salleh Graduate School of Arts and Sciences

UUM College of Arts and Sciences

Universiti Utara Malaysia

06010 UUM Sintok

## Abstrak

Penggugusan teks digunakan oleh enjin carian untuk meningkatkan recall dan precision dalam bidang capaian maklumat. Memandangkan enjin carian beroperasi menggunakan kandungan Internet yang selalu berubah, maka satu algoritma penggugusan yang menawarkan pengumpulan item secara automatik tanpa maklumat awal berkenaan koleksi berkenaan adalah diperlukan. Kaedah penggugusan sedia ada menghadapi masalah untuk menentukan bilangan gugusan yang optimal dan gugusan yang padat. Dalam penyelidikan ini, satu algoritma penggugusan teks hierarki yang adaptif telah dicadangkan berdasarkan algoritma Firefly. Algoritma Firefly Adaptive (AFA) yang dicadangkan mempunyai tiga komponen: penggugusan dokumen, pembaikan gugusan dan penggabungan gugusan. Komponen pertama memperkenalkan algoritma Weight-based Firefly (WFA) yang berupaya untuk mengenal pasti pusat awalan dan gugusannya secara automatik bagi sesuatu koleksi teks. Bagi memperbaiki gugusan yang telah diperolehi, algoritma kedua iaitu Weight-based Firefly dengan Relocate (WFA<sub>R</sub>) telah dicadangkan. Kaedah ini membolehkan penempatan semula dokumen yang telah ditempatkan ke dalam gugusan yang baharu terhasil. Komponen ketiga, Weight-based Firefly Algorithm dengan Relocate dan Merging (WFA<sub>RM</sub>), bertujuan mengurangkan bilangan gugusan yang terhasil dengan menggabungkan gugusan bukan asli ke dalam gugusan asli. Eksperimen telah dilaksanakan untuk membandingkan algoritma yang dicadangkan dengan tujuh kaedah sedia ada. Peratusan kejayaan memperolehi bilangan gugusan yang optimal oleh AFA ialah 100% dengan mendapat purity dan f-measure 83% lebih tinggi daripada kaedah penanda aras. Bagi ukuran entropy, AFA menghasilkan nilai terendah (0.78) apabila dibandingkan dengan kaedah sedia ada. Keputusan ini memberi indikasi bahawa Algoritma Firefly Adaptif boleh menghasilkan gugusan yang padat. Penyelidikan ini menyumbang kepada domain perlombongan teks memandangkan penggugusan teks hierarki membantu pengindeksan dokumen dan proses pencapaian maklumat.

**Kata kunci:** Perlombongan teks, Penggugusan teks hierarki, Swarm Intelligence, Firefly Algorithm

## Abstract

Text clustering is essentially used by search engines to increase the recall and precision in information retrieval. As search engine operates on Internet content that is constantly being updated, there is a need for a clustering algorithm that offers automatic grouping of items without prior knowledge on the collection. Existing clustering methods have problems in determining optimal number of clusters and producing compact clusters. In this research, an adaptive hierarchical text clustering algorithm is proposed based on Firefly Algorithm. The proposed Adaptive Firefly Algorithm (AFA) consists of three components: document clustering, cluster refining, and cluster merging. The first component introduces Weight-based Firefly Algorithm (WFA) that automatically identifies initial centers and their clusters for any given text collection. In order to refine the obtained clusters, a second algorithm, termed as Weight-based Firefly Algorithm with Relocate (WFA<sub>R</sub>), is proposed. Such an approach allows the relocation of a pre-assigned document into a newly created cluster. The third component, Weight-based Firefly Algorithm with Relocate and Merging (WFA<sub>RM</sub>), aims to reduce the number of produced clusters by merging non-pure clusters into the pure ones. Experiments were conducted to compare the proposed algorithms against seven existing methods. The percentage of success in obtaining optimal number of clusters by AFA is 100% with purity and f-measure of 83% higher than the benchmarked methods. As for entropy measure, the AFA produced the lowest value (0.78) when compared to existing methods. The result indicates that Adaptive Firefly Algorithm can produce compact clusters. This research contributes to the text mining domain as hierarchical text clustering facilitates the indexing of documents and information retrieval processes.

**Keywords:** Text mining, Hierarchical text clustering, Swarm Intelligence, Firefly Algorithm

## **Acknowledgement**

Firstly, I would like to express my gratitude to Allah (S.W.T.) who helps me to complete my thesis.

Highly appreciate and gratefully acknowledges to my supervisors, Dr. Yuhanis Yusof and Dr. Husniza Husni who they support me, continues encourage me and guides me during my study.

I would like to thank my family for being here with me and supporting me during my study.



## Table of Contents

Permission to Use .....	i
Abstrak .....	ii
Abstract .....	iii
Acknowledgement .....	iv
Table of Contents .....	v
List of Tables .....	ix
List of Figures .....	xiii
List of Appendices .....	xvii
List of Abbreviations .....	xviii
<b>CHAPTER ONE INTRODUCTION .....</b>	<b>1</b>
1.1 Research Background.....	4
1.1.1 Clustering .....	5
1.1.2 Text Clustering.....	7
1.2 Problem Statement .....	9
1.3 Research Questions .....	10
1.4 Research Objectives .....	10
1.5 Research Significance .....	11
1.6 Scope and Limitations of the Research.....	12
1.7 Organization of the Research .....	12
<b>CHAPTER TWO LITERATURE REVIEW .....</b>	<b>15</b>
2.1 Introduction .....	15
2.2 Clustering Methods .....	16
2.2.1 Static Approach.....	16
2.2.1.1 Traditional Methods .....	16
2.2.1.1.1 Partitional Text Clustering .....	17
2.2.1.1.2 Density-based Text Clustering .....	22
2.2.1.1.3 Grid-based Text Clustering .....	25
2.2.1.1.4 Model-based Text Clustering .....	27
2.2.1.1.5 Hierarchical Text Clustering .....	29
Agglomerative Clustering .....	29

Divisive Clustering .....	35
2.2.1.2 Optimization Methods .....	42
2.2.1.2.1 Particle Swarm Optimization .....	46
2.2.1.2.2 Ant Colony Optimization .....	50
2.2.1.2.3 Firefly Algorithm.....	53
2.2.1.2.4 Hybrid of Clustering Techniques and other Search Optimization .....	62
2.2.2 Dynamic Approach .....	66
2.2.2.1 Estimation Approach .....	66
2.2.2.2 Population-based Approach .....	68
2.3 Research Gap .....	74
2.4 Summary .....	76
<b>CHAPTER THREE RESEARCH METHODOLOGY .....</b>	<b>78</b>
3.1 Research Design.....	79
3.1.1 Data Acquisition Phase .....	80
3.1.1.1 Data Collection .....	81
3.1.1.2 Data Pre-processing .....	82
Step 1: Data Cleaning .....	83
Step 2: Data Representation .....	86
3.1.2 Clustering Phase.....	88
3.1.3 Cluster Refining Phase.....	93
3.1.4 Cluster Merging Phase .....	95
3.2 Implementation of Algorithms .....	98
3.3 Evaluation .....	99
3.3.1 Performance Metrics .....	100
3.3.1.1 Internal and Relative Quality Metrics .....	100
3.3.1.2 External Quality Metrics .....	101
3.3.2 Statistical Analysis .....	103
3.4 Summary .....	104
<b>CHAPTER FOUR DOCUMENT CLUSTERING.....</b>	<b>106</b>
4.1 Weight-based Firefly Algorithm (WFA) .....	106

4.1.1 Initialization of Parameters .....	106
4.1.2 Data Clustering .....	108
4.2 Evaluation .....	119
4.3 Summary .....	127
<b>CHAPTER FIVE CLUSTER REFINING.....</b>	<b>128</b>
5.1 Introduction .....	128
5.2 Document Re-locating .....	128
5.3 Evaluation .....	132
5.3.1 Comparison between $WFA_R$ and $WFA$ .....	132
5.3.2 Comparison between $WFA_R$ and Other Methods .....	140
5.4 Summary .....	147
<b>CHAPTER SIX CLUSTER MERGING.....</b>	<b>149</b>
6.1 Introduction .....	149
6.2 Cluster Merging Algorithm.....	150
6.2.1 Merge Clusters .....	150
6.2.2 Refine Merged Clusters .....	152
6.3 Evaluation .....	159
6.3.1 Comparison between $WFA_{RM}$ and $WFA_R$ .....	160
6.3.1.1 Number of Clusters between $WFA_{RM}$ and $WFA_R$ .....	160
6.3.1.2 Performance Metrics between $WFA_{RM}$ and $WFA_R$ .....	161
6.3.1.3 Paired Samples T-test between $WFA_{RM}$ and $WFA_R$ .....	168
6.3.2 Comparison between $WFA_{RM}$ and Static Methods.....	169
6.3.2.1 Number of Clusters between $WFA_{RM}$ and Static Methods.....	169
6.3.2.2 Performance Metrics between $WFA_{RM}$ and Static Methods.....	170
6.3.2.3 Independent Samples T-test between $WFA_{RM}$ and Static Methods.....	177
6.3.3 Comparison between $WFA_{RM}$ and Dynamic Methods .....	179
6.3.3.1 Number of Clusters between $WFA_{RM}$ and Dynamic Methods .....	179
6.3.3.2 Performance Metrics between $WFA_{RM}$ and Dynamic Methods .....	180
6.3.3.3 Independent Samples T-test between $WFA_{RM}$ and Dynamic Methods .....	185
6.4 Summary .....	186



<b>CHAPTER SEVEN EVALUATION OF ADAPTIVE FA ON VARIOUS DATASETS.....</b>	<b>188</b>
7.1 Introduction .....	188
7.2 Comparison WFA <sub>RM</sub> with Static Methods .....	188
7.2.1 Evaluation Number of Clusters between WFA <sub>RM</sub> and Static Methods.....	189
7.2.2 Evaluation of Performance Metrics between WFA <sub>RM</sub> and Static Methods .....	190
7.2.3 Evaluation Independent Samples T-test between WFA <sub>RM</sub> and Static Methods.....	204
7.3 Comparison WFA <sub>RM</sub> with Dynamic Methods .....	213
7.3.1 Evaluation Number of Clusters between WFA <sub>RM</sub> and Dynamic Methods.....	214
7.3.2 Evaluation Performance Metrics between WFA <sub>RM</sub> and Dynamic Methods .....	215
7.3.3 Evaluation Independent Samples T-test between WFA <sub>RM</sub> and Dynamic Methods.....	227
7.4 Summary .....	234
<b>CHAPTER EIGHT CONCLUSION AND FUTURE WORK.....</b>	<b>236</b>
8.1 Research Contribution.....	236
8.2 Future Work .....	237

## List of Tables

Table 2.1 Summary of existing researches in partitional text clustering. ....	21
Table 2.2 Summary of existing researches in partitional numerical clustering. ....	22
Table 2.3 Summary of existing researches in hierarchical text clustering. ....	40
Table 2.4 Summary of existing researches in hierarchical numerical clustering. ....	42
Table 2.5 .Summary of existing researches in Particle Swarm Optimization in text clustering. ....	49
Table 2.6 Summary of existing researches in Particle Swarm Optimization in numerical clustering. ....	50
Table 2.7 Summary of existing researches in Ant Colony Optimization in text clustering. ...	52
Table 2.8 Summary of existing researches in Ant Colony Optimization in numerical clustering. ....	53
Table 2.9 Summary of existing researches in Firefly Algorithm in web intelligent data. ....	61
Table 2.10 Summary of existing researches in Firefly Algorithm in numerical clustering. ..	61
Table 2.11 Summary of existing researches in the hybridization of clustering techniques and other search optimization in text clustering. ....	65
Table 2.12 Summary of existing researches in the hybridization of clustering techniques and other search optimization in numerical clustering. ....	65
Table 3.1 Description of Datasets. ....	81
Table 4.1 Parameters setting in WFA. ....	111
Table 4.2 External quality metrics of clustering: WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means. ....	120
Table 4.3 Internal and relative quality metrics of clustering: WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means. ....	121
Table 4.4 Average number of clusters of WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means. ....	126
Table 4.5 Results of quality performance of WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means. ....	127
Table 5.1 External quality metrics: WFA vs. WFA <sub>R</sub> . ....	133
Table 5.2 Internal and relative quality metrics: WFA vs. WFA <sub>R</sub> . ....	134
Table 5.3 Average number of clusters: WFA vs. WFA <sub>R</sub> . ....	139
Table 5.4 Summary of quality performance: WFA vs. WFA <sub>R</sub> . ....	139

Table 5.5 ... External quality metrics: WFA <sub>R</sub> vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means. ....	141
Table 5.6 Internal and Relative quality metrics: WFA <sub>R</sub> vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means.....	142
Table 5.7 Average number of clusters: WFA <sub>R</sub> vs. PSO vs. K-means vs. FA K-means vs. Bisect K-means. ....	146
Table 5.8 Summary of quality performance: WFA <sub>R</sub> vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means. ....	147
Table 6.1 Average number of clusters of WFA <sub>R</sub> & WFA <sub>RM</sub> . ....	160
Table 6.2 External quality metrics of clustering and standard deviation: WFA <sub>R</sub> vs. WFA <sub>RM</sub> . ....	162
Table 6.3 Internal and relative quality metrics of clustering and standard deviation: WFA <sub>R</sub> vs. WFA <sub>RM</sub> . ....	163
Table 6.4 Quality performance of WFA <sub>R</sub> & WFA <sub>RM</sub> algorithms. ....	168
Table 6.5 The P-value between WFA <sub>R</sub> & WFA <sub>RM</sub> algorithms. ....	169
Table 6.6 Average number of clusters: WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means.....	170
Table 6.7 External quality metrics of clustering: WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means.....	172
Table 6.8 Internal and relative quality metrics of clustering and standard deviation: WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means.....	173
Table 6.9 Summary of external quality performance results: WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means. ....	176
Table 6.10 Summary of internal and relative quality performance results: WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means. ....	177
Table 6.11 The P-value between WFA <sub>RM</sub> & static methods.....	177
Table 6.12 Average number of clusters: WFA <sub>RM</sub> vs. PGSCM vs. DCPG. ....	179
Table 6.13 External quality metrics of clustering and standard deviation: WFA <sub>RM</sub> vs. PGSCM vs. DCPG.....	180
Table 6.14 Internal and relative quality metrics of clustering and standard deviation: WFA <sub>RM</sub> vs. PGSCM vs. DCPG. ....	182
Table 6.15 Summary of quality performance results: WFA <sub>RM</sub> vs. PGSCM vs. DCPG.....	184
Table 6.16 The P-value between WFA <sub>RM</sub> & dynamic methods.....	185
Table 7.1 Average numbers of clusters: WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means using different datasets. ....	189

Table 7.2 External quality Purity (average, best, worst, standard deviation): WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets (balanced and un-balanced datasets).....	191
Table 7.3 External quality F-measure (average, best, worst, standard deviation): WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets.....	193
Table 7.4 External quality Entropy (average, best, worst, standard deviation): WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means using different datasets.....	195
Table 7.5 Internal quality ADDC (average, best, worst, standard deviation): WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means using different datasets. ....	197
Table 7.6 Relative quality DBI (Average, Best, Worst, standard deviation): WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets. ....	199
Table 7.7 Relative quality DI (average, best DI, worst DI, standard deviation): WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets.....	201
Table 7.8 Summary of quality performance results: WFA <sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means.....	203
Table 7.9 The P-value between WFA <sub>RM</sub> & static methods using average purity results (sig 2 tailed) with different datasets.....	205
Table 7.10 The P-value between WFA <sub>RM</sub> & static methods using average F-measure results (sig 2 tailed) with different datasets.....	207
Table 7.11 The P-value between WFA <sub>RM</sub> & static methods using average Entropy results (sig 2 tailed) with different datasets.....	208
Table 7.12 The P-value between WFA <sub>RM</sub> & static methods using average ADDC results (sig 2 tailed) with different datasets.....	210
Table 7.13 The P-value between WFA <sub>RM</sub> & static methods using average DBI results (sig 2 tailed) with different datasets.....	211
Table 7.14 The P-value between WFA <sub>RM</sub> & static methods using average DI results (sig 2 tailed) with different datasets.....	212
Table 7.15 Average number of clusters: WFA <sub>RM</sub> vs. PGSCM vs. DCPG using different datasets.....	214

Table 7.16 External quality Purity (average, best, worst, standard deviation): WFA <sub>RM</sub> vs. PGSCM vs. DCPG using different datasets. ....	216
Table 7.17 External quality F-measure (average, best, worst, standard deviation): WFA <sub>RM</sub> vs. PGSCM vs. DCPG using different datasets. ....	218
Table 7.18 External quality Entropy (average, best, worst, standard deviation): WFA <sub>RM</sub> vs. PGSCM vs. DCPG using different datasets. ....	219
Table 7.19 Internal quality ADDC (average, best, worst, standard deviation): WFA <sub>RM</sub> vs. PGSCM vs. DCPG using different datasets. ....	221
Table 7.20 Relative quality DBI (average, best, worst, standard deviation): WFA <sub>RM</sub> vs. PGSCM vs. DCPG using different datasets. ....	222
Table 7.21 Relative quality DI (average, best DI, worst DI, standard deviation): WFA <sub>RM</sub> vs. PGSCM vs. DCPG using different datasets. ....	224
Table 7.22 Summary of quality performance results: WFA <sub>RM</sub> vs. PGSCM vs. DCPG. ....	225
Table 7.23 The P-value between WFA <sub>RM</sub> & dynamic methods using average purity results (sig 2 tailed) with different datasets. ....	228
Table 7.24 The P-value between WFA <sub>RM</sub> & dynamic methods using average F-measure results (sig 2 tailed) with different datasets. ....	229
Table 7.25 The P-value between WFA <sub>RM</sub> & dynamic methods using average Entropy results (sig 2 tailed) with different datasets. ....	230
Table 7.26 The P-value between WFA <sub>RM</sub> & dynamic methods using average ADDC results (sig 2 tailed) with different datasets. ....	231
Table 7.27 The P-value between WFA <sub>RM</sub> & dynamic methods using average DBI results (sig 2 tailed) with different datasets. ....	232
Table 7.28 The P-value between WFA <sub>RM</sub> & dynamic methods using average DI results (sig 2 tailed) with different datasets. ....	233

## List of Figures

Figure 1.1. Text analytics techniques and external disciplines .....	3
Figure 2.1. Proposed taxonomy of clustering methods.....	15
Figure 2.2. Steps of K-means algorithm .....	17
Figure 2.3. The Single Linkage Hierarchical Clustering (SLHC).....	30
Figure 2.4. The Complete Linkage Clustering Hierarchical (CLHC).....	31
Figure 2.5. The Un-weighted Pair Group Method with Arithmetic Mean (UPGMA) Resource. Manning, Raghavan, and Schütze (2008) .....	32
Figure 2.6. The process of Bisect K-means .....	36
Figure 2.7. The taxonomy of optimization algorithms .....	43
Figure 2.8. The step-by-step process of PSO clustering .....	47
Figure 2.9. Pseudo code of Firefly Algorithm .....	55
Figure 2.10. Pseudo code of integrated Firefly with K-means clustering algorithm .....	58
Resource. Tang, Fong, Yang, and Deb (2012).....	58
Figure 2.11. Pseudo code of integrated Bat with K-means clustering algorithm.....	59
Figure 2.12. Pseudo code of integrating Particle Swarm Optimization with Genetic Algorithm (DCPG).....	70
Figure 2.13. Pseudo code of practical General Stochastic Clustering Method (PGSCM).....	73
Figure 3.1. The experimental research steps .....	78
Figure 3.2. The components of the proposed Adaptive Firefly algorithm for hierarchical text clustering.....	79
Figure 3.3. The phases of proposed hierarchical text clustering.....	80
Figure 3.4. An example of document from the Reuters dataset .....	83
Figure 3.5. An example of a cleaned document.....	83
Figure 3.6. An example of extracted terms .....	84
Figure 3.7. An example of words with the length more than two.....	84
Figure 3.8: An example of the removed stop words. ....	85
Figure 3.9. An example of word frequency .....	85
Figure 3.10. The term frequency matrix .....	86
Figure 3.11. TFIDF matrix.....	88
Figure 3.12. Flow of Hierarchical Text clustering using Weight-based Firefly Algorithm (WFA).....	89
Figure 3.13. An example of the total weight matrix .....	90

Figure 3.14. The process of Weight-based Firefly Algorithm (WFA).....	92
Figure 3.15. Process of document re-locating.....	93
Figure 3.16. Comparison between clusters for document re-locating.....	94
Figure 3.17. Process of merging similar clusters in enhanced Un-weighted Pair Group Method with Arithmetic Mean (eUPGMA).....	96
Figure 4.1. One dimension search space.....	107
Figure 4.2. An example of normalized positioning.....	107
Figure 4.3. An example of competition in standard Firefly Algorithm (FA).....	109
Figure 4.4. An example of competition in Weight-based Firefly Algorithm (WFA) .....	109
Figure 4.5. Weight-based Firefly Algorithm (WFA) for hierarchical text clustering.....	113
Figure 4.6. An example of TFIDF for 20Newsgroups.....	114
Figure 4.7. An example of cosine similarity table for 20Newsgroups dataset.....	115
Figure 4.8. An example of Euclidean distance table for 20Newsgroups dataset .....	115
Figure 4.9. An example of total weight for 20Newsgroups dataset.....	116
Figure 4.10. An example of normalized initial positioning for 20Newsgroups dataset.....	117
Figure 4.11. Graphical representation of initial document positioning for 20Newsgroups dataset .....	117
Figure 4.12. An example of graphical representation of final document positioning for 20Newsgroups dataset .....	118
Figure 4.13. Graphical representation of quality metrics of WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means; a) Purity, b) F-measure, c) Entropy, d) ADDC, e) DBI, and f) DI.....	122
Figure 5.1. The pseudo code of Document Re-locating.....	129
Figure 5.2. The process of $WFA_R$ .....	129
Figure 5.3. Steps of the $WFA_R$ algorithm .....	130
Figure 5.4. Graphical representation of quality metrics between WFA & $WFA_R$ ; a) Purity, b) F-measure, c) Entropy, d) ADDC, e) DBI, and f) DI.....	135
Figure 5.5. Graphical representation of quality metrics of $WFA_R$ vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means; a) Purity, b) F-measure, c) Entropy, d) ADDC, e) DBI, and f) DI.....	143
Figure 6.1. Process in $WFA_{RM}$ .....	149
Figure 6.2. Process of cluster merging Algorithm (eUPGMA) .....	150
Figure 6.3. Pseudo code for selecting pure clusters .....	153
Figure 6.4. Pseudo code of identifying centers for pure clusters .....	153
Figure 6.5. Pseudo code of relocating non-pure clusters .....	154

Figure 6.6. Cosine similarity matrix between cluster1 and cluster2 .....	155
Figure 6.7. Results of merging clusters for 20Newsgroups dataset .....	157
Figure 6.8. An example of TFIDF of documents in Cluster1 and center calculation .....	158
Figure 6.9. An example of TFIDF of document 28 in Cluster 3 .....	158
Figure 6.10. An example of the centers of selected pure clusters .....	158
Figure 6.11. Calculation of minimum distance between centers of pure clusters and members of non-pure cluster .....	159
Figure 6.12. Number of produced clusters by $WFA_R$ and $WFA_{RM}$ .....	161
Figure 6.14. Graphical representation of quality metrics: $WFA_{RM}$ vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means (a) Purity, (b) F-measure, (c) Entropy, (d) ADDC, (e) DBI, and (f) DI. ....	174
Figure 6.15. External quality metrics: $WFA_{RM}$ vs. PGSCM vs. DCPG .....	181
Figure 6.16. Internal and relative quality metrics: $WFA_{RM}$ vs. PGSCM vs. DCPG .....	183
Figure 7.1. Results of the number of generated clusters by $WFA_{RM}$ and the real number of clusters of all static methods .....	190
Figure 7.2. Average Purity results: $WFA_{RM}$ vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets .....	192
Figure 7.3. Average F-measure result: $WFA_{RM}$ vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets .....	194
Figure 7.4. Average Entropy result: $WFA_{RM}$ vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets .....	196
Figure 7.5. Average ADDC result: $WFA_{RM}$ vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets .....	198
Figure 7.6. Average DBI result: $WFA_{RM}$ vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets .....	200
Figure 7.7. Average DI result: $WFA_{RM}$ vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets .....	202
Figure 7.8. Number of generated clusters: $WFA_{RM}$ vs. the real number of clusters vs. PGSCM vs. DCPG .....	215
Figure 7.9. Average Purity result: $WFA_{RM}$ vs. PGSCM vs. DCPG using different datasets .....	217
Figure 7.10. Average F-measure result: $WFA_{RM}$ vs. PGSCM vs. DCPG using different datasets .....	218
Figure 7.11. Average Entropy result: $WFA_{RM}$ vs. PGSCM vs. DCPG using different datasets .....	220



Figure 7.12. Average Entropy result: $WFA_{RM}$ vs. PGSCM vs. DCPG using different datasets .....	221
Figure 7.13. Average DBI result: $WFA_{RM}$ vs. PGSCM vs. DCPG using different datasets	223
Figure 7.14. Average DI result: $WFA_{RM}$ vs. PGSCM vs. DCPG using different datasets. .	225



## List of Appendices

Appendix A Samples of Documents Datasets .....	253
Appendix B Stop Words List .....	259



## List of Abbreviations

<b>ACK</b>	Ant Colony with Kernal method
<b>ACO</b>	Ant Colony Optimization
<b>ACPSO</b>	Automatic Clustering Particle Swarm Optimization
<b>ALHC</b>	Average Linkage Hierarchical Clustering
<b>AP</b>	Affinity Propagation
<b>BIC</b>	Bayesian Information Criterion
<b>BKM</b>	Bisect K-means
<b>C-bat</b>	Bat algorithm with K-means
<b>C-cuckoo</b>	Cuckoo algorithm with K-means
<b>C-firefly</b>	Firefly algorithm with K-means
<b>CFWS</b>	Clustering based on Frequent Word Sequence
<b>CLHC</b>	Complete Linkage Hierarchical Clustering
<b>CLIQUE</b>	Clustering In QUEst
<b>CMS</b>	Clustering based on Maximal Frequent Sequence
<b>CPSO</b>	Particle Swarm Optimization with K-means
<b>CRC</b>	Corrected Rand Coefficient
<b>C-wolf</b>	Wolf algorithm with K-means
<b>DBI</b>	Davies Bouldin Index
<b>DBSCAN</b>	Density-Based Spatial Clustering of Application with Noise
<b>DCGA</b>	Dynamic Clustering Genetic Algorithm
<b>DCPG</b>	Dynamic Clustering Particle Swarm Optimization with Genetic Algorithm
<b>DCPSO</b>	Dynamic Clustering using Particle Swarm Optimization
<b>DF</b>	Document Frequency
<b>DHC</b>	Dynamic Hierarchical Compact
<b>DHS</b>	Dynamic Hierarchical Star
<b>DI</b>	Dunn Index
<b>ES</b>	Evolution Strategy
<b>FA</b>	Firefly Algorithm
<b>FIHC</b>	Frequent Itemset based Hierarchical Clustering
<b>FTC</b>	Frequent Term based Clustering
<b>GA</b>	Genetic Algorithm

<b>GGCA</b>	General Grid Clustering Approach
<b>GSA</b>	Gravitational Search Algorithm
<b>GSA-KM</b>	Gravitational Search Algorithm with K-means
<b>HBMO</b>	Honey Bee Mating Optimization
<b>HCM</b>	Hierarchical Clustering Method
<b>HS</b>	Harmony Search
<b>IDF</b>	Inverse Document Frequency
<b>KCPSO</b>	K-means with Particle Swarm Optimization
<b>KFA</b>	K-means with Firefly Algorithm
<b>KHM</b>	K-Harmonic Means algorithm
<b>KPSO</b>	K-means with Particle Swarm Optimization
<b>NMI</b>	Normalized mutual information
<b>NN</b>	Neural Networks
<b>OptiGrid</b>	Optimal Grid clusteing
<b>PDDP</b>	Principal Direction Divisive Partitioning
<b>PGSCM</b>	Practical General Stochastic Clustering Method
<b>PSO</b>	Particle Swarm Optimization
<b>PSOKHM</b>	Particle Swarm Optimization with K-Harmonic Means
<b>RFA</b>	Reachback Firefly Algorithm
<b>SA</b>	Simulated Annealing
<b>SAP</b>	Seed Affinity Propagation
<b>SLHC</b>	Single Linkage Hierarchical Clustering
<b>SOM</b>	Self Organizing Map
<b>STING</b>	Statistical Information Grid-based method
<b>TC</b>	Term Contribution
<b>TFIDF</b>	Term Frequency–Inverse Document Frequency
<b>TSP</b>	Travelling Salesman Problem
<b>UPGMA</b>	Un-weighted Pair Group Method with Arithmetic Mean
<b>VI</b>	Validity Index
<b>VSM</b>	Vector Space Model
<b>WFA</b>	Weight-based Firefly Algorithm
<b>WFA<sub>R</sub></b>	Weight-based Firefly Algorithm with relocating
<b>WFA<sub>RM</sub></b>	Weight-based Firefly Algorithm with relocating with merging algorithm

# **CHAPTER ONE**

## **INTRODUCTION**

Adaptation in computer science is the process of a system. Adaptive system adapts its behavior to users depending on the information that can be collected from users and the environment. An adaptive system is a set of entities that interact between them and change their behavior in response to their environment. The aim of adaptive change is to achieve the goal. Artificial systems, such as robots, can adapt with the environment by sensing the new condition through the use of feedback loops (i.e. the output of the system becomes input). Furthermore, it can adapt a parameter from the environment based on the change of the conditions; for example, a new adaptive parameter (speed) changes based on the color of the agent added in the adaptive flocking algorithm (Folino, Forestiero, & Spezzano, 2009), and the value of pheromone at each location introduced in the picking and dropping probability functions of the adaptive ant colony clustering algorithm, and it also improves the similarity scaling factor by automatic adoption (El-Feghi, Errateeb, Ahmadi, & Sid-Ahmed, 2009). The adaptive system utilizes machine learning to adapt its behavior over time (Glass, 2011). Swarm Intelligence provides a useful paradigm for implementing adaptive systems (Kennedy & Eberhart, 2001).

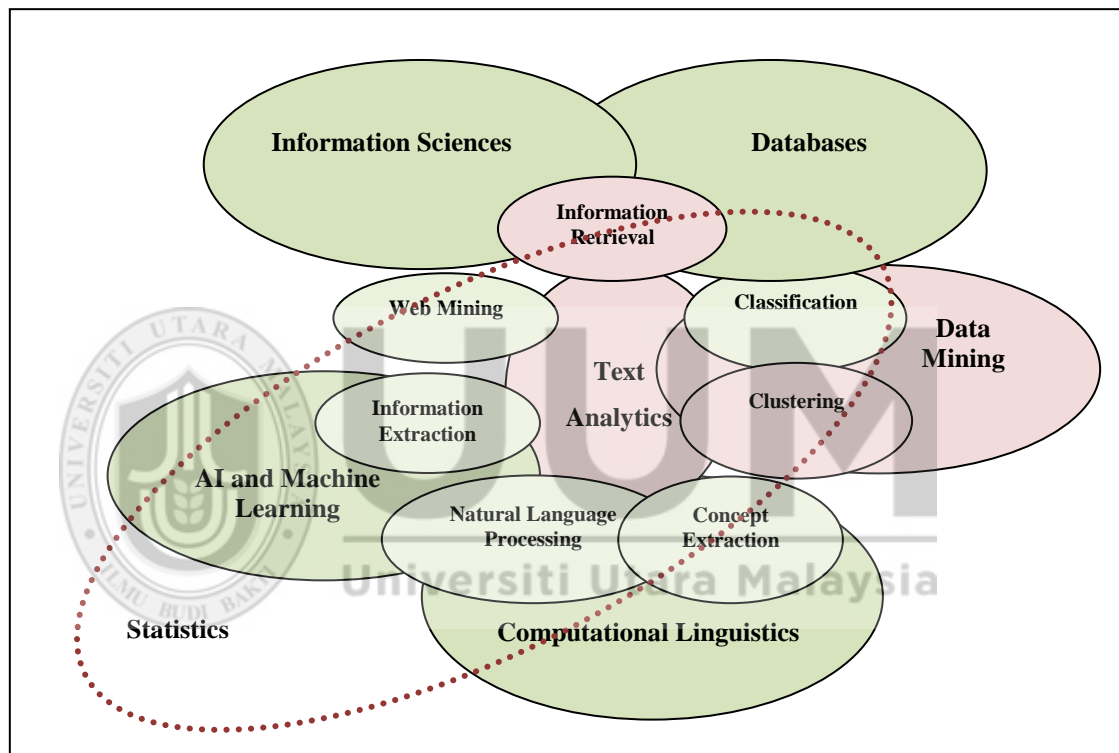
Swarm Intelligence or Swarm Computing is “the emergent collective intelligence of groups of simple agents” (Bonabeau, Dorigo, & Theraulaz, 1999). It is useful to solve some problems that cannot be processed using traditional methods. It is used to find optimal solutions in hard problems, such as Travelling Salesman Problem (TSP)

(Dorigo & Gambardella, 1997), sound processing (Muñoz, Llanos, Coelho, & Ayala-Rincon, 2011), and text clustering (Feng et al., 2010).

An example of Swarm Intelligence algorithms is the Firefly Algorithm (FA). FA is developed by Xin-She Yang in 2007 at Cambridge University. Firefly algorithm has two important issues, the light intensity and the attractiveness. For maximum optimization problems, the light intensity  $I$  of a firefly at a particular location  $x$ , termed as  $I(x)$ , can be determined by objective function  $f(x)$ . On the other hand, the attractiveness  $\beta$  is relative where the change depends on the distance between two fireflies (Yang & He, 2013; Yang, 2010a, 2010b). Firefly algorithm has been used in many applications, such as economic emission load dispatch problem (Apostolopoulos & Vlachos, 2011; Yang, Hosseini, & Gandomi, 2012), speech recognition (Hassanzadeh, Faez, & Seyfi, 2012), image segmentation (Hassanzadeh, Vojodi, & Moghadam, 2011; Horng & Jiang, 2010), reliability-redundancy allocation problem (dos Santos Coelho, de Andrade Bernert, & Mariani, 2011), semantic web service composition (Pop et al., 2011), data classification (Nandy, Sarkar, & Das, 2012), anomaly detection (Adaniya, Abr̃ao, & Proenc,a Jr., 2013), and parallel and distributed systems (Falcon, Almeida, & Nayak, 2011).

The main idea of text analysis is to extract valuable information from various resources on the Internet, such as web text, social media and blogs. The obtained information is later converted into numerical values which can be combined with other structured data before being analyzed using one of the data mining techniques. Hence, text analytics involves seven different areas, which are document classification, document clustering, information extraction, natural language

processing, concept extraction, information retrieval, and web mining. These seven areas are related with six fields, such as statistics, computational linguistics, data mining, databases, artificial intelligence and machine learning, and information sciences. Figure 1.1 shows the text analytics techniques and external disciplines involved (Miner et al., 2012, p.31).



*Figure 1.1.* Text analytics techniques and external disciplines  
Resource. Miner et al., (2012, p.31).

For information retrieval performed via search engines, users suffer from information overload. They may be presented with irrelevant information that have been retrieved by the search engine. In order to automate the organization of documents, text clustering may be useful in enhancing the search and retrieval process. In practice, when users submit a text query to a search engine, the searching

process will require some time to map the submitted words (i.e. query) against indexing databases. The larger the amount of documents uploaded in the web, the larger the indexing database would be. Hence, efficient clustering techniques are beneficial in creating a better structure of the indexing database.

Existing text clustering techniques suffer from drawbacks such as the determination of the number of clusters as initial value and random of initial centers, that later produce poor clustering results. Hence, many researchers have integrated existing text clustering techniques with Swarm Intelligence (SI) algorithms; for example, divisive clustering framework was integrated with Particle Swarm Optimization (PSO) and the result validates the effectiveness of this integration (Feng et al., 2010). On the other hand, there is an FA that is successful in many domains including numerical data clustering (Banati & Bajaj, 2013; Senthilnath, Omkar, & Mani, 2011). In addition, it is also noted that FA has a higher capacity to find an optimal solution compared to PSO (Yang, 2010a). Nevertheless, it has yet to be reported on work that utilizes FA in text clustering. Hence, this study extends the existing work of FA by adapting it into hierarchical text clustering.

## **1.1 Research Background**

This study focuses on text clustering based on the behavior of Firefly Algorithm. In particular, this study looks into the adaptive characteristic of the Firefly Algorithm (FA) in grouping text documents.



### 1.1.1 Clustering

Cluster analysis refers to the process of grouping un-labelled patterns or objects into multi-groups depending on their similarity. Each group is called a cluster, which contains objects that are similar between them and dissimilar from objects in different cluster (Das, Abraham, & Konar, 2009; Yin, Kaku, Tang, & Zhu, 2011). Clustering is unsupervised learning that does not need training data and does not assign target class for each instance in the dataset (Jensi & Jiji, 2013). Various clustering algorithms have been proposed by many researchers. In general, these algorithms are classified into five categories (Zhang, Cao, & Lee, 2013; Zhang & Cao, 2011): partitional clustering, hierarchical clustering, density-based clustering, grid-based clustering, and model-based clustering.

Partitional clustering algorithms attempt to split a dataset into a set of dissimilar clusters. The splitting process depends on the objective function that confirms the data local structure. The objective function attempts to minimize the summation of square error between the center of a cluster (centroid point) and all points in a cluster (Anitha Elavarasi, Akilandeswari, & Sathiyabhama, 2011). Clusters that are produced must include at least one object and the object must not belong to another cluster; this is called hard clustering. Another type is called soft clustering, where an object belongs to multiple clusters with membership degree (Bordogna & Pasi, 2012; Youssef, 2011).

The hierarchical clustering algorithm constructs a hierarchy of clusters. There are two approaches of this method (Das, Abraham, & Konar, 2009). The first approach is agglomerative hierarchical clustering which operates from the bottom to the top,

where every object is located in a single cluster, and merges them based on similarity between clusters (Gil-Garicia & Pons-Porrata, 2010; Wilson, Boots, & Millward, 2002). The second approach is divisive hierarchical clustering which starts from the top to the bottom, where all objects are initially being assigned in one cluster, and splits them using one of the partitional clustering approaches (Bordogna & Pasi, 2012). The advantage of hierarchical clustering is that it does not require the number of  $k$  clusters which is the drawback of the partitional clustering. Hierarchical clustering is a very suitable method for text clustering (Feng et al., 2010; Zhu, Fung, Mu, & Li, 2008). Hierarchical clustering methods build a hierarchy of nested quality clusters (Murugesan & Zhang, 2011a, 2011b; Wilson, Boots, & Millward, 2002).

Density-based clustering is a method to build clusters based on dense regions of objects in high-dimensional space that are isolated by low density areas. The idea of this algorithm is to detect the area of high density and the area of lower density (Das, Abraham, & Konar, 2009). The features of density-based algorithm are noise tolerate, the ability of handling arbitrary shaped clusters, and requires only a single scan on the input dataset. Additionally, it also requires the initialization of density parameters (Anitha Elavarasi et al., 2011).

The grid-based clustering algorithms split the space into finite numbers of rectangular cells at a high level. Then, in the next lower level, each cell is divided into a number of smaller cells (Zhao, Cao, Zhang, & Zhang, 2011). Every small cell contains parameters that are calculated like count, mean, min, and max. Higher level cells can easily access into parameters in the lower level. There are several methods

such as STING (Wang, Yang, & Muntz, 1997), CLIQUE (Agrawal, Gehrke, Gunopulos, & Raghavan, 1998), OptiGrid (Hinneburg & Keim, 1999).

On the other hand, the model-based clustering method tries to optimize the fit between some mathematical models and data. In addition, it characterizes the description of data groups. Hence, each group appears with a class or concept. The Neural Networks (NN) (Wang et al., 2011) and Self Organizing Map (SOM) (Kohonen, 1998) are two types of model-based clustering (Zhang, Cao, & Lee, 2013; Zhang & Cao, 2011).

### **1.1.2 Text Clustering**

Text clustering or document clustering organizes text documents as clusters; similar documents are in one group and dissimilar ones in another group (Xinwu, 2010). Document clustering is applicable in document organization and browsing which the hierarchical approach can be very beneficial for documents to browse systematically.

It also has been applied in document summarization. Document clustering summarizes a large quantity of documents using the key concepts that are extracted from the documents. In addition, it discovers a hidden pattern based on the similarities between the documents (Aggarwal & Zhai, 2012).

In text clustering, the documents are represented as a vector in a vector space model (VSM). Each document is treated as a bag of words which represents the document features (Guan, Shi, Marchese, Yang, & Liang, 2011). Most of the existing texts clustering algorithms use the similarities between the texts in the text clustering process. Similarity means points, features, or details that are alike in two documents.

There are many similarity functions, such as Cosine similarity, Jaccard similarity and Hamman similarity (Yin, Kaku, Tang, & Zhu, 2011), that can be used in determining the similarity between documents.

The performance of text clustering algorithms is measured by using statistical mathematical functions. These functions are based on the similarity or dissimilarity between the documents, such as Davies Bouldin Index (DBI) (Davies & Bouldin, 1979), Dunn Index (DI) (Dunn, 1974), Entropy (Shannon, 1948), F-measure (Meghabghab & Kandel, 2008), purity (Murugesan & Zhang, 2011a, 2011b), and Average Distance between Documents and Center (ADDC) (Murugesan & Zhang, 2011a, 2011b). The lower value of DBI, high value of DI, lower value of Entropy, high value of F-measure, high value of purity, and lower value of ADDC, means good quality cluster (Das, Abraham, & Konar, 2009). For example, Murugesan and Zhang (2011a) produced low quality clusters where the result of Entropy was 1.41, F-measure was 0.29 and purity was 0.49 for TR11 dataset.

There exist various reported works on text clustering. Some of them utilize classical text clustering algorithms (Gupta & Sharma, 2010; Xinwu, 2010; Zhang, Yoshida, Tang, & Wang, 2010) and the more recent works introduce meta-heuristics algorithms for text clustering (Feng et al., 2010; Wang, Shen, & Tang, 2009).

The meta-heuristics text clustering has been applied to achieve global optimal solutions or nearly optimal without the need for prior knowledge about the data set (Das, Abraham, & Konar, 2009). Meta-heuristics algorithms, such as Firefly Algorithm (FA) (Yang & He, 2013; Yang, 2010a, 2010b), Particle Swarm

Optimization (PSO) (J. Kennedy & Eberhart, 1995), and Ant Colony Optimization (ACO) (Dorigo, 1992), have been used in many clustering works.

## **1.2 Problem Statement**

The problem of text clustering has been studied widely, especially using clustering techniques such as partitional (Yao, Pi, & Cong, 2012) and hierarchical (Gupta & Sharma, 2010). Agglomerative clustering does not work well with high dimensional data (Zhu, Fung, Mu, & Li, 2008), on the other hand, divisive clustering is efficient and useful in document clustering and information retrieval (Feng et al., 2010; Kashef & Kamel, 2009). However, existing works on divisive hierarchical clustering produced clusters with less quality (Bordogna & Pasi, 2012; Gupta & Sharma, 2010).

Recently, there exist hybrid works of hierarchical approaches that combine the divisive and agglomerative clustering approaches and produced better clusters (Murugesan & Zhang, 2011a, 2011b; Zhu, Fung, Mu, & Li, 2008). Unfortunately, the divisive techniques that were utilized in existing hybrid hierarchical approaches, such as partitional clustering (Zhu, Fung, Mu, & Li, 2008) and Bisect K-means (Murugesan & Zhang, 2011a, 2011b), have drawbacks in determining an optimal number of  $k$  clusters (Hassanzadeh & Meybodi, 2012; Xu, 2005; Youssef, 2011; Zhong, Liu, & Li, 2010). Such a problem arises as there is no prior knowledge on the utilized datasets. The works in Bisect K-means fall into local optima because of the random initialization of the centroids (Chen et al., 2005; Hassanzadeh & Meybodi,

2012; Rana, Jasola, & Kumar, 2010; Tang, Fong, Yang, & Deb, 2012). Different initial centers will produce different clusters, and hence, generate different qualities.

In addition, in divisive hierarchical clustering such as Bisect K-means, documents which are assigned to a cluster cannot be re-assigned to another cluster (Forsati, Mahdavi, Shamsfard, & Meybodi, 2013; Murugesan & Zhang, 2011a, 2011b; Xu, 2005). This means that a document in a higher level of hierarchy cannot be relocated into another cluster (at a lower hierarchy) even though it is identified to be more similar to the center of the newly created cluster. On top of that, the divisive approach produces a large number of clusters. This later affects the performance quality (Murugesan & Zhang, 2011a, 2011b).

### **1.3 Research Questions**

- i. How to adapt the standard FA in clustering to identify the initial number of clusters and its centroids?
- ii. How to design an approach that allows the re-location of an item once it has been grouped in a particular cluster?
- iii. How to merge between two similar clusters?
- iv. How to evaluate the proposed algorithms?

### **1.4 Research Objectives**

The main objective of this thesis is to construct an adaptive hierarchical clustering algorithm based on the Firefly Algorithm (FA) for text documents. In order to achieve the main objective, the following sub-objectives must be addressed:

- i. To design a divisive algorithm based on the firefly algorithm to identify the initial centroids.
- ii. To design an algorithm for the re-location of a document upon the creation of a new cluster at a lower hierarchy to improve cluster quality.
- iii. To construct an enhanced cluster merging algorithm based on the clusters obtained from Objective (2) in order to obtain the optimal clusters.
- iv. To evaluate the proposed algorithms based on the performance metrics.

### **1.5 Research Significance**

In this research, an algorithm for hierarchical text clustering will be developed based on the Firefly Algorithm. The importance of the developed algorithm is demonstrated in the three objectives as described in Section 1.4, where the first objective provides means in identifying the initial centroids. The benefit of the second objective is in improving the cluster quality as the assigned document can be relocated (if required) into a newly created cluster. Additionally, the third objective contributes in identifying the number of clusters and enhances the performance quality.

The proposed hierarchical text clustering algorithm can be realized in a search engine that represents and organizes documents in a structured manner. A well-structured indexing database would contribute to a better retrieval process, and hence, facilitate users in decision making.

## **1.6 Scope and Limitations of the Research**

This study focuses on text clustering based on the behavior of Firefly Algorithm. In particular, this study looks into the adaptive characteristic of the Firefly Algorithm (FA) in grouping text documents.

In addition, this study focuses on text documents which are obtained from different resources; UCI machine learning repository (Bache & Lichman, 2013) and 20 Newsgroup website (20NewsgroupsDataSet, 2006). The datasets are of two types: balanced (the number of documents in each class is equal) and un-balanced. The 20Newsgroups dataset (20NewsgroupsDataSet, 2006; Bache & Lichman, 2013) and Reuters-21578 dataset (Lewis, 1999) are balanced datasets as each class in the datasets includes the same number of documents. Each document in the Reuters-21578 database contains only one topic that means every document refers to only one class. On the other hand, TR11, TR12, TR23 and TR45 were retrieved from CLUTO toolkit (Karypis, 2002), and have already been pre-processed by Zhao and Karypis (2001), and they were derived from Text Retrieval Conference (TREC) collections (TREC, 1999), is an un-balanced dataset. However, the number of terms in each resource was less than 10,000.

Furthermore, in this research, the focus is on the quality of the produced clusters rather than the consumed computational effort (time and resources).

## **1.7 Organization of the Research**

This thesis is organized in eight chapters. Chapter One contains the introduction that discusses Swarm Intelligence, text analytics, and information retrieval. Furthermore,



this chapter also includes problem statement, research questions, research objectives, significance and research scope.

The second chapter presents a proposed taxonomy of clustering methods. Existing literature on different categories of text clustering techniques are discussed and the utilization of the swarm approach in text clustering is focused.

Chapter Three provides the methodology used in conducting this research. It includes the architecture of the proposed hierarchical text clustering that includes: data acquisition, clustering using Weight-based Firefly Algorithm, cluster refining, and cluster merging.

Chapter Four presents the realization of the proposed clustering using Weight-based Firefly algorithm. The Weight-based Firefly algorithm is tested on the standard benchmark dataset that is mostly used in text clustering.

Chapter Five presents the Document Re-locating algorithm combined with Weight-based firefly algorithm. The proposed Document Re-locating algorithm changes the location of documents (if necessary) when new clusters are constructed.

Later, in Chapter Six the elaboration on the proposed merging algorithm is presented. Evaluation is performed based on external and internal metrics and compared against static and dynamic methods.

Chapter Seven includes experimental results; the evaluation and analysis of the proposed implementation of the adaptive FA that includes the entire prior proposed

algorithm (WFA, re-locating, and merging algorithm). The adaptive algorithm is evaluated using balanced and un-balanced datasets.

Finally, Chapter Eight gives the concluding remarks on the proposed hierarchical text clustering. It includes the research contribution and recommendations for future research work relating to Firefly algorithm.



## CHAPTER TWO

### LITERATURE REVIEW

#### 2.1 Introduction

Currently, there exist various methods on text clustering. These include the development of enhanced algorithms and hybridization of existing clustering algorithms. Diverse clustering algorithms have been presented over the years. Algorithms that are based on the initial information of a data collection (for example, the number of clusters) can be categorized into two approaches: static and dynamic (Gil-Garcia & Pons-Porrata, 2010). Furthermore, traditional methods are divided into five types: Partitional clustering, Hierarchical clustering, Density-based clustering, Model-based clustering and Grid-based clustering (Han et al., 2011; Zhang, Cao, & Lee, 2013). Figure 2.1 illustrates the proposed taxonomy of clustering methods. The following sections review the state of the art in text clustering based on the figure.

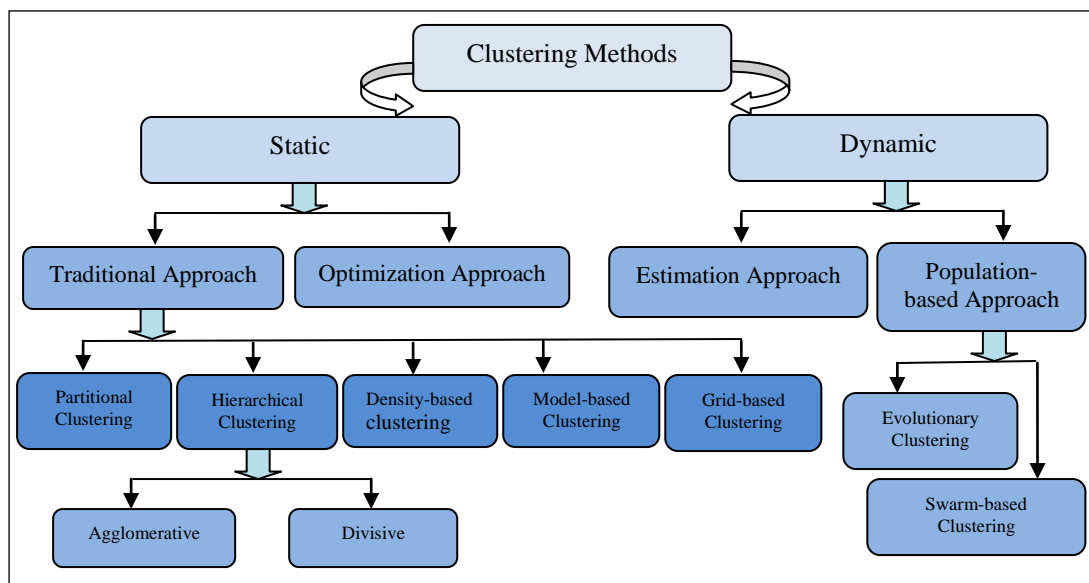


Figure 2.1. Proposed taxonomy of clustering methods

## **2.2 Clustering Methods**

In clustering, various clustering methods have been presented over the years to help optimize the field. These methods can be sighted in two separate types of groups: a) static methods, and b) dynamic methods, with their respective approaches from traditional methods to evolutionary ones. This chapter includes the elaboration on the description of several clustering methods categorized as a static or dynamic approach.

### **2.2.1 Static Approach**

Clustering involving static approach requires information on the number of clusters (i.e. the value of  $k$ ) prior to the clustering process. The static approach can further be categorized as either using the traditional clustering or optimization methods. To date, various optimization methods have been utilized to overcome issues that arise while using the traditional methods.

#### **2.2.1.1 Traditional Methods**

Traditional methods include five types of clustering: Partitional clustering (Jain, 2010), Density-based clustering (Sander, 2010), Grid-based clustering (Ilango & Mohan, 2010), Model-based clustering (Ding & Fu, 2012; Zhang & Cao, 2011), and Hierarchical clustering (Jain, 2010; Vijayalakshmi, MCA, & Devi, 2012). Details on these five types of methods are as presented in Sections 2.2.1.1.1 to 2.2.1.1.5.

### 2.2.1.1.1 Partitional Text Clustering

Partitional clustering algorithms divide a dataset into groups based on the inter-similarity between documents. The most popular and efficient partitional clustering algorithm is K-means (Hartigan & Wong, 1979; Hu, Zhou, Guan, & Hu, 2008; Jain, 2010) which was first introduced in 1957 by Hugo Steinhaus and was first utilized by James MacQueen in 1967. The K-means steps are shown in Figure 2.2.

#### **K-means algorithm**

**Step 1:** Select an initial partition with k clusters; repeat Steps 2 and 3 until cluster membership is stabilized.

**Step 2:** Generate a new partition by assigning each pattern to its closest cluster center.

**Step 3:** Compute new cluster centers.

*Figure 2.2. Steps of K-means algorithm*  
Resource. Jain (2010)

The implementation of K-means generates problems that include the randomly selected initial centroids. Existing studies (Gu, Zhou, & Chen, 2009; Hu, Zhou, Guan & Hu 2008; Mishra, Nayak, Rath, & Swain, 2012; Poomagal & Hamsapriya, 2011; Singh & Bhatia, 2011; Yao, Pi & Cong, 2012) indicate that different initial centroids produce different quality of clusters. Hence, this indicates that it is important to accurately determine the initial centroids.

With this, researches have been improving the standard K-means for text clustering by selecting initial cluster (Xinwu, 2010; Yao, Pi & Cong, 2012). A work by Xinwu (2010) used the sampling method (random sampling) on online datasets to validate the proposed algorithm. The results on F-measure scattered from 0.60 and 0.75 in the

standard K-means; while in the improved K-means, the F-measure results scattered from 0.75 and 0.85, which are better than the one obtained using the standard K-means. The obtained results were similar to the findings by Yao, Pi and Cong (2012) which were performed on Chinese corpus from Sogou news site and used isolated texts. Nevertheless, the number of k cluster was assumed.

The modified K-means algorithm for document clustering was presented by Poomagal and Hamsapriya (2011), and Singh and Bhatia (2011). Singh and Bhatia (2011) proposed that the highest frequency of points has the highest probability to be included as the centroid. Their approach reduced the computational time and minimized the complexity, but the work was not implemented on a real dataset and clustering measurements were absent to validate the proposed algorithm. This differs with the work by Poomagal and Hamsapriya (2011), who calculated the midpoint for each term. The researchers utilized 200 results that were selected from Yahoo, Google and Bing. The experimental result demonstrated less intra-cluster and high inter-cluster than the other two algorithms of Snippets and URL and tag contents. However, the drawbacks of these two works are the number of k clusters which was assumed.

A novel partition-based algorithm for text clustering to solve the initial centers of cluster was presented by Hu, Zhou, Guan and Hu (2008), and Wang, Liu, Chen, and Tang (2011). A constrained K-means clustering method, named S3-Kmeans, integrates prior knowledge of documents (pairwise constraints ML terms must link) into Euclidean distance function of K-means (Hu, Zhou, Guan & Hu, 2008). In the evaluation, they used two metrics: normalized mutual information (NMI) and

corrected rand coefficient (CRC). The proposed methods of S3-Kmeans performed better than K-means, Single Linkage and improved Single Linkage. However, the shortcoming is in determining the number of k clusters and this is similar to the weakness presented by Wang, Liu, Chen and Tang (2011), in which they calculated the document average similarity of document k, then, found the document similarity list based on average similarity, and later, they ordered the list and retrieved the largest value as centroids. The F-measure result of the proposed algorithm is better than Agglomerative, Bisect K-means and Graph-based in CLUTO (Karypis, 2002).

Existing researches on k-means are based on initial point and this leads to coverage local minima. This problem was solved by Yang (2010), who proposed a fast greedy k-means algorithm. In the experiment, they used a dataset that includes 33,409 web documents gathered from the Waterloo University website. The precision result compares k-means against the fast greedy k-means and it is learned that the proposed method has better precision result than K-means. However, the shortcoming of the work is the pre-defined value for k.

To minimize the processing complexity for text document clustering, Guan et al. (2011) introduced two contributions: i.e. a similarity formula and a new Seed Affinity Propagation (SAP). A similarity formula is based on three feature sets: co-feature set represents the feature in two objects; unilateral set represents the feature in one object not belonging to another object; and significant set represents the most important feature in one object belonging to another object. In the experiment conducted, the researchers used the Reuters-21578 dataset with three measurements: Entropy, F-measure and CPU time. The results of the proposed SAP outperformed

the original Affinity Propagation AP(C), SAP(C) and K-means(C) that used cosine similarity and AP that used Tri-set similarity in F-measure with 0.599 and in Entropy with 0.472. However, in CPU execution time, the Affinity Propagation AP(C) has the best time of 12.6 ms. and yet, the evaluation was only based on external measurement and does not include the compactness of the clusters.

The problem of initial cluster centers was solved using various methods in numerical datasets (Gu, Zhou & Chen, 2009; Mishra, Nayak, Rath & Swain, 2012). In the first work, Gu, Zhou and Chen (2009) proposed to refine the initial centers of the K-means method by utilizing the partition algorithm twice. In the evaluation phase, two datasets are employed: breast cancer and Iris from the UCI repository. The results indicated that the proposed refined K-means is better than the standard K-means in CPU-time, but the clusters' quality performance is absent. This is different than the work of Mishra, Nayak, Rath and Swain (2012), who selected the farthest distance between two pairs and found that the computational time of standard K-means is better than the proposed far efficient K-means. In addition, the result of Dunn's index (DI) in the proposed far efficient K-means is 0.047 and Bouldin's index (DBI) is 0.688 which is better than the standard k-means. However, the number of k cluster is provided by the user.

The aforementioned researches contribute in random initial centroids selection in K-means algorithm, but the quality of the clusters can still be improved (Hu, Zhou, Guan, & Hu, 2008; Mishra, Nayak, Rath, & Swain, 2012; Poomagal & Hamsapriya, 2011; Singh & Bhatia, 2011; Yang, 2010; Yao, Pi & Cong, 2012). There is still a gap in the existing literature which is the determination of the number of k clusters and



initial centers in which this research tries to solve them. Table 2.1 presents the summary of existing works in partitional text clustering.

Table 2.1

*Summary of existing researches in partitional text clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Hu, Zhou, Guan, & Hu (2008)	A constrained K-means clustering method, S3-Kmeans	Random initial centroids selection in k-means algorithm	TREC, Reuters-21578, Newsgroup-20 and WebACE.	Predefine k clusters
Xinwu (2010)	Improved K-means sampling method	Initial selection of centers	Used online datasets to validate the algorithm	Predefine k clusters
Yang (2010)	A fast greedy K-means algorithm	Existing researches for K-means not scaled to large data point numbers and are slow depending on initial point which lead to coverage local minima	33,409 web documents gathered from Waterloo University website	Predefine k clusters
Poomagal & Hamsapriya (2011)	Determination of initial centroids by calculating the midpoint using optimized K-means	Random initial centroids selection in K-means algorithm	Used 200 queries in Yahoo, Google and Bing	Predefine k clusters
Singh & Bhatia (2011)	Modified K-means algorithm	Initial center of clusters	No real dataset	Predefine k clusters.
Wang, Liu, Chen, & Tang (2011)	Partition algorithm	Sensitivity of partition algorithm for initial centroids	20 Newsgroups, Reuters-21578 and two Chinese datasets	Predefine k clusters

Table 2.1 continued

Guan, Shi, Marchese, Yang, & Liang (2011)	A new similarity formula, Seeds, and a new Seed Affinity Propagation (SAP)	Minimized the processing complexity	Ruters-21578	Only the external evaluation are conducted and not measured the compactness of clustering
Yao, Pi, & Cong (2012)	Improved K-means for Chinese text clustering	Improved initial centers by isolated text	Corpus from Sogou news site	Predefine k clusters

Table 2.2

*Summary of existing researches in partitional numerical clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Gu, Zhou, & Chen (2009)	K-means with refined initial center algorithm.	The members of clusters are unstable for a large sample data, besides, the initial selection of seed points	Breast cancer and Iris	Only measures computational time, but clusters quality performance is absent
Mishra, Nayak, Rath, & Swain (2012)	Far Efficient K-means algorithm	Finding initial cluster centers	UCI (Iris, Wine and Abalone)	Predefine k clusters

#### 2.2.1.1.2 Density-based Text Clustering

Density-based clustering algorithm is a technique to construct clusters based on the dense regions of objects in high-dimensional space that are isolated by low density areas. Density-Based Spatial Clustering of Application with Noise (DBSCAN) is the

most known density-based method used for data clustering (Ester, Kriegel, Sander, & Xu, 1996). It randomly selects points and finds the neighborhood by using query. A cluster is constructed based on these points and each neighbor examined to see if it can be included in the cluster (Chehreghani, Abolhassani, & Chehreghani, 2008). This type of clustering is useful for this study in constructing clusters which is based on similarity threshold.

A new density-based clustering method for web data was proposed by Chehreghani, Abolhassani, and Chehreghani (2008). The proposed method included three stages: the insertion stage, the extraction stage and the combined stage. It solved the shortcoming of density-based methods to cluster web data. Three datasets were used from DMOZ collection, News collection and Reuter's documents. The results of the proposed method were better than K-means, Single Linkage and improved Single Linkage in two measurements; purity and F-measure. The result of the number of examined data items was compared with DBSCAN (Ester, Kriegel, Sander, & Xu, 1996).

Improving clustering solution by assigning weight to documents and studying the density-based functions' performance in three parts, internal, external and hybrid, was proposed by Aliguliyev (2009a, 2009b). The researcher developed weighted clustering functions and un-weighted clustering functions, and they used a modified differential evolution algorithm to optimize these functions. In the evaluation stage, five datasets were used, namely Reuters-21578, WebACE, TREC-5, 20Newsgroups and WebKb. The result showed that all twelve criterion functions produced accuracies of 80% and 77%, except for one weighted function which was most

sensitive. The sensitive function produced 42.17% for inter cluster, 71.65% for intra cluster, 0.40% for purity, 0.18% for Entropy, 7.47% for Mirkin, 4.16% for F-measure, 15.36% for variation information, and 0.02% for V-measure.

A review on existing document clustering methods that used frequent patterns and a method called Maximum Capturing was presented by Zhang, Yoshida, Tang, and Wang, (2010). The researchers developed three Maximum Capturing methods that depend on three similarity measures. The researchers solved three problems; the first one was the similarity between two documents, the second determines the appropriate number of clusters, and the third was achieving clustered documents exactly the same as natural clusters. The experimental result showed that Maximum Capturing performed better than CFWS, CMS, FIHC and FTC in F-measure value on two benchmark datasets, Reuters-21578 and Chinese corpus.

To sum up the foregoing research (Aliguliyev, 2009a, 2009b; Chehreghani, Abolhassani, & Chehreghani, 2008; Zhang, Yoshida, Tang, & Wang, 2010), it can be concluded that Aliguliyev (2009a, 2009b) improved the clustering solution by designing weight clustering functions and validating performance. Furthermore, the work of Chehreghani, Abolhassani, and Chehreghani (2008) improved web pages hierarchical clustering using density-based methods. Apart from that, Zhang, Yoshida, Tang, and Wang (2010) succeeded in solving the problem of similarity between two documents. However, the number of k clusters is predefined in all the previous researches.

### 2.2.1.1.3 Grid-based Text Clustering

The grid-based clustering approach uses a multi-resolution grid data structure. It divides data space into several levels of cells. The process of clustering is performed inside these cells. The parameters of higher level cells can be computed using lower level cells. The quality of clustering is based on the number of cells in lower level cells. If it is too coarse, this will lead to the quality of cluster being reduced (Han & Kamber, 2006). In addition, grid clustering does not have a relationship between neighbors, which means there are no children and parent cells to be represented hierarchically. There are various recognized grid-based approaches such as STING (Wang, Yang, & Muntz, 1997), CLIQUE (Agrawal, Gehrke, Gunopulos, & Raghavan, 1998), and OptiGrid (Hinneburg & Keim, 1999).

The advantage of grid clustering is that it does not require input parameters and this solved the problem of parameter sensitivity in hierarchical clustering (Yue, Wei, Wang, & Wang, 2008). An approach of grid clustering was built based on the combined idea of divisive and agglomerative hierarchical clustering GGCA. The approach bisects the grid into two grids that are equal in size. This process is the same process of divisive clustering. The output of this process is the optimal grid size and the determination of core grid. The core grids are later merged. The merging process is the same as implemented in agglomerative clustering. The proposed approach was tested on eight datasets, five artificial and three real datasets. The researchers measured the robustness of the clustering, runtime, error and determination of cluster number and compared them against three approaches, HCM, Clique and Shift. For artificial datasets, HCM is learned to be not suitable. Shift is

appropriate to determine the cluster number, but produced high error rates. Clique divides the dataset into many clusters that are meaningless. The proposed approach, GGCA, outperformed in CPU runtime and cluster number and obtained less errors compared to the other methods. On the other hand, for real datasets, GGCA produces clusters that are the same as the original clusters, but with less CPU time. The proposed approach was robust in clustering two types of datasets, but their research was not evaluated on text datasets.

The problem of high dimensionality data in density-based clustering and computational time was solved by suggesting a grid-density clustering for large datasets (Zhao, Cao, Zhang, & Zhang, 2011). The suggestion included four features: dealing with objects as atomic units; dealing with neighbors as a couple of groups; density compensation; and finally minimal subspace distance. During experimentation, they used synthetic and public data. The result of the proposed AGRID+ approach using synthetic data was compared with AGRID in terms of density (that is based on threshold) and compared with NAIVE, IORDER and AGRID in terms of accuracy and CPU time. Whereas, the result of the proposed AGRID+ approach using public data was compared with Random Projection and IORDER in terms of Conditional Entropy (CE) and Normalized Mutual Information (NMI). For density comparison, AGRID+ is better than AGRID because the proposed approach detected more objects. The accuracy in AGRID+ was 93.7%, IORDER was 85.3%, AGRID was 83.1% and NAIVE was 95.0%. This means that NAIVE is more accurate than the proposed AGRID+ despite the CPU time for the proposed AGRID+ was 4.60, less than NAIVE's 44.21. The CE in AGRID+ was

0.466, less than IORDER's 0.517 and Random Projection's 0.706. Moreover, NMI in the proposed AGRID+ was 0.845, which is higher than IORDER's 0.822 and Random Projection's 0.790. This means the proposed AGRID+ approach is a high quality clustering, but is not implemented on document clustering.

From the aforementioned researches that solved the problem of parameter sensitivity in hierarchical clustering (Yue, Wei, Wang, & Wang, 2008), and the problem of high dimensionality of data in density-based clustering and computational time (Zhao, Cao, Zhang, & Zhang, 2011), It can be concluded that the two approaches were robust and accurate in the clustering of datasets, but do not measure the robustness in text datasets.

#### **2.2.1.1.4 Model-based Text Clustering**

Model-based clustering is the method that tries to optimize the fit between some mathematical models and data. The Self Organizing Map (SOM) is a type of model-based clustering and Neural Network which was presented by Kohonen (1998, 2001). The SOM algorithm has been used in many applications, like semantic map, clustering, and so on (Yin, Kaku, Tang, & Zhu, 2011). Model-based clustering is sensitive to the initial selection of weight vector, as well as to its different parameters, such as the learning rate and neighborhood radius (Rokach & Maimon, 2005).

The problem of text clustering with high dimensional features was solved by Liu, Wu, and Liu (2011). The researchers used two processes to maintain the text clustering system with high efficiency, which are semantic quantization and feature

extraction. These processes work in the offline stage. The online stage used fast similarity and incremental clustering. In the conducted experiment, the researchers used two datasets, which are 20Newsgroups and news web documents, retrieved from the Internet. For measuring the performance of fast SOM, F-measure is used in the 20Newsgroups dataset, while accuracy and CPU time is used in news web documents (because the datasets were large). The accuracy and CPU time result of fast SOM were compared with another method that used vector space model (VSM). Furthermore, the F-measure result is compared with clustering system, GHSOM. CPU time for fast SOM was 0.13, less than the method that used vector space model which was 4.70 in 50,000 documents. When the document number was increased, the time also increased. The accuracy of text clustering in both datasets produced best results of 0.85 and 0.82 when the documents that were used in clustering were 85% and 80% from all the datasets and the documents that were used in incremental clustering were 15% and 20%. This work is very efficient in large datasets using SOM, but needs more improvement by the implementation with other methods.

Ding and Fu (2012) constructed two maps using Self Organizing Maps (SOM) to solve the problem of easily retrieved relevant documents in search engines and enhance the search process combining word map with document map. The first map is word map that clusters the words which appear in one document into one neuron. The second one is document map that clusters similar documents into one neuron. Similar documents are measured depending on similar words between documents. The research finds that integrating two maps will increase the retrieval of relevant documents by word search. The disadvantage of this work is that it does not use any



real or synthesis dataset and also does not use any measurements to evaluate the work.

The researches that have been addressed in this section have several weaknesses and advantages in work and findings (Ding & Fu, 2012; Liu Wu & Liu, 2011). Liu Wu and Liu (2011) stated that the advantage of their work was that it is very efficient in large datasets using SOM, but needs more improvement by implementing it in other methods. Ding and Fu (2012) identified the advantage of their proposed approach as the increase of the retrieval of relevant documents by word search, but the shortcoming is the proposed approach not realized in any real or synthesis datasets and is also not evaluated or compared with other researches.

#### **2.2.1.1.5 Hierarchical Text Clustering**

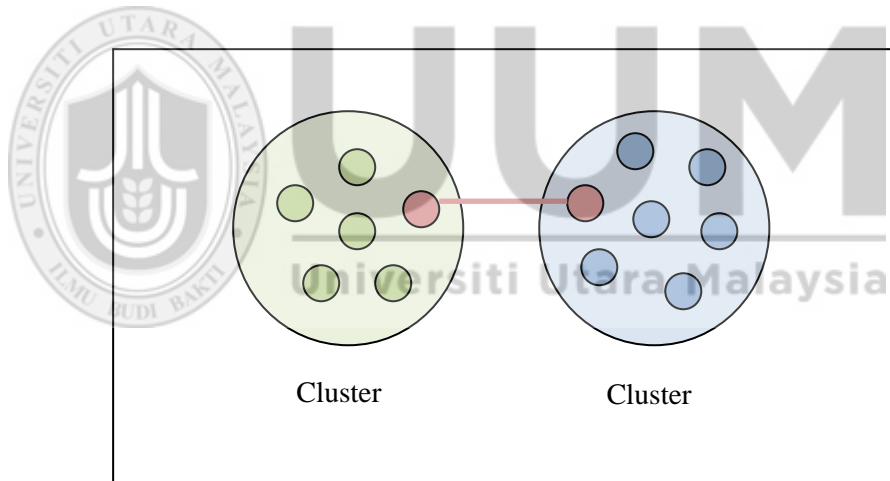
Hierarchical clustering constructs a hierarchical structure for text documents. There are two types of hierarchical clustering: agglomerative clustering and divisive clustering (Rafsanjani, Varzaneh, & Chukanlo, 2012).

#### **Agglomerative Clustering**

Agglomerative clustering starts with multi-clusters; each cluster includes one document or more in which two clusters are later merged using the merging algorithms. There are three agglomerative clustering merging algorithms: The Single Linkage Hierarchical Clustering (SLHC), the Complete Linkage Hierarchical Clustering (CLHC) and the Average Linkage Hierarchical Clustering (ALHC) or

called UPGMA (Manning, Raghavan, & Schütze, 2008; Yin, Kaku, Tang, & Zhu, 2011).

The Single Linkage Hierarchical Clustering (SLHC) is a simple agglomerative clustering that depends on the similarity between two objects. SLHC merges two objects that have a high similarity or have the least amount of distance. The two clusters are merged based on a single link between the two elements in different clusters that have the shortest distance or highest similarity. SLHC is sensitive in dealing with noise and outliers (Tan, Steinbach, & Kumar, 2006). Figure 2.3 illustrates the Single Linkage Hierarchical Clustering (SLHC).



*Figure 2.3. The Single Linkage Hierarchical Clustering (SLHC)*

Resource. Manning, Raghavan, and Schütze (2008)

SLHC for merging two clusters is evaluated based on the similarity between objects where the maximum similarity will be chosen. The formula of merging is based on cosine similarity as shown in Equation 2.1.

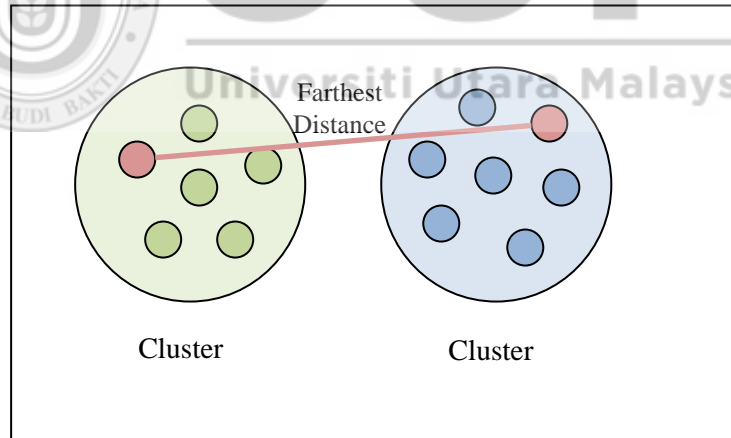
$$SLHC_{D_i, D_j} = \text{Max}_{i, j \in C} \text{Similarity}(D_i, D_j) \quad (2.1)$$

Where:  $D_i, D_j$  are documents in two different clusters  $i$  and  $j$ ,  $C$  is all of the clusters.

The formula of merging is based on distance similarity as shown in Equation 2.2.

$$SLHC_{D_i, D_j} = \min_{i, j \in C} \text{Distance}(D_i, D_j) \quad (2.2)$$

On the other hand, the Complete Linkage Hierarchical Clustering (CLHC) merges two objects with minimum similarity or with maximum distance which is a reverse of SLHC (Manning, Raghavan, & Schütze, 2008). The two clusters are merged based on all links between any two elements in different clusters and the highest distance or lower similarity is chosen. CLHC is less vulnerable to noise and outliers, but it can break large groups and prefers spherical shapes (Tan, Steinbach, & Kumar, 2006). Figure 2.4 illustrates the Complete Linkage Hierarchical Clustering (CLHC).



*Figure 2.4. The Complete Linkage Clustering Hierarchical (CLHC)*

Resource. Manning, Raghavan, and Schütze (2008)

For merging two clusters, CLHC is evaluated based on the similarity between objects and the farthest distance is chosen. The formula of merging is based on cosine similarity as shown in Equation 2.3.

$$CLHC_{D_i, D_j} = \text{Min}_{i, j \in C} \text{Similarity} (D_i, D_j) \quad (2.3)$$

The formula of merging is based on distance similarity as shown in Equation 2.4.

$$CLHC_{D_i, D_j} = \text{Max}_{i, j \in C} \text{Distance} (D_i, D_j) \quad (2.4)$$

The Un-weighted Pair Group Method with Arithmetic Mean (UPGMA) is one of the most popular agglomerative clustering algorithms that is used for merging two clusters (Manning, Raghavan, & Schütze, 2008; Yujian & Liye, 2010). UPGMA is based on the average similarity between all elements in two clusters. The advantage of this method is that it can transact with dynamic data sets and does not allow for overlapping (Gil-Garcia & Pons-Porrata, 2010). However, the weakness of UPGMA is the time complexity (Murugesan & Zhang, 2011a, 2011b). Figure 2.5 illustrates the UPGMA clustering method.

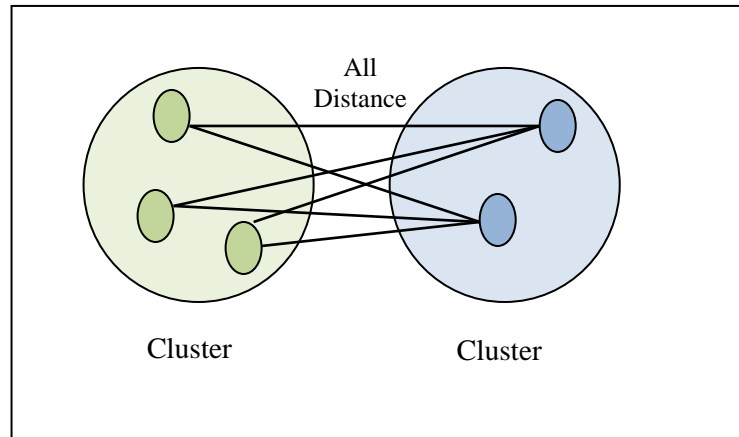


Figure 2.5. The Un-weighted Pair Group Method with Arithmetic Mean (UPGMA) Resource. Manning, Raghavan, and Schütze (2008)

The formula of merging two clusters based on cosine similarity is shown in Equation 2.5.

$$UPGMA_{i,j} = \frac{\sum_{i \in C} \sum_{j \in C} \text{Similarity}(D_i, D_j)}{N_i N_j} \quad (2.5)$$

Where:  $N_i$  is the number of documents in cluster  $i$ ,  $N_j$  is the number of documents in cluster  $j$ . The formula of merging two clusters based on distance similarity is shown in Equation 2.6.

$$UPGMA_{i,j} = \frac{\sum_{i \in C} \sum_{j \in C} \text{Distance}(D_i, D_j)}{N_i N_j} \quad (2.6)$$

Yujian and Liye (2010) presented an improved Un-weighted Multiple Group Method with Arithmetic Mean (UMGMA) to solve the problem of tie trees (two or more trees created from analyzing related populations) in UPGMA. The result enhances UPGMA and produces a unique tree. However, this process needs high computational time. Murugesan and Zhang (2011a, 2011b) proposed to utilize UPGMA to refine the clusters that are generated by Bisect K-means and to reduce the time complexity of UPGMA. The result of the proposed method outperformed Bisect K-means in three performance metrics. In CHAMELEON (Karypis, Han, & Kumar, 1999), two phases of clustering were proposed; the first phase groups the data based on the graph partition algorithm, the second phase utilizes the agglomerative clustering algorithm to detect the real clusters.

Recently, several researches integrated partitional clustering with hierarchical clustering to produce better clusters, and at the same time, solve the problem of time complexity of hierarchical clustering. Hybrid partitional with hierarchical agglomerative for document clustering method utilizes the feature of partitional clustering to handle large datasets efficiently and the feature of agglomerative to construct a hierarchical structure for documents ( Murugesan & Zhang, 2011a, 2011b; Zhu, Fung, Mu, & Li, 2008).

Zhu, Fung, Mu, and Li (2008) grouped the document collection into specific clusters based on function that maximizes the sum of average pair-wise similarities between documents, and then splits the lower average pair-wise similarities into the same specific clusters until the last cluster includes the limited document. Then, the merging is performed based on internal closeness and internal inter-connectivity which are adopted from Kalman (1960). The result of the proposed approach showed lower entropy (0.261) and higher purity (0.7860) and F-measure (0.789), for the TR12 dataset, and the merging is better than UPGMA. However, the number of clusters is predefined.

In conclusion, there is no doubt that the aforementioned researches (Murugesan & Zhang, 2011a, 2011b; Yujian & Liye, 2010; Zhu, Fung, Mu, & Li, 2008) contribute to clustering, yet there is room to improve the cluster quality and to predict the cluster number.

## **Divisive Clustering**

Divisive text clustering starts with a single cluster that contains all documents and splits them into a number of clusters. The Principal Direction Divisive Partitioning (PDDP) is one of the important divisive hierarchical methods (Boley, 1998). PDDP builds a binary tree where the root contains the entire set of documents and the leaf contains the output clusters. The root is divided into two partitions; the right partition is for cluster more than 0 and the left partition is for less than 0. The results using the PDDP algorithm on 185 documents retrieved from the World Wide Web in two scaling norms and TFIDF were compared with the ones from the agglomerative algorithm. The Entropy result for PDDP was 0.69 in norm scale, which is better than the agglomerative algorithm. In addition, the CPU execution time is better in PDDP.

Bisect K-means is a well-known divisive hierarchical clustering and is a variant of K-means (Kashef & Kamel, 2009; Murugesan & Zhang, 2011a, 2011b). In this algorithm, at each level of constructing a hierarchy, Bisect K-means selects one cluster C (initially C represents the whole dataset) and classifies the objects in C into two partitions (C1 and C2) by randomly choosing two centers and assigning objects to the closest centers (using the K-means algorithm). This process continues until it reaches the stopping condition of either the number of iterations or specific number of clusters. At each step of classifying, the chosen cluster is tested by some criteria: a) minimum intra similarity; b) larger cluster size (cluster includes higher number of objects); or c) size of cluster and similarity (Kashef & Kamel, 2009; Murugesan & Zhang, 2011a, 2011b). Figure 2.6 shows the step-by-step process of Bisect K-means.

**Bisect K-means algorithm**

**Step 1:** Randomly choose two cluster centers.

**Step 2:** Cluster using K-means.

**Step 3:** If number of clusters is not reached, choose the cluster that has smallest intra similarity.

**Step 4:** Repeat Step 1 until number of clusters is reached.

*Figure 2.6.* The process of Bisect K-means

Resources. Kashef and Kamel (2009), and Murugesan and Zhang (2011a, 2011b)

Bisect K-means requires a refinement step to re-cluster the resulting solutions at each level of constructed tree. This drawback attracts researchers to combine Bisect K-means with K-means. In the work of Kashef and Kamel (2009, 2010), the clustering solution of Bisect K-means (also known as BKM) and K-means at each level is cooperated between them by cooperative and merging metrics, named CBKM, for clustering text, artificial and gene datasets. The result from BKM was revised using K-means (KM) at every stage of the construction of the binary tree. In each level, the process of CBKM includes three stages: general clustering by K-means and BKM, the cooperation or intersection stage between K-means and BKM, and the merging stage. For documents dataset, the quality percentage result of F-measure for the proposed approach on the SN dataset was 31.05%, correspondingly, the result of Entropy was 37.13% on the UW dataset, the result of Purity was highest in 20NG dataset, which was 39.25%, the result of NMI was highest in the 20NG dataset, which was 51.48%, and the result of SI was also highest in the 20NG dataset, 38.41%. In addition, the time complexity of the proposed CBKM was better than the



SL algorithm. This proposed approach improved the quality of clusters; nevertheless, the number of k cluster is still pre-defined.

UPGMA (a type of agglomerative clustering) merges the obtained clusters from Bisect K-means (where, Bisect K-means generates clusters larger than k) until it reaches the k number of clusters (Murugesan & Zhang, 2011a, 2011b). In the Bisect K-means stage, the document collection groups into a cluster greater than the predefined cluster. Then, the created centroids are passed from the first stage to UPGMA stage to merge them and produce k clusters. The result produced an average Entropy of 1.410, average F-measure of 0.29 and average purity of 0.49 for the Reuters dataset. Such a result is better than the ones obtained by Bisect K-means. However, Bisect needs a refinement to re-cluster the resulting solutions and needs to define k number of clusters.

Bordogna and Pasi (2012) constructed a divisive hierarchical clustering using Fuzzy technique to solve the problem of diversification of topics in information retrieval. A cluster is split into sub-clusters based on multi dimensions evaluation such as cohesion, mass cardinality and fuzziness. In the undertaken experiment, two datasets were used (20 Newsgroup and Reuters RCV1) and the measurements were the MC index and Rand Index (RI). The proposed Fuzzy clustering was compared with EM, FCM and hierarchical Bisect K-means. The result on the average MC index was 0.77 using the Reuters RCV1. The result of Rand Index was 0.93 using the 20Newsgroup. The proposed Fuzzy approach outperformed other methods and is suitable for large datasets because of faster convergence. However, the result of this work may contain irrelevant information retrieved to users.

The determination of the number of clusters was proposed in two different ways (Gupta & Sharma, 2010; Ye, Gauch, Wang, & Luong, 2010). A map between documents and concepts was constructed, and then five methods which are already implemented in CLUTO (Ye, Gauch, Wang, & Luong, 2010) are used. These methods are repeated bisection, optimized RB, direct clustering, agglomerative and graph. These methods are utilized with different numbers of cluster from (1-10) with three tasks (each task is based on the CiteSeer database with known categories of numbers). The best cluster quality is chosen to compare the predicted number of cluster with the original number. The result demonstrated that direct clustering has a better value of maximum Purity and minimum Entropy compared to other methods. However, most methods used require the number of  $k$  cluster. Gupta and Sharma (2010) selected the first document as the cluster center and identified documents that are similar to it. Later, the similar document takes the role of the first document to find more similar documents until there are no more similar documents and the cluster is completed. Such an approach consumes time, as the numbers of clusters increases, the execution time of the algorithm will also increase. In addition, the proposed algorithm has weaknesses in evaluation measurements and was not implemented on real datasets.

In 2010, two dynamic clustering algorithms were also proposed, which are the dynamic hierarchical compact (DHC) that created disjoint clusters, and the dynamic hierarchical star (DHS) that produced overlapping clusters (Gil-Garcia & Pons-Porrata, 2010). The two proposed algorithms were evaluated using fifteen benchmark dataset collections. In the evaluation phase, the researchers used overall

F-measure and average BCubed metrics which were proposed by Amigo, Gonzalo, Artiles, and Verdejo (2009). The results of the two new algorithms were compared against UPGMA and BKM. The results indicated that the overall F-measure for DHC was 0.631, 0.662 for DHS, 0.677 for BKM and 0.678 for UPGMA in all collections. Hence, such a result indicates that UPGMA is a better method than the others. Furthermore, the result of average BCubed for DHC was 0.426, DHS was 0.340, BKM was 0.236 and UPGMA was 0.038 in all collections. This result indicated that the proposed method DHC was the best method compared to the other methods. Nevertheless, the weakness is in validation as the stability of the algorithm was not determined.

Cao and Yang (2010) proposed a k-medoids that depends on CF-tree, termed as CFK. The research solved the problem of k-medoids with scalability of large dataset and time complexity. It employs the idea of radius or diameter of cluster to control the cluster boundary. In the experimental work, the DSI dataset, used in CHAMELEON hierarchical clustering (Karypis, Han, & Kumar, 1999) was utilized. The result indicated that run time of CFK is better than the standard k-medoids and it also produces better clustering quality. However, the weakness lies in the tree structure as the CF-tree does not work well when the cluster shape is not spherical.

Lahane, Kharat, and Halgaonkar (2012) proposed a divisive approach for educational data clustering. The approach includes two phases: the first phase splits the original cluster into two clusters and then measures the homogeneity (based on intra-cluster and inter-cluster homogeneity). The second phase checks for the stability of the cluster by changing the location of one member of a cluster to another. If the quality

(homogeneity) produced from changing the members is better than the existing homogeneity, the change is performed, or else it remains in the same cluster. The experimental result showed high quality clusters were created. However, changing the members from one cluster to another in each split partition is time consuming and generates memory overflow.

Concluded from previous researches (Bordogna & Pasi, 2012; Gil-Garicia & Pons-Porrata, 2010; Gupta & Sharma, 2010; Kashef & Kamel, 2009; Murugesan & Zhang, 2011a, 2011b), existing works have drawbacks in the evaluation of the proposed algorithms. On the other hand, Boley's work (1998) has problems in relocating documents. Additionally, Forsati, Mahdavi, Shamsfard, and Meybodi (2013) mentioned that existing hierarchical clustering methods do not relocate documents. Hence, such approach will affect the clustering quality, especially the ones on the higher level. Table 2.3 and Table 2.4 present the summary of existing researches in hierarchical text clustering and hierarchical numerical clustering.

Table 2.3

*Summary of existing researches in hierarchical text clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Boley (1998)	A new divisive algorithm called PDDP	The large data scalability	185 documents from WWW	Fixed clusters – cannot re-locate
Zhu, Fung, Mu, & Li (2008)	Hybrid partition with agglomerative	Agglomerative clustering has problems with high dimensionality data space	Used three datasets: TR12, Re1 and WAP	Predefined k cluster
Kashef & Kamel	Cooperative approach between Bisect K-	To enhance the Bisect K-means	Nine different datasets: two	Predefined k cluster

Table 2.3 continued

(2009)	means and K-means named CBKM	BKM to construct and refine better cluster result	artificial, four documents and three gene expression	
Ye, Gauch, Wang, & Luong (2010)	Construct map between documents and concepts. Then utilize five methods to predict the number of clusters for CiteSeer	Most existing works not include construct mappings between concepts and documents and also the time and precision	CiteSeer	Predefined k cluster
Gil-García & Pons- Porrata (2010)	Two dynamic clustering algorithms were proposed, which are dynamic hierarchical compact DHC and dynamic hierarchical star DHS Partitional algorithm to split the documents to	Created disjoint clusters and produced overlapping clusters	Fifteen benchmark text collections obtained from Karypis	Calculation of the algorithm stability
Gupta & Sharma (2010)	clusters and applied the proposed clustering algorithm in hierarchical clustering	Large size of index file	No real dataset	High computational time
Murugesan & Zhang (2011)	Bisect K-means clustering algorithm combined with UPGMA	Bisect K-means generates better clusters. Second problem is the time complexity of UPGMA	Ten datasets from Karypis datasets	Bisect K-means needs a refinement to re-cluster the resulting solutions and needs to define k number of clusters
Bordogna & Pasi	Construct a divisive hierarchical clustering	Diversification of topics in	20 Newsgroup and Reuters	Low retrieval result

(2012)	using Fuzzy	information retrieval	RCV1
--------	-------------	--------------------------	------

Table 2.4

*Summary of existing researches in hierarchical numerical clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Yujian & Liye (2010)	Improving Un-weighted Multiple Group Method with Arithmetic Mean UMGMA.	Tie trees in UPGMA that produced two or more tree from analyses related populations	Drosophila_Adh. meg and mtDNA haplotypes.	High computational time
Cao & Yang (2010)	A k-medoids depending on CF-tree.	Scalability of large dataset and time complexity	DS1 dataset which was used in CHAMELEON hierarchical clustering	CF-tree does not work well when the cluster shapes are not spherical
Lahane, Kharat & Halgaonkar (2012)	Proposed divisive clustering for education data	Clustering high dimensional categorical data	Educational Data contains 50 instances	Time consuming and memory overflow

### 2.2.1.2 Optimization Methods

Optimization can be defined as the identification of the optimal or near optimal solution (best solution) from all appropriate solutions. Finding the best solution is identified by formulating an objective function (minimum or maximum function), where, the objective function is designed depending on the problem in-hand (Rothlauf, 2011). Optimization problems include two categories based on the variables whether discrete or continuous. When the variable is discrete, the

optimization problem is called a combinatorial optimization problem. In a combinatorial optimization problem, searching for the best object is from a finite set in dimensional space (object such as integer or graph). To find the optimal solution for the system, we must design an objective function (fitness function). The objective function is the minimizing or maximizing value. This function needs parameters which can be included in the analysis of the problem. From these parameters, the function returns values which makes the system response optimal (Das, Abraham, & Konar, 2009).

Optimization algorithms (as in Figure 2.7) are divided into Traditional Methods (exact) and Modern Heuristics (approximate).

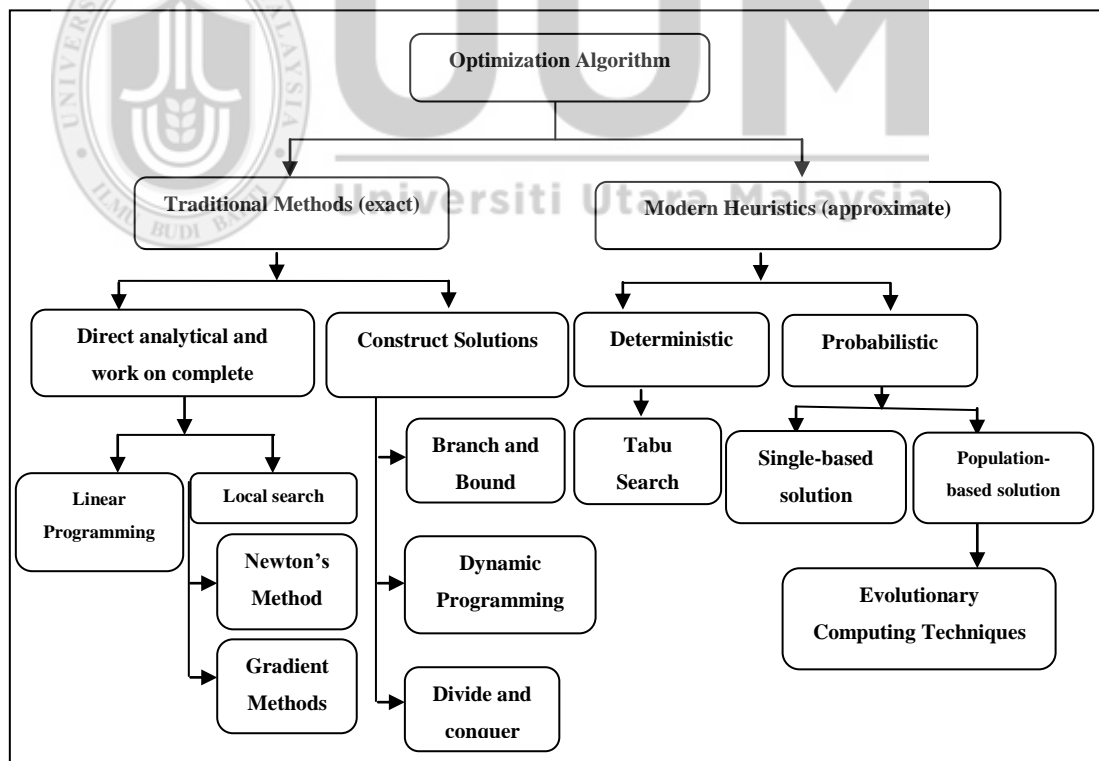


Figure 2.7. The taxonomy of optimization algorithms

Resource. Das, Abraham, and Konar (2009, p.27)

Exact search algorithms include two classes: the first constructs solutions during the search in which it can find the optimal solution in the bounded time for a finite problem such as branch and bound, dynamic programming and divide and conquer. Another class of exact search works on a complete solution which is split into two subclasses as linear programming and local search. On the other hand, the heuristic search algorithms identify a good solution, but are non-optimal in less execution time.

Modern heuristic search algorithms (meta-heuristics) are divided into two classes: deterministic and probabilistic. Deterministic approach includes Tabu search (Glover, 1986), while probabilistic approach can be categorized as either single-based meta-heuristics solution or population-based meta-heuristics (Boussaïd, Lepagnot, & Siarry, 2013; Das, Abraham & Konar, 2009; El-Abd & Kamel, 2005).

Over the years, meta-heuristic approach has proven successful to find the best solution in many disciplines (Beasley, Bull, & Martin, 1993; Cui, Potok, & Palathingal, 2005; Glover, 1986; He, Hui, & Sim, 2006; Kirkpatrick, Gelatt, & Vecchi, 1983). The current meta-heuristic approach can be classified into two categories: single meta-heuristic solution and population meta-heuristic solution (Boussaïd, Lepagnot & Siarry, 2013). Single meta-heuristic solution operates with a single solution and keeps trying to enhance it, for example, the two popular algorithms: Simulated Annealing (Kirkpatrick, Gelatt & Vecchi, 1983), and Tabu Search (Glover, 1986). On the other hand, population meta-heuristic solution operates with a set of solutions and evaluates them (using objective function) to select the best one, such as Genetic Algorithm (Holland, 1992; Beasley, Bull, &



Martin, 1993), Evolutionary Programming (Fogel, 1994), Differential Evolution (Aliguliyev, 2009a, 2009b), and nature-inspired algorithms (Bonabeau, Dorigo, & Theraulaz, 1999; Boussaïd, Lepagnot, & Siarry, 2013; Das, Abraham & Konar, 2009).

In meta-heuristic algorithms, there are two important components: exploration (diversification) and exploitation (intensification). The balance between these components is the key for success of any optimization algorithms to solve any problems. The exploration process explores the search space globally and generates diverse solutions. Meanwhile, exploitation focuses the search on local region and exploits information in a current good solution found in this region (Boussaïd, Lepagnot, & Siarry, 2013; Yang & He, 2013). Single-based meta-heuristic solution contributes in exploitation and population-based meta-heuristic is more exploration (Boussaïd, Lepagnot & Siarry, 2013).

Meta-heuristic or modern heuristic algorithms are used to discover the optimal solution by exploring the search space, and at the same time, avoiding local optimality (Aljanabi, 2010; Rothlauf, 2011). Meta-heuristics are generally applied to hard optimization problems. Hard problems can be continuous or discrete, constrained or unconstrained, static or dynamic, and mono or multi objective functions (Boussaïd, Lepagnot & Siarry, 2013). In clustering, the aim is to achieve high similarity among objects in a cluster and less similarity between clusters. Such a situation can be considered as an optimization problem (Banati & Bajaj, 2013). This research focuses on the population-based meta-heuristic, in particular, investigating the swarm intelligence algorithm in the area of text mining.

Nature-inspired algorithm (also named Swarm Intelligence) studies the behaviors of social insects or animals in nature to be mimicked by converting them into heuristic rules to find solutions for problems faced by humans (Rothlauf, 2011). Examples of Swarm Intelligence algorithms include Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995), Ant Colony Optimization (ACO) (Dorigo, 1992), and Firefly Algorithm (Yang, 2010b). These algorithms have proved to be a success in complex optimization problems. In this section, researches that integrate clustering techniques with swarm are reviewed (Cui, Potok, & Palathing, 2005; Feng et al., 2010; Forsati, Mahdavi, Shamsfard, & Meybodi, 2013; Hassanzadeh & Meybodi, 2012; Rana, Jasola, & Kumar, 2010; Tang, Fong, Yang, & Deb, 2012; Youssef, 2011).

#### **2.2.1.2.1 Particle Swarm Optimization**

Particle Swarm Optimization Algorithm (PSO) is a computational method and a type of population-based algorithms. The idea of PSO comes from social behaviors of swarm groups to reach a goal. Swarm groups include schools of fish, flocks of bird (Kennedy & Eberhart, 1995). PSO offers advantages such as simple structure, convergence rate and strong optimization. This leads researchers to apply it in many applications such as clustering (Cui, Potok, & Palathing, 2005; Feng et al., 2010; Lu, Wang, Li, & Zhou, 2009; Rana et al., 2010; Youssef, 2011).

Particle Swarm Optimization (PSO) has been presented in partitional document clustering (Cui, Potok, & Palathing, 2005; Lu, Wang, Li, & Zhou, 2009). Cui, Potok, and Palathing (2005) worked on the hybridization of PSO with K-means and applied

it on document datasets. The result demonstrated that the proposed hybrid PSO outperformed K-means and PSO. The step-by-step process of PSO clustering is as shown in Figure 2.8.

**Particle Swarm Optimization algorithm**

**Step 1:** Each particle, randomly choose  $k$  cluster centers.

**Step 2:** For each particle.

**Step 3:** Assign each document to the closest center.

**Step 4:** Compute the fitness value based on average distance between documents and center (ADDC).

**Step 5:** Update the velocity and position of particle.

**Step 6:** Repeat Step 2 until one of the stop conditions is reached; the maximum number is reached or the average change in the center is less than the threshold (predefined value).

*Figure 2.8. The step-by-step process of PSO clustering*

Resource. Cui, Potok, and Palathing (2005)

Lu, Wang, Li, and Zhou (2009) proposed an objective function for PSO that maximized the document similarity in a cluster. This objective function was based on the extended Jaccard coefficient. The result indicated that the proposed approach outperformed K-means, agglomerative, Bisect K-means and Graph based. However, the number of  $k$  cluster was pre-assigned in the two approaches (Cui, Potok, & Palathing, 2005; Lu, Wang, Li, & Zhou, 2009).

Particle Swarm Optimization (PSO) was also utilized in partitional clustering for numerical datasets (Rana, Jasola, & Kumar, 2010; Toreini & Mehrnejad, 2011; Youssef, 2011). Rana, Jasola, and Kumar (2010) combined PSO with K-means to solve the problem of trapping in local optima in K-means and solve the problem of

slow convergence of PSO. They used the Artificial I, Artificial II classification datasets which were used by Van der Merwe and Engelbrecht (2003), and they used two real datasets: Wine and Iris. The result of the proposed approach was not better than PSO, but was positive for Wine and Iris datasets. However, the weakness was on the number of clusters which was predetermined. This problem was solved by Youssef (2011) by combining PSO with evolutionary based (EFPSC). In synthetic datasets, EFPSC has lower fitness indicator 7.45, larger Dunn Index (DI) value 1.46, and lower Davies Boulden Index (DBI) values 0.41 than K-means, modify ant and PSDC techniques. The lower fitness indicator, larger DI and lower DBI values mean better performance. In real data sets, EFPSC performs better than other methods, but in time execution, it is worse than others. Toreini and Mehrnejad (2011) utilized the work of Van der Merwe and Engelbrecht (2003) and proposed to improve PSO with FCM fitness function. The result indicated that PSO with FCM function improved the quality performance and time.

A novel hierarchical divisive clustering combined with discrete Particle Swarm Optimization (FPDC) was proposed by Feng et al. (2010). It solved the problem of the difficulty of getting a better trade-off between clustering performance and computational execution time. The experiment was implemented in documents clustering, numerical clustering and image clustering. It used eight document datasets from different resources and also used six numerical datasets from UCI machine learning repository. In document datasets, the result of average entropy for the proposed algorithm FPDC was the lowest value in six datasets out of eight. For example, in the RE0 dataset, the average entropy was 0.379, less than in Bisect K-

means (BKMS) which was 0.388, Hybrid Genetic K-means (HGKA) was 0.387 and Canonical Particle Swarm Optimization (PSOC) was 0.450. In numerical datasets, the result of average adjusted rand index for the proposed algorithm FPDC was the highest in four datasets out of six. For example, in the Zoo dataset, the average adjusted rand index was 0.710 in FPDC, 0.632 in BKMS, 0.611 in HGKA, and 0.515 in PSOC.

It can be concluded from the previous researches (Cui, Potok, & Palathing, 2005; Feng et al., 2010; Lu, Wang, Li, & Zhou, 2009; Rana, Jasola, & Kumar, 2010) that they have drawbacks in the number of clusters which was determined in the input of the algorithm. Youssef's (2011) study has a weakness in the execution time. Tables 2.5 and 2.6 clarify the summary of existing researches in the hybridization of clustering techniques and Particle Swarm Optimization in text clustering and numerical clustering.

Table 2.5

*Summary of existing researches in Particle Swarm Optimization in text clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Cui, Potok, & Palathing (2005)	Hybrid PSO with K-means	The local optima in K-means	TREC collections	Predefined k cluster
Lu, Wang, Li, & Zhou (2009)	Objective function for PSO which was extended from PSOVW	The problem of text clustering	20 Newsgroup and text datasets from CLUTO	Predefined k cluster
Feng et al., (2010)	An improved discrete particle swarm optimizer	The difficulty of getting a better trade-off between	Eight document datasets and six numerical	Predefined k cluster

Table 2.5 continued

	for divisive clustering	clustering performance and computational execution time	datasets from UCI
--	-------------------------	---	-------------------

Table 2.6

*Summary of existing researches in Particle Swarm Optimization in numerical clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Rana, Jasola, & Kumar (2010)	A hybrid K-Means with Particle Swarm Optimization algorithm	Trapping in local optima of K-means and slow convergence of Particle Swarm Optimization	Artificial I, Artificial II, Wine and Iris.	Predefined k cluster
Youssef (2011)	A new hybrid evolutionary-based data clustering using Fuzzy Particle Swarm Optimization	The number of k clusters in large datasets	Used synthetic datasets and Iris, Wine, Breast cancer and Glass	High execution time
Toreini & Mehrnejad (2011)	Improved PSO with FCM fitness function	The problem of fitness functions of PSO in Vender Merwe and Engelbrecht (2003) approach	Iris, Glass and Wine	Predefined k cluster

#### 2.2.1.2.2 Ant Colony Optimization

Ant Colony Optimization (ACO) is a type of swarm intelligence algorithm that imitates the behavior of ants in searching for the optimal solution and the shortest

path in space which is proposed by Marco Dorigo (1992). The advantages of ACO attracted researchers to implement it in many optimization problems, such as classification (Martens et al., 2007), and clustering (He, Hui, & Sim, 2006; Wang, Shen, & Tang, 2009; Zhang, Cao, & Lee, 2013; Zhang & Cao, 2011).

The Ant Colony Optimization solved the problem of combinatorial optimization clustering in document clustering (He, Hui, & Sim, 2006) and solved the problem of higher number of clusters and slow convergence (Wang, Shen, & Tang, 2009). He, Hui, and Sim (2006) tested the 20Newsgroups dataset and found that the F-measure was higher than ant-based method and K-means. Wang, Shen, and Tang (2009) integrated Ant Colony Optimization with agglomerative clustering algorithm. They tested it on the Wine dataset and real documents gathered from the Internet. The result of F-measure is not better than the ACO algorithm, but the CPU time is better. However, the quality performance needs more improvement.

The clustering of numerical dataset using Ant-based clustering was proposed by Zhang and Cao (2011) and Zhang, Cao, and Lee (2013). The proposed method changed the random object projection in the initial running by two ways. Firstly, it was integrated with kernel method (ACK) which was also integrated in the feature space to calculate the distance for similarity measure between objects. The results of the proposed algorithm (ACK) took more time than K-means, kernel based K-means, LF algorithm, ATTA, ant clustering with PCA, ACP-F and ACK-I, but in clustering quality, it outperformed the other algorithms (Zhang & Cao, 2011). Secondly, it was integrated with Kernel Entropy Component Analysis (KECA) and used Renyi Entropy to determine the object movement after a new object is added to a dataset.

The proposed NAC-RE was efficient only in the DI metric (Zhang, Cao, & Lee, 2013). However, the weakness is in the quality performance.

To sum up the foregoing researches (Zhang, Cao, & Lee, 2013; Zhang & Cao, 2011), it can be concluded that the time cost is very high in both researches; Zhang and Cao (2011) focused only on the basic process of ACK, while Zhang, Cao, and Lee (2013) had a weakness in the quality performance of their algorithm. Both researches focused only on the numerical datasets. Wang, Shen, and Tang (2009) had a weakness in quality performance which needs more improvement. Tables 2.7 and 2.8 summarize the existing researches in the hybridization of clustering techniques and ACO in text clustering and numerical clustering.

Table 2.7

*Summary of existing researches in Ant Colony Optimization in text clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
He, Hui, & Sim (2006)	A novel Ant Colony Optimization for document clustering	The problem of combinatorial optimization clustering	4 subsets (each set includes 300 documents) from 20Newsgroup	Experiment conducted on small datasets
Wang, Shen, & Tang (2009)	Combined Ant Colony Optimization with Agglomerative clustering	The too high number of cluster and slow convergence	Wine dataset and real documents gathered from the Internet	The weakness in quality performance which needs more improvement



Table 2.8

*Summary of existing researches in Ant Colony Optimization in numerical clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Zhang & Cao (2011)	The Kernel method combined with ant-based clustering ACK for data clustering	The improvement of the algorithm's efficiency and the algorithm's quality	Five synthetic datasets (square, ring, line, moon and 2D3C). Five real datasets (Wine, Iris, Zoo, Wisconsin and Yeast)	The time cost was very high and the research focused only on the basic process of ACK and only on numerical datasets
Zhang, Cao, & Lee (2013)	Ant clustering algorithm using Kernel Entropy Component Analysis and Renyi Entropy	The proposed algorithm solved three problems: the algorithm's efficiency, the algorithm's adaptability with special datasets structure and the parameters' simplification	Four synthetic datasets (square, ring, line, and moon), and four real datasets (Wine, Iris, Zoo, Wisconsin)	The weakness of the proposed algorithm was in quality performance. Efficiency in DI when Kernel Entropy Component Analysis (KECA) is used

### 2.2.1.2.3 Firefly Algorithm

Fireflies are winged beetles which produce short and rhythmic flashes. The flashing light is generated by the bioluminescence process. Firefly uses bioluminescence to attract mates or prey. The firefly flashing characteristics have three idealized rules (Yang & He, 2013; Yang, 2010a, 2010b):

1. All fireflies are unisex as well one firefly attracts to other.
2. Attractiveness is directly proportional to their brightness, so that the flashing for any two fireflies, the brighter one will be attractive than the less bright ones.
3. The firefly brightness is determined by the search space of objective function.

The Firefly Algorithm (FA) was developed by Xin-She Yang in 2007 at Cambridge University. FA has two important issues: light intensity and attractiveness. For maximum optimization problems, the light intensity  $I$  of a firefly at a particular location  $x$ , termed as  $I(x)$ , can be determined by objective function  $f(x)$ . The attractiveness  $\beta$  is relative. It changes depending on the distance between two fireflies. The pseudo code of the Firefly Algorithm is shown in Figure 2.9 (Yang & He, 2013; Yang, 2010a, 2010b).

In Step 8 in Figure 2.9, the movement of less brighter firefly towards brighter one is calculated based on Equation (2.7) (Yang, 2010b).

$$X_i(t + 1) = X_i(t) + \beta * (X_j(t) - X_i(t)) + \alpha \varepsilon_i \quad (2.7)$$

where,  $X_i(t+1)$  is a new position of firefly,  $\alpha$  is a randomization parameter between (0, 1) (Yang, 2010b),  $\varepsilon_i$  is a vector of random numbers drawn from a Gaussian distribution or uniform distribution (Yang & He, 2013; Yang, 2010a). In Step 9 in Figure 2.9, the attractiveness between two fireflies is calculated using Equation (2.8) (Yang, 2010b).

### **Firefly Algorithm**

**Step 1:** Objective function  $f(x)$ ,  $x = (x_1, \dots, x_d)^T$

**Step 2:** Generate initial population of firefly randomly  $x_i$  where  $i=1, 2, \dots, n$ ,  $n$ =number of fireflies.

**Step 3:** Initial Light Intensity  $I$  at  $x_i$  is determined by  $f(x)$ .

**Step 4:** Define light absorption coefficient  $\gamma$ .

**Step 5:** While ( $t < \text{Max Generation}$ )

**Step 6:** For  $i=1$  to  $N$  all  $n$  fireflies

**Step 7:** For  $j=1$  to  $N$  all  $n$  fireflies (inner loop)

**Step 8:** If ( $I_i < I_j$ ) { Move firefly  $i$  towards  $j$ ; end if }

**Step 9:** Vary attractiveness with distance  $r$  via  $\exp[-\gamma r]$

**Step 10:** Evaluate new solutions and update light intensity.

**Step 11:** End for  $j$

**Step 12:** End for  $i$

**Step 13:** Rank the firefly and find the current global best  $g^*$ .

**Step 14:** End while

**Step 15:** Post process results and visualization.

Figure 2.9. Pseudo code of Firefly Algorithm

Resource. Yang (2010b, p.82)

$$\beta = \beta_0 \exp(-\gamma r_{ij}^2) \quad (2.8)$$

where,  $\beta_0$  is the initial attractiveness,  $\gamma$  is the light absorption coefficient and in most application sets to 1,  $r_{ij}$  is the distance between two documents  $i$  and  $j$  that is calculated using Cartesian distance (Yang, 2010b) as shown in Equation 2.9.

$$\text{Cartesian\_distance}(X_j, X_i) = \sqrt{(X_j - X_i)^2} \quad (2.9)$$

where,  $X_i$ ,  $X_j$  is the position of two fireflies. In Step 10 in Figure 2.9, the new solution is evaluated using objective function and updates the value of light intensity that is related with objective function.

Firefly Algorithm has two main advantages over other algorithms. The first advantage is automatic subdivision which is related with the fact that the whole population can automatically subdivide into subgroups. The second advantage is the ability to deal with multimodality and this is related with the fact that each group can swarm around each mode or local optimum. From all these modes, the best global solution can exist (Fister, Jr, Yang, & Brest, 2013; Yang & He, 2013).

Firefly Algorithm has been implemented in many optimization problems in different topics, such as speech recognition (Hassanzadeh, Faez, & Seyfi, 2012), image segmentation (Hassanzadeh, Vojodi, & Moghadam, 2011; Horng & Jiang, 2010), reliability-redundancy allocation problems (dos Santos Coelho, de Andrade Bernert, & Mariani, 2011), discrete optimization problems (Sayadi, Hafezalkotob, & Naini, 2013), semantic web service composition (Pop et al., 2011), data classification (Nandy, Sarkar, & Das, 2012), anomaly detection (Adaniya Abr̃ao & Proenc,a Jr., 2013), parallel and distributed systems (Falcon, Almeida, & Nayak, 2011), mobile network (Bojic, Podobnik, Ljubi, Jezic, & Kusek, 2012), and economic dispatch problems (Yang, Hosseini, & Gandomi, 2012). In all of the previous fields, Firefly Algorithm has successfully solved the problems and identified the optimal solution.

The Firefly algorithm has also been studied in numerical clustering (Abshouri & Bakhtiary, 2012; Hassanzadeh & Meybodi, 2012; Senthilnath, Omkar, & Mani,

2011; Tang, Fong, Yang, & Deb, 2012). The algorithm has been represented to improve the performance of supervised clustering (Senthilnath, Omkar, & Mani, 2011). The goal of the algorithm is to find the center of clusters that minimizes the sum of distance from the center to each object in the same cluster. In the evaluation phase, 13 benchmark datasets were used, which are Balance, Cancer, Cancer-int, Credit, Dermatology, Diabetes, E.Coli, Glass, Heart, Horse, Iris, Thyroid and Wine. The result was compared with Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), and nine other methods. It is noted that FA was efficient, robust, and reliable for generating optimal cluster centers. Banati and Bajaj (2013) presented a new approach of Firefly Algorithm for unsupervised clustering. They used trace within criteria and variances ratio criteria to evaluate the solutions. The results indicated that the proposed approach performed better than PSO and DE.

Furthermore, Firefly Algorithm was also integrated into two different clustering techniques, one with K-means (KFA) and one with K-harmonic (Abshouri & Bakhtiary, 2012; Hassanzadeh & Meybodi, 2012). Both methods solved the problem of local optima for K-means and K-harmonic and the identification of the center of clusters. The proposed hybrid, KFA, has the ability to minimize the intra-cluster distance in five datasets and has been compared against K-means, PSO and KPSO (Hassanzadeh & Meybodi, 2012). However, the validation of the proposed algorithm is only based on intra-cluster which determines the similarity between objects and does not use inter-cluster measurement. The result of the proposed hybrid K-harmonic (Abshouri & Bakhtiary, 2012) on four datasets indicated that F-measure is the highest value and KHM is the least value in all datasets compared to KHM,

PSOKHM, and Genetic PSO-KHM. The weakness of the proposed methods is random initialization of the centers of clusters.

Due to the random initial centroids, K-means is always caught in the trapping in local optima, this problem has attracted researchers to hybrid K-means with five nature-inspired optimization algorithms; Firefly, Cuckoo, Bat, Ant, and Wolf (Tang, Fong, Yang, & Deb, 2012), termed as C-firefly, C-cuckoo, C-bat, C-Ant, and C-wolf. The pseudo code of the hybrid Firefly with K-means is shown in Figure 2.10.

#### **Hybrid Firefly with K-means Algorithm**

- Step 1:** Determine the number of k clusters; initialize the population of fireflies N, and related parameters.
- Step 2:** Randomly assign k clusters for each N firefly.
- Step 3:** For each firefly, Select k objects from S data objects as initial centroids, by taking the mean values of the attributes of the objects within their given clusters.
- Step 4:** Calculate the fitness of the centroid in each firefly, and find the best solution that is represented by the total fitness values of centroid in a firefly.
- Step 5:** For each firefly, update its light intensity according to its fitness value (objective function).
- Step 6:** For each firefly, update its attractiveness that varies with distance.
- Step 7:** Merge the fireflies by allowing the less bright one to be attracted by the brighter one.
- Step 8:** Are there no brighter fireflies than the given firefly, if yes continue, else go to Step11.
- Step 9:** The firefly will move randomly.
- Step 10:** Update centroids in each firefly according to their latest positions.
- Step 11:** Rank the fireflies and find the current best.
- Step 12:** Reassign the clusters according to the best solution.
- Step 13:** Output the best cluster configuration that is represented by the firefly that has the greatest fitness.
- Step14:** Are the exit criteria met yet? If yes end, or else return to Step 5.

*Figure 2.10.* Pseudo code of integrated Firefly with K-means clustering algorithm

Resource. Tang, Fong, Yang, and Deb (2012)

In the experiment, five datasets from the UCI repository were employed. For the evaluation, the squared error function was used to measure the objective function

and measure the execution time for each algorithm. The experiment results compared five proposed algorithms with the benchmark K-means algorithm and indicated that C-cuckoo, C-bat, and C-wolf performed the best objective values compared with C-firefly and C-ant. Furthermore, C-bat requires less time for execution, hence, C-bat is indicated as the fastest algorithm for finding the optimal solution. However, the drawback with these proposed algorithms was the number of  $k$  clusters which was determined in the initial part of the clustering. The pseudo code of the hybrid Bat algorithm with K-means is shown in Figure 2.11.

#### **Hybrid Bat with K-means Algorithm**

- Step 1:** Determine the number of  $k$  clusters, initialize the population of bats  $N$ , for each bat define the frequency factor  $Q$  and loudness  $A$ .
- Step 2:** Randomly assign  $k$  clusters for each  $N$  bat.
- Step 3:** For each bat, select  $k$  objects from  $S$  data objects as initial centroids, by taking the mean values of the attributes of the objects within their given clusters.
- Step 4:** Calculate the fitness of the centroid in each bat, and find the best solution that is represented by the total fitness values of centroid in a bat.
- Step 5:** Generate a new solution by adjusting the frequency, updating the velocity and creating new centroid values.
- Step 6:** If  $\text{random}(0, 1) > \text{pulse rate } R$
- Step 7:** For each bat, select a solution among a set of best solutions from the other bats, and generate a new local solution around the selected best solution.
- Step 8:** If  $\text{random}(0, 1) < A$  and  $f(x_i) < f(x^*)$
- Step 9:** Accept the new solution, increase  $r$  and reduce  $A$ .
- Step 10:** Reassign the clusters.
- Step 11:** Output the best cluster configuration represented by the bat that has the greatest fitness.
- Step 12:** End if.
- Step 13:** End if.
- Step 14:** Are the exit criteria met yet, if yes end, else return to Step 4.

*Figure 2.11.* Pseudo code of integrated Bat with K-means clustering algorithm

Resource. Tang, Fong, Yang, and Deb (2012)

Rui, Fong, Yang, and Deb (2012) applied the previous work of Tang, Fong, Yang, and Deb (2012) of hybrid K-means with four nature-inspired optimization

algorithms: Firefly, Cuckoo, Bat and Wolf on web intelligence data. The results demonstrated that this hybridization outperformed hybrid PSO with K-means (C-PSO) and K-means. However, C-Firefly requires high CPU time to execute. Further experiment has been reported by Fong, Deb, Yang, and Zhuang (2014) by testing the combined four optimization algorithms named ACO, Bat, Cuckoo, and Firefly with K-means on real-life datasets and image segmentation.

Demir and Karci (2015) used golden ratio method to set the parameters of Firefly algorithm. They tested their proposed clustering method on breast cancer data. The result of Firefly algorithm with golden ratio shows better success rate compared to the standard Firefly algorithm.

For detecting the brain tumor, Alsmadi (2014) proposed a dynamic clustering method that is based on hybrid model of Firefly algorithm and Fuzzy c-means (FCM). The use of the Firefly algorithm is to overcome the drawbacks of the FCM which has slow convergence rate, trapped in local optima, and is sensitive to initial centers. The hybridization method was able to detect the number of clusters and the location of the points. It shows better results than state-of-the-art methods such as the hybridization between harmony search and FCM, and combination between Genetic algorithm and point symmetry-based index.

It can be concluded that in the validation part of Hassanzadeh and Meybodi (2012), the weakness is in measuring the inter-cluster (Abshouri & Bakhtiary, 2012; Hassanzadeh & Meybodi, 2012; Senthilnath, Omkar & Mani, 2011; Tang, Fong, Yang & Deb, 2012). Moreover, Tang, Fong, Yang, and Deb (2012) tested only the



proposed algorithm, but did not measure the validation of the clusters' output from each algorithm. Abshouri and Bakhtiary (2012) had a weakness in the proposed method, which is the random center of clusters. Senthilnath, Omkar, and Mani (2011) studied the firefly algorithm for supervised clustering for numerical data set. Table 2.9 and Table 2.10 present the summary of existing researches in Firefly Algorithm in web intelligent data and numerical clustering.

Table 2.9

*Summary of existing researches in Firefly Algorithm in web intelligent data.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Rui, Fong, Yang, & Deb (2012)	Adopting work of Tang, Fong, Yang, and Deb (2012) and testing on web intelligent data	Testing nature optimization algorithms in web data clustering	Page Blocks, Ipad Auctions on eBay, Internet Usage and Spam base	C-Firefly requires high CPU time to execute

Table 2.10

*Summary of existing researches in Firefly Algorithm in numerical clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Senthilnath, Omkar, & Mani (2011)	Studied the Firefly Algorithm performance in clustering	Studied the performance of the Firefly Algorithm in data clustering	Balance, Cancer, Cancer-int, Credit, Dermatology, Diabetes, E.Coli, Glass, Heart, Horse, Iris, Thyroid and Wine	Implemented only on numerical data set
Hassanzadeh & Meybodi (2012)	Hybrid Firefly Algorithm and K-means for data clustering KFA	Initial centroid for K-means and local optimal convergence	UCI (Iris, Wdbc, Sonar, Glass and Wine)	Used only intra-cluster which determines the similarity between

Table 2.10 continued

				objects in one cluster, but did not used inter-cluster measurement which determines the differences between clusters Testing only the proposed algorithm, but did not measure the validation of the clusters' output from each algorithm
Tang, Fong, Yang, & Deb (2012)	Hybrid Firefly, Cuckoo, Bat, Ant and Wolf with K-means	K-means trapped in local optima due the random initial centroids	Five datasets from UCI	
Abshouri & Bakhtiary (2012)	Hybrid Firefly with K-harmonic means algorithm	Local optima	Iris, Glass, Wine and contraceptive	Predefined k cluster
Banati & Bajaj (2012)	Firefly algorithm for data clustering	Clustering problem	Iris, Wine, Cancer and Thyroid	Predefined k cluster
Fong, Deb, Yang, & Zhuang (2014)	Extensive experiments on Hybrid Firefly, Cuckoo, Bat, and Ant with K-means	Testing nature optimization algorithms	real life datasets and image segmentation	Predefined k cluster
Demir & karci (2015)	Golden ratio method to set parameters of firefly algorithm.	Heuristic algorithm find only near optimal solutions.	Breast cancer data	Not compare the result with others in same data

#### 2.2.1.2.4 Hybrid of Clustering Techniques and other Search Optimization

Hybridization integrates different data mining techniques or meta-heuristic algorithms in one approach to construct a clustering model (Stahlbock, Crone, &

Lessmann, 2010). Many researchers suggest to hybrid partitional clustering with an optimization algorithm to enhance the quality performance of partitional clustering (Forsati, Mahdavi, Shamsfard, & Meybodi, 2013; Hatamlou, Abdullah, & Nezamabadi-pour, 2012; Luo, Li, & Chung, 2009; Mahmuddin, 2008; Zhong, Liu, & Li, 2010).

A new approach for text document clustering was proposed by Luo, Li, and Chung (2009) and Forsati, Mahdavi, Shamsfardand, and Meybodi (2013). The problem of similarity measure between documents was solved based on mixed cosine and link functions (Luo, Li, & Chung, 2009). This approach determines the initial cluster centers based on document ranks. The split between clusters is based on heuristic function which depends on neighbors. The researchers used thirteen datasets from different resources and used two quality metrics: F-measure and Purity. The proposed method acquired the best result of F-measure 0.75 on MED1 and Top1 datasets. Furthermore, the purity value was 0.8 in the CACM1 dataset and 0.818 in the Top1 dataset.

In addition, the problem of randomly selecting initial centers was solved by the hybridization of k-means with Harmony search (HS) in three different versions, based on the stage of k-means during execution (Forsati, Mahdavi, Shamsfard, & Meybodi, 2013). The versions are sequential hybrid, interleaved hybrid and one step of HS hybrid. The quality results of one step of HS are better than K-means, HSCLUST, Sequential hybrid, interleaved hybrid and GA. However, this version is not suitable for large datasets due to the increasing execution time when the number of documents increases (linear relation).

A hybrid K-means with optimization algorithm to solve the problem of trapping in local optima in numerical datasets was presented in Hatamlou, Abdullah, and Nezamabadi-pour (2012), and Mahmuddin (2008). A hybrid Bee's algorithm with K-means algorithm to find the optimum cluster centers for un-labelled data was proposed by Mahmuddin (2008). A hybrid gravitational search algorithm (Rashedi, Nezamabadi-pour, & Saryazdi, 2009) with K-means GSA-KM for data clustering was proposed (Hatamlou, Abdullah, & Nezamabadi-pour, 2012). The hybridization is useful to speed up the convergence of GSA. The GSA-KM is learned to be the best in intra-cluster distance and fitness function compared to K-means, GA, SA, ACO, HBMO, PSO, and GSA. However, the initial cluster number is determined in GSA-KM.

Zhong, Liu, and Li (2010) solved the weakness of the FCM algorithm for outliers' sensitization and determination of exact cluster number. They presented Fuzzy C-means (FCM) algorithm for clustering that depends on gravity, which identifies the initial centers and isolates outliers by calculating the force between two point masses. One point mass is represented as the number of objects around point mass, so the bigger the mass is represented, the larger the density is. In addition, a novel FCM merged the produced clusters to obtain an appropriate cluster number. The result of the proposed FCM improves the standard FCM, where the average recall improvement was 1.34% in the Iris dataset and 1.72% in the breast cancer Wisconsin dataset. Furthermore, the average precision was 1.03% in the Iris dataset and 0.12% in the breast cancer Wisconsin dataset. However, the number of k cluster is identified and the algorithm is also tested on small datasets. Tables 2.11 and 2.12

contain the summary of existing researches in the hybridization of clustering techniques and other search optimization in text clustering and numerical clustering.

Table 2.11

*Summary of existing researches in the hybridization of clustering techniques and other search optimization in text clustering.*

Authors	Contribution	Problem Solved	Dataset	Weakness
Luo, Li, & Chung (2009)	A new approach for text documents clustering. The determination of initial cluster centers depending on document ranks, the similarity between documents based on mixed cosine and link functions and lastly the splitting of the clusters based on heuristic function which depends on neighbors	Similarity measure between documents when using cosine function	Thirteen datasets from different resources such as Reuters - 21578, MEDLINE, CA CM, CISI and TREC	Predefined k cluster
Forsati, Mahdavi, Shamsfard, & Meybodi (2013)	A new approach of Harmony search optimization method for document clustering, and hybrid Harmony Search optimization with K-means	K-means randomly selected initial cluster centers	Politics, Trec, Dmoz, 20 Newsgroup and WebAce	Computational cost is high

Table 2.12

*Summary of existing researches in the hybridization of clustering techniques and other search optimization in numerical clustering.*

Authors	Contribution	Problem-Solved	Dataset	Weakness
Zhong, Liu, & Li (2010)	A novel Fuzzy C-means FCM depending on gravity	The weakness for FCM algorithm for outliers' sensitization	Iris, and Breast cancer Wisconsin	Predefined k cluster and the

Table 2.12 continued

		algorithm
	and determination of	is tested on
	exact cluster number	small
		numerical
		datasets
Hatamlou , Abdullah, & Nezamaba di-pour (2012)	Hybrid gravitational search algorithms with K-means	K-means is trapped in local optima and convergence of GSA Wine, Iris, CMC, Cancer and Glass Predefined k cluster

## 2.2.2 Dynamic Approach

In contrast to the static approach of clustering, dynamic clustering has an ability to automatically identify the number of clusters. Hence, the literature covering this matter is categorized into two approaches: estimation and population-based.

### 2.2.2.1 Estimation Approach

Estimation clustering is initialized by identifying a range of k value (minimum and maximum) and using one of the validity indices. Clustering is undertaken for identifying the number of clusters and clusters with the best quality (best value of validity indices). Bayesian Information Criterion (BIC) has been proposed as a solution for K-means problem (the number of cluster), where it can estimate the best k (Pelleg & Moore, 2000). BIC is used to measure the improvement of the cluster structure between a cluster and its two children clusters. The result shows that the use of BIC can reveal the actual number of classes and it outperforms K-means. The number of clusters in a dataset has been estimated by integrating a modified K-

means and Bees algorithm. The purpose of using Bees algorithm is to identify the possible optimal or near optimal centroids, whilst, the aim of utilizing K-means is to detect the best cluster (Mahmuddin, 2008).

In the work of Sayed, Hacid, and Zighed (2009), the agglomerative hierarchical clustering with validity index (VI) has been presented. VI for two closest clusters is measured before and after the merging step, where the merging of these two clusters occurs if the VI improves after merging. The process of merging continues until the optimal clustering solution is reached.

Besides that, the combination between K-means and Particle Swarm Optimization has been proposed for dynamic data clustering, named as KCPSO (Kao & Lee, 2009). KCPSO has the capability to identify the optimal number of clusters without any information given before performing the clustering process. The aim of using PSO is to optimize the number of clusters, while the aim of using K-means is to identify the best clustering result. The results indicated that KCPSO can obtain the best or equal clustering results with less time compared with two dynamic clustering algorithms: Dynamic Clustering using Particle Swarm Optimization (DCPSO) and Genetic Clustering for Unknown K (GCUK).

Another automatic clustering method has been proposed by Kuo and Zulvia (2013), termed as Automatic Clustering using Particle Swarm Optimization (ACPSO). In this method, Particle Swarm Optimization is used to identify the number of clusters and K-means to adjust the clustering centers. ACPSO initializes by determining a range of cluster numbers  $[2, N_{max}]$ . The experimental results indicated that ACPSO

outperforms Dynamic Clustering Genetic algorithm (DCGA), Dynamic Clustering using Particle Swarm Optimization (DCPSO) and Dynamic Clustering using Particle Swarm Optimization and Genetic Algorithm (DCPG) in producing better accuracy and less CPU time.

It is learned from previous studies (Kao & Lee, 2009; Kuo & Zulvia, 2013; Mahmuddin, 2008; Pelleg & Moore, 2000; Sayed, Hacid, & Zighed, 2009) that the estimation approach is appropriate to find solutions for problems (i.e. determining the number of clusters that requires little or no knowledge of datasets). However, there exists a difficulty to identify the range of clusters (lower and upper values of cluster numbers).

#### **2.2.2.2 Population-based Approach**

This approach can be classified into two sub-approaches: evolutionary clustering methods and swarm-based methods. Evolutionary clustering methods are such as Evolution Strategies (Lee & Antonsson, 2000), Evolutionary Programming (Sarkar, Yegnanarayana, & Khemani, 1997), Differential Evolution Algorithm (Das, Abraham, & Konar, 2008) and Genetic Algorithms (Kuo, Syu, Chen, & Tien, 2012), while, Swarm-based methods are such as Flocking-based approach (Cui, Gao, & Potok, 2006; Picarougne, Azzag, Venturini, & Guinot, 2007) and Ant-based clustering (Tan, Ting, & Teng, 2011a, 2011b).

Lee and Antonsson (2000) developed an Evolution Strategy (ES) to overcome the problem of static number of clusters in partitional clustering. ES effectively searches for both the optimal center and optimal number for clusters by performing variable



length genomes. Evolutionary programming-based clustering algorithm for discovering the number of clusters and cluster centers has been proposed by Sarkar, Yegnanarayana, and Khemani (1997). They used two object functions (minimization functions) in the algorithm; Davies Bouldin Index (DBI) for global view (i.e. to give the optimum number of clusters) and the overall sum of the squared errors between objects and cluster center for local view (i.e. to determine which object belongs to which cluster). The result demonstrated the usefulness of the proposed Evolutionary Programming to avoid local minima and generate proper numbers of clusters.

A genetic-based clustering method (Liu, Wu, & Shen, 2011) has been proposed for automatically identifying the number of clusters. In the design, a balance between selection and variety of the solution was retained by implementing two operations: noising selection and division absorption mutation. The Davies Bouldin Index was used for evaluating the fitness of individuals. The experimental results indicated that the genetic-based clustering method has the ability to construct the number of clusters and obtain the clustering solution automatically.

The integrating between Particle Swarm Optimization (PSO) and Genetic Algorithm (GA), termed as the DCPG method, has been proposed by Kuo, Syu, Chen, and Tien (2012). It operates without any knowledge of the  $k$  number of clusters (a dynamic approach). As presented in Figure 2.12, the DCPG method initially selects cluster centroids  $M$  from a set of dataset  $Z$ , and randomly determines the position and velocity. The length of particle is equal to the number of cluster centroids  $M$ , where if the bit in  $M$  is equal to 1, then the point is centroid, otherwise, it will not be selected as a centroid.

### **Integrating Particle Swarm Optimization & Genetic Algorithm**

**Step 1:** Generate the initial cluster centroid, position and velocity randomly.

**Step 2:** While not reach the pre-specified number of iterations does.

**Step 3:** Calculate the fitness value of each particle.

**Step 4:** Select personal best ( $P_{id}$ ) and a global best ( $P_{gd}$ ).

**Step 5:** Update the position and velocity of each particle.

**Step 6:** Perform Steps 6.1 and 6.2 for the updated parent.

**Step 6.1:** copy all particles to generate population one.

**Step 6.2:** perform a two point crossover for  $P_{id}$  and  $P_{gd}$  in Step 4 and mutation for  $P_{gd}$  to generate population two.

**Step 7:** Combine populations one and two and calculate the fitness value of each particle.

**Step 8:** Perform an elitist selection for populations one and two.

**Step 9:** Do while not reach pre-specified number of iterations, return to Step 3.

**Step 10:** Perform K-means algorithm.

**Step 11:** Resolved the optimal center of clusters.

**Step 12:** End While.

*Figure 2.12.* Pseudo code of integrating Particle Swarm Optimization with Genetic Algorithm (DCPG)

Resource. Kuo, Syu, Chen, and Tien (2012)

The calculation of the fitness function for each particle is performed by constructing the clusters of each particle, and calculating the sum of Euclidean distance between objects and centroids. Then, a personal best ( $P_{id}$ ) and a global best ( $P_{gd}$ ) are selected, followed by updating the position and velocity of each particle by two steps: 1) Copy all particles to generate population one and perform a two-point crossover for  $P_{id}$  and  $P_{gd}$  and mutation for  $P_{gd}$  to generate population two; 2) Combine these two populations and calculate the fitness for each particle. The next iterative population is generated by performing an elitist selection for the two populations. The previous processes continue until the number of iterations is satisfied. After that, the K-means algorithm is applied to adjust the number of

clusters. The result indicates that the DCPG algorithm can generate the right number of clusters and can realize the best clustering results compared to DCPSO, ACMP SO, and DCGA algorithms.

Swarm-based methods utilize swarm like agents to group data directly without the need to define the number of clusters. The dynamic swarm-based approach can automatically discover the appropriate number of clusters, in a given data collection, without any support. Hence, it offers a more convenient cluster analysis. Dynamic swarm-based approach adapts the mechanism of a specific insect or animal that is found in the nature and converts it to heuristic rules. Each swarm is treated like an agent that follows the heuristic rules to carry out the sorting and grouping of objects (Tan, 2012). In literature, there are examples of such approach in solving clustering problems such as flocking-based approach (Cui, Gao, & Potok, 2006; Picarougne, Azzag, Venturini, & Guinot, 2007) and ant-based clustering (Tan, Ting, & Teng, 2011a, 2011b).

The flocking-based approach is related with the behaviors of swarm intelligence (Bonabeau, Dorigo, & Theraulaz, 1999) where a group of flocks of swarm move in a 2D or 3D search space following the same rules of flocks; get close to similar agents or far away from dissimilar agents (Picarougne, Azzag, Venturini, & Guinot, 2007).

In general, the flocking model includes three rules of behavior: separation, cohesion and alignment. In separation, the agent has the ability to maintain a specific distance from other agents; while in cohesion, the agent has the ability to associate with other nearby agents. In the alignment rule, the agent aligns with closer characters. Folino,

Forestiero, and Spezzano (2009) presented the adaptive flocking algorithm that is based on the simple flocking model. They introduced a new adaptive parameter (speed) that changes based on the color of the agent, and the new position of the agent is based the position of other agents (red and white agents). This approach is computationally expensive as it requires multiple distance computations.

On the other hand, the ant-based approach deals with the behaviors of ants, where each ant can perform the sorting and corpse cleaning. This approach works by distributing the data object randomly in the 2D grid search space, then determining a specific number of ants (agents) that move randomly in this grid to pick up a data item if it does not hold any object (item) and drop the object (item) if it finds a similar object. This process continues until it reaches a specific number of iterations (Deneubourg et al., 1991).

El-Feghi, Errateeb, Ahmadi, and Sid-Ahmed (2009) proposed an adaptive ant colony clustering algorithm, termed as AACA, which improved the picking and dropping probability functions of standard ant-based clustering. The improvement is done by adding a new parameter which represents the value of pheromone at each location on the grid search space. Additionally, it improves the similarity scaling factor by automatic adoption (by reflecting the frequency of the agents' successful picking and dropping processes). The AACA performs K-means and agglomerative hierarchical clustering in terms of accuracy and obtained number of clusters.

A practical General Stochastic Clustering Method (PGSCM) that is a simplification of the ant-based clustering approach has been proposed (Tan, Ting, & Teng, 2011a).

PGSCM is used to cluster multivariate real world data. The pseudo code of PGSCM is illustrated in Figure 2.13. The input of PGSCM is a dataset,  $D$ , that contains  $n$  objects and the output is the number of clusters discovered by the PGSCM method, without any prior knowledge.

**Practical General Stochastic Clustering Method**

**Step 1:** Input the dataset  $D$  with  $n$  objects.

**Step 2:** The dissimilarity threshold is calculated for all  $n$  objects.

**Step 3:** Each object in the dataset is allocated to a bin.

**Step 4:** Do while iteration  $\leq$  Max iteration

**Step 5:** Choose two objects from dataset  $D$  randomly and they must not be equal.

**Step 6:** If the distance between two selecting objects  $<$  minimum dissimilarity threshold of two objects.

**Step 7:** Store the comparison outcome.

**Step 8:** If the level of support (first object)  $<$  level of support (second object)

**Step 9:** Move first object to second object.

**Step 10:** Else move second object to first object.

**Step 11:** End If

**Step 12:** End While

**Step 13:** Output a set of clusters that represent all non-empty bins.

*Figure 2.13.* Pseudo code of practical General Stochastic Clustering Method (PGSCM)

Resource. Tan, Ting, and Teng (2011a)

In the initialization of PGSCM, the dissimilarity threshold for  $n$  objects is estimated. Then, it creates  $n$  bins where each bin includes one object from the dataset  $D$ . Through the work of PGSCM, it selects two objects randomly from a dataset; if the distance between these two objects is less than the dissimilarity threshold of them, then the level of support of the two objects is compared. If object  $i$  has less support than  $j$ , then the lesser one is moved to the greater one and vice versa. At the end of

the iterations, a number of small and large bins are created. The large bins are selected as output clusters, while the small bins are reassigned to large bins (objects in small bins are assigned to similar centers in large bins). The selection process of large bins is based on the threshold of  $(50, n/20)$  (i.e. the threshold is 5% of the size of dataset,  $n$ ), this threshold is based on the criterion used in Picarougne, Azzag, Venturini, and Guinot (2007). The result of the PGSCM method performs well compared to the state-of-the-art methods. However, randomly selecting two objects in every iteration may create other issues. There is a chance that in some iterations, the same objects are selected or some objects are not selected at all. Furthermore, the selection process initially requires a large number of iterations to increase the probability of selecting different objects.

### **2.3 Research Gap**

Partitional clustering such as K-means suffers from local optima due to the random initial centers (Wang Liu, Chen, & Tang, 2011; Yang, 2010). It needs to define the number of  $k$  clusters as the initial value. Density-based clustering has a weakness in clustering documents which is usually of high dimensionality data (Chehreghani, Abolhassani, & Chehreghani, 2008; Zhao, Cao, Zhang, & Zhang, 2011). Grid-based clustering needs to determine the number of cells as the initial value (Han & Kamber, 2006). Model-based clustering, such as SOM and NN, is sensitive to the initial selection of weight and needs to set the parameters of the learning rate and neighborhood radius (Rokach & Maimon, 2005). All of the previous approaches produce a flat clustering which present users with single level clusters. On the other hand, hierarchical clustering organizes the documents in a hierarchical structure

(Murugesan & Zhang, 2011a, 2011b; Zhu, Fung, Mu, & Li, 2008). It is an efficient method for clustering documents in information retrieval which can create taxonomy of structure set of clusters.

Hierarchical clustering includes two approaches: divisive and agglomerative (Das, Abraham, & Konar, 2009). Agglomerative is not efficient with large datasets (Zhu, Fung, Mu, & Li, 2008). On the other hand, the divisive approach such as Bisect K-means needs to refine the cluster result and needs to determine the initial centers (Kashef & Kamel, 2009; Murugesan & Zhang, 2011a, 2011b).

The existing works of hybrid hierarchical clustering which merges partitional and agglomerative methods, such as in Murugesan and Zhang (2011a, 2011b), and Zhu, Fung, Mu, and Li (2008) still suffer from local optima in partitional steps and needs to define k clusters. Many researchers attempted to solve these problems by utilizing meta-heuristic optimization algorithms such as harmony search (Forsati, Mahdavi, Shamsfard, & Meybodi, 2013), gravitational search (Hatamlou, Abdullah, & Nezamabadi-pour, 2012) and swarm algorithms (Feng et al., 2010; Lu, Wang, Li, & Zhou, 2009; Wang, Shen, & Tang, 2009). All of the previous solutions still need to determine the number of k clusters as the initial value.

Over the years, the problem of determining the number of k clusters has been solved by two approaches: estimation approach and swarm-based approach. From the literature (Kao & Lee, 2009; Kuo & Zulvia, 2013; Mahmuddin, 2008; Pelleg & Moore, 2000; Sayed, Hacid, & Zighed, 2009), the first approach (estimation approach) is appropriate to identify a solution for problems (i.e. determining the

number of clusters that requires little or no knowledge of datasets). However, there exists a difficulty to identify the range of clusters (lower and upper values of cluster numbers). In the second approach, the swarm-based approach, such as ant-based clustering, generates too many clusters and it is possible that the objects are still carried by ants or left alone in the 2D grid when the stopping condition is reached (He, Hui, & Sim, 2006). In the flocking approach, the computational process is expensive as it requires multiple distance computations.

## **2.4 Summary**

Many resources are available on the Internet and they are of various areas, such as sports, news, science articles, etc. These resources need a technique that is able to organize them in a structural form. The most significant technique is the hierarchical text clustering.

From previous researches, it can be concluded that hierarchical text clustering is an efficient method to organize text documents. However, in divisive hierarchical text clustering, there exists a problem in relocating documents when it has been assigned to one cluster. Furthermore, the divisive hierarchical clustering utilizes partitional clustering techniques that require the determination of  $k$  number of clusters. Previous researches have been suggested to solve the problem of local optima by integrating clustering techniques with swarm intelligence algorithms such as Ant Colony Optimization, Particle Swarm Optimization, Bees Algorithm, etc. These algorithms are efficient in finding the optimal solution. A new swarm intelligence algorithm called Firefly Algorithm (FA) which was presented by Yang (2007) has been



efficient in finding the global optimal solution. There exist researches in utilizing Firefly Algorithm to improve the performance of clustering in numerical datasets, but works on text documents have yet to be reported.

Additionally, previous works have been suggested a pre-defined value for  $k$  which represents the number of clusters, but this is not appropriate for text clustering as there is no prior knowledge about the datasets.



## CHAPTER THREE

### RESEARCH METHODOLOGY

This chapter includes the methodology that was utilized to undertake the research. It is based on the experimental methodology which was widely used in computer science fields such as NP-hard problems, games, neural network, and machine learning (Doding, 2002). The utilized experimental research steps are as illustrated in Figure 3.1.

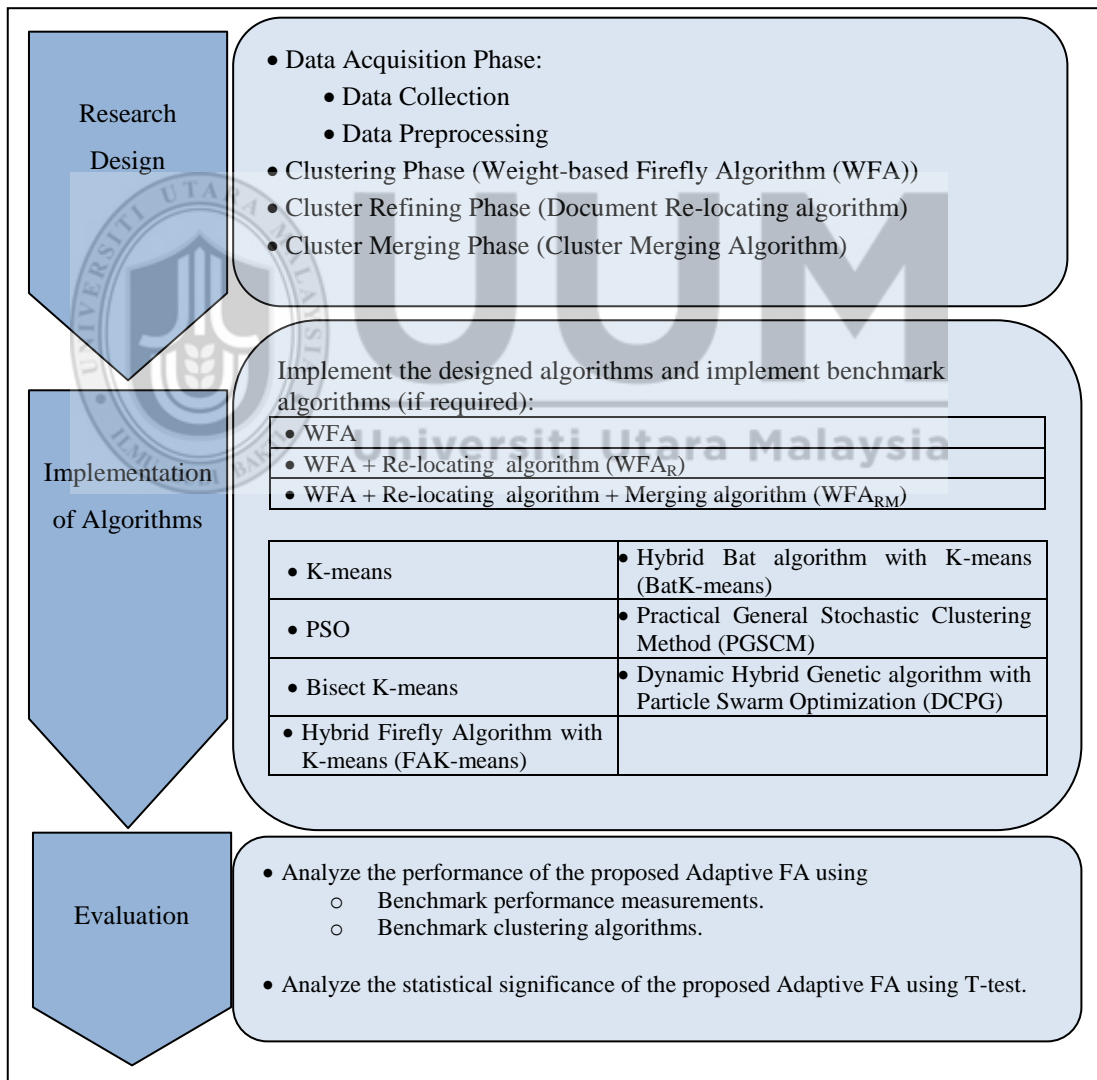
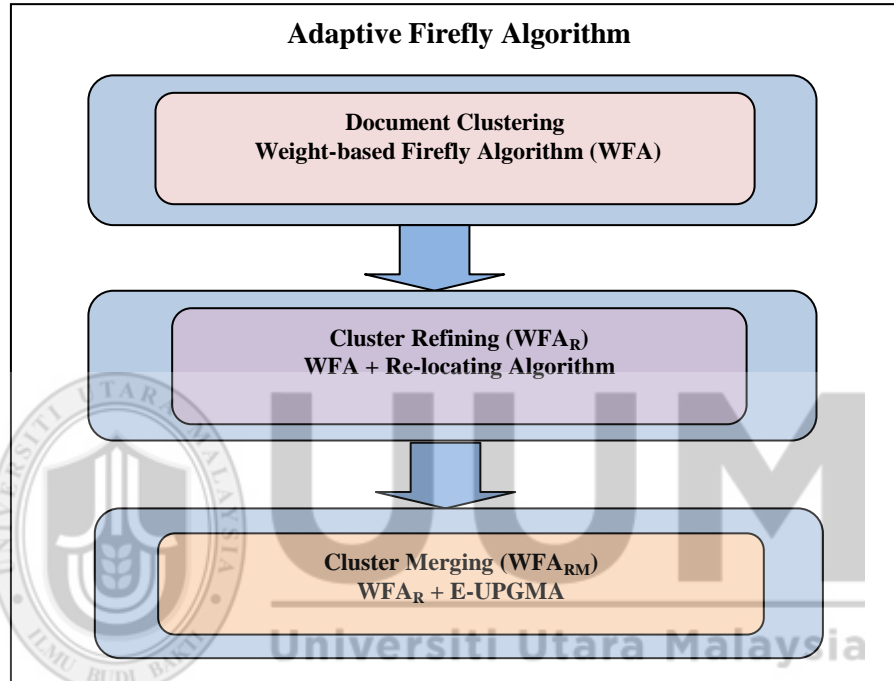


Figure 3.1. The experimental research steps

### 3.1 Research Design

This research suggests a hierarchal text clustering algorithm based on Firefly Algorithm. Figure 3.2 shows the components of the proposed Adaptive Firefly Algorithm for hierarchical text clustering.



*Figure 3.2.* The components of the proposed Adaptive Firefly algorithm for hierarchical text clustering

The proposed algorithm merges two approaches: divisive and agglomerative. For the divisive approach, this research proposes a divisive algorithm using a new objective function based on the standard FA and a re-locating algorithm. The re-locating algorithm changes the location of documents between clusters based on an identified similarity measurement. The re-locating algorithm helps to enhance the performance metrics (i.e. purity) that happen in the early stage of clustering. On the other hand, the Un-weighted Pair Group Method with Arithmetic Mean (UPGMA) is adapted as

the agglomerative approach. An alternative mean of merging clusters is introduced to merge two similar clusters. The enhanced UPGMA reduces the number of  $k$  clusters which later contributes to the performance of text clustering.

The proposed hierarchical text clustering contains four main phases: Data Acquisition, Clustering, Cluster Refining, and Cluster Merging. Figure 3.3 illustrates the phases of the proposed hierarchical text clustering.

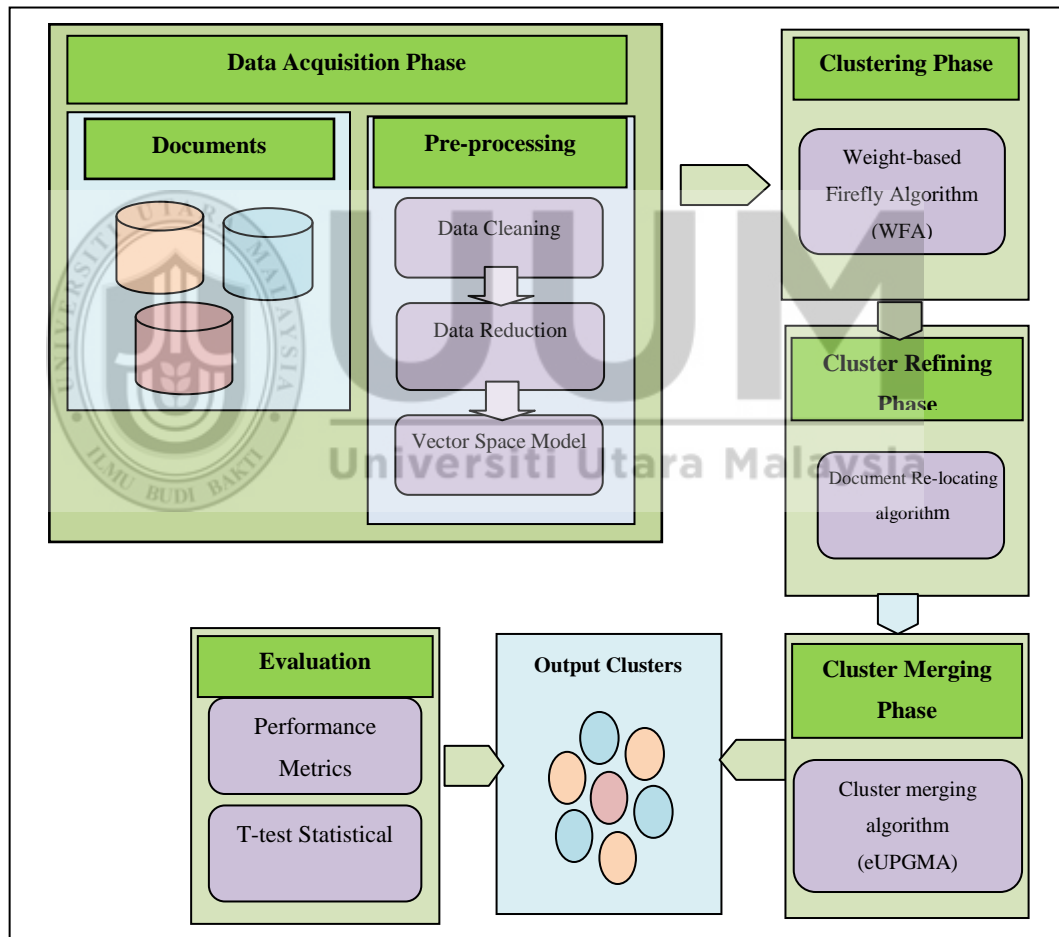


Figure 3.3. The phases of proposed hierarchical text clustering

### 3.1.1 Data Acquisition Phase

Data acquisition includes two steps: data collection and data pre-processing.

### 3.1.1.1 Data Collection

In this research, six benchmark datasets from different resources are employed. The datasets include 20Newsgroups (20NewsgroupsDataSet, 2006; Bache & Lichman, 2013), Reuters-21578 (Lewis, 1999), and TREC collection (Karypis, 2002). Samples of the datasets are presented in Appendix A. The 20Newsgroups and Reuters-21578 datasets are balanced datasets, where each class in the datasets includes the same number of documents, while the TREC collection datasets represent the un-balanced datasets as the numbers of documents in each class are different. Table 3.1 summarizes the characteristics of these datasets.

Table 3.1  
*Description of Datasets.*

Datasets	Resources	Number of documen ts	Number of classes	Minimum number of documents in class	Maximum number of documents in class	Number of terms
20Newsgroups	20 Newsgroup datasets	300	3	100	100	2275
Reuters	Reuters-21578	300	6	50	50	1212
TR11	TREC Collection from CLUTO toolkit	414	9	6	132	6429
TR12		313	8	9	93	5804
TR23		204	6	6	91	5832
TR45		690	10	14	160	8261

The first dataset which is denoted by 20Newsgroups (Bache & Lichman, 2013) was extracted from the UCI machine learning repository and is available online (<http://archive.ics.uci.edu/ml>). The 20Newsgroups dataset contains 300 documents

that it distributed in three different classes, which are: hardware, baseball and electronic. Each class includes 100 documents and the number of terms involved is 2275.

The second dataset, Reuters, comes from Reuters-21578 (Lewis, 1999), and was also obtained from the UCI machine learning repository. It includes 300 documents with six different classes, which include: earn, sugar, trade, ship, money-supply and gold. Each class contains 50 documents and the number of terms is 1212.

The four remaining datasets are TR11, TR12, TR23 and TR45, and were retrieved from CLUTO toolkit (Karypis, 2002). These datasets have already been pre-processed by Zhao and Karypis (2001), and they originated from the Text Retrieval Conference (TREC) collections (TREC, 1999). TR11 includes 414 documents from nine different classes and the number of terms is 6429. TR12 contains 313 documents from eight different classes with 5804 terms. TR23 includes 204 documents distributed in six classes and the number of terms is 5832. The last dataset, TR45, contains 690 documents from ten classes and consists of 8261 terms.

#### **3.1.1.2 Data Pre-processing**

The data pre-processing phase is a significant phase in text mining, where it extracts features from large documents and represents them in the form of a database. This phase contains two steps:

## Step 1: Data Cleaning

Figure 3.4 shows an example of document source code from the Reuters database. In this step, documents are entered as source code. Later, the text will be extracted from them which are the tags that contain title and body.

```
<PEOPLE></PEOPLE>
<ORGS></ORGS>
<EXCHANGES></EXCHANGES>
<COMPANIES></COMPANIES>
<UNKNOWN>
&#5;&#5;&#5;F
&#22;&#22;&#1;f0772&#31;reute
r f BC-COBANCO-INC-&lt;CBCO>-YE 02-26 0058</UNKNOWN>
<TEXT>&#2;
<TITLE>COBANCO INC &lt;CBCO> YEAR NET</TITLE>
<DATELINE> SANTA CRUZ, Calif., Feb 26 -</DATELINE>
<BODY>
    Shr 34 cts vs 1.19 dlrs
    Net 807,000 vs 2,858,000
    Assets 510.2 mln vs 479.7 mln
    Deposits 472.3 mln vs 440.3 mln
    Loans 299.2 mln vs 327.2 mln
    Note: 4th qtr not available. Year includes 1985
    extraordinary gain from tax carry forward of 132,000 dlrs, or
    five cts per shr.
    Reuter
    &#3;
```

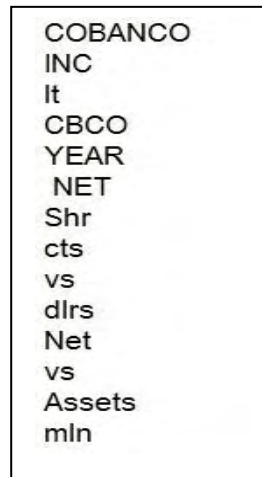
Figure 3.4. An example of document from the Reuters dataset

The selected text is cleaned from special characters and digits. Upon the completion of data cleaning, the output will be such as in Figure 3.5.

```
COBANCO INC It CBCO YEAR NET
Shr cts vs dlrs
Net vs
    Assets mln vs mln
    Deposits mln vs mln
    Loans mln vs mln
    Note: th qtr not available. Year includes
    extraordinary gain from tax carry forward of dlrs or
    five cts per shr.
Reuter
```

Figure 3.5. An example of a cleaned document

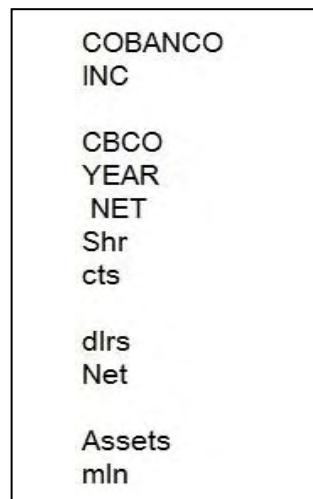
The second step is to split the cleaned text into words. Figure 3.6 shows an example of the output.



```
COBANCO
INC
It
CBCO
YEAR
NET
Shr
cts
vs
dlrs
Net
vs
Assets
mln
```

*Figure 3.6.* An example of extracted terms

In the third step, all of the words in each document are analyzed for their length. Words with a length of at least three are left for further processing. Figure 3.7 shows an example of words with the length greater than two.



```
COBANCO
INC

CBCO
YEAR
NET
Shr
cts

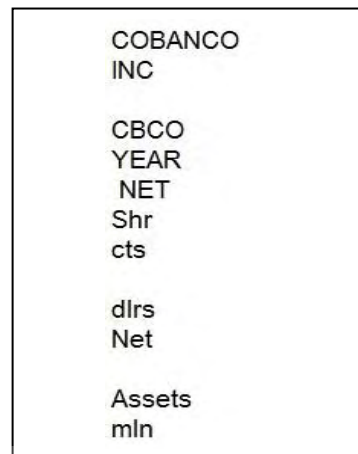
dlrs
Net

Assets
mln
```

*Figure 3.7.* An example of words with the length more than two



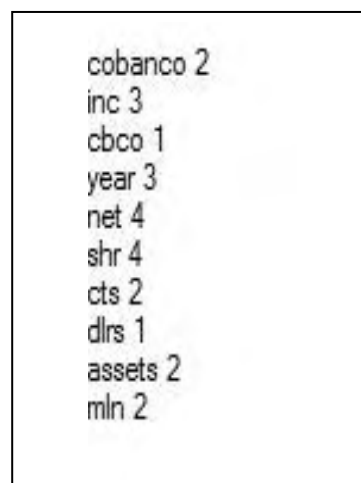
Later, the word is removed if it is listed as stop words, as shown in Appendix B. Examples of stop words are the, they, that, etc. Figure 3.8 shows an example of the removed stop words.



```
COBANCO
INC
CBCO
YEAR
NET
Shr
cts
dlrs
Net
Assets
mln
```

*Figure 3.8: An example of the removed stop words.*

The fifth step is stemming all the words by returning them to their original form such as “running” becomes “run”, and “evaluated” becomes “evaluate”. Finally, the frequency of the words is calculated as shown in Figure 3.9.

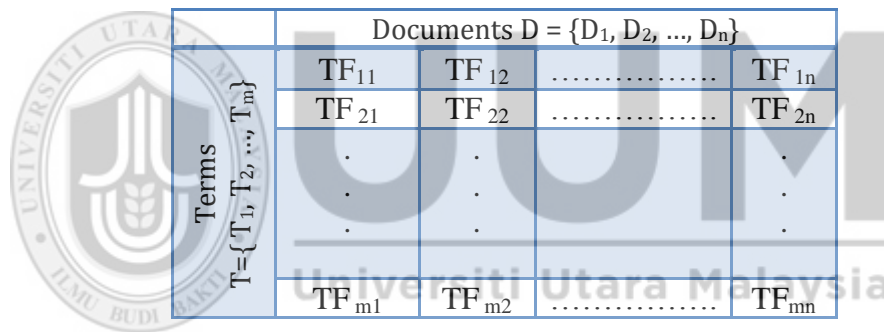


```
cobanco 2
inc 3
cbco 1
year 3
net 4
shr 4
cts 2
dlrs 1
assets 2
mln 2
```

*Figure 3.9. An example of word frequency*

## Step 2: Data Representation

Vector Space Model (VSM) is widely used in text mining to represent the words in documents. In VSM, let  $D = \{D_1, D_2, \dots, D_n\}$  be a document collection and  $n$  represents the number of documents in the collection. Let  $T = \{T_1, T_2, \dots, T_m\}$  be the terms in each documents and  $m$  represents the number of terms. In vector space model, the document  $D$  is represented as a vector in the  $m$  dimensional space (Aliguliyev, 2009a, 2009b). The vector  $D$  is related with the terms by a degree value. This degree is the occurrence of the term in the document; this relation is called term frequency ( $TF$ ). Figure 3.10 shows the term frequency matrix.



	Documents $D = \{D_1, D_2, \dots, D_n\}$			
Terms $T = \{T_1, T_2, \dots, T_m\}$	$TF_{11}$	$TF_{12}$	.....	$TF_{1n}$
	$TF_{21}$	$TF_{22}$	.....	$TF_{2n}$
	.	.		.
	.	.		.
	$TF_{m1}$	$TF_{m2}$	.....	$TF_{mn}$

Figure 3.10. The term frequency matrix

When the number of documents increases, the term frequency matrix will also increase. This can cause high dimensionality of dataset. There are some benefits of dimensionality reduction: 1) The clustering algorithm works better in lower attributes; 2) The model can understand in fewer attributes; 3) Data can visualize easily; and 4) The time of execution and memory can be reduced in lower dimensionality (Tan, Steinbach, & Kumar, 2006). In this research, data

dimensionality will be reduced by removing the terms that appear in all documents. These terms are not useful to discriminate the documents (Aggarwal & Zhai, 2012).

The Euclidean Normalized database, known as *EN*, is the second step in creating a vector space model. Where, the occurrence of terms is normalized to the value between (0, 1) through, initially calculating the document length using Equation 3.1 (Manning, Raghavan, & Schütze, 2008).

$$Length = \sqrt{\sum_{i=1}^m TF_i(d)^2} \quad (3.1)$$

where, TF is the term frequency and *d* is the document. Later, term frequency is divided over document length using the Equation 3.2 (Manning, Raghavan, & Schütze, 2008).

$$EN = \frac{TF}{Length} \quad (3.2)$$

The weight of every term in the document is calculated using TFIDF (Term Frequency-Inverse Document Frequency). The inverse documents frequency, *idf*, is calculated using Equation 3.3 (Manning, Raghavan, & Schütze, 2008).

$$idf = \log N/dft \quad (3.3)$$

where, N is the number of documents in the collection, and *dft* is the number of documents containing a term in the collection. Then, the weight of the term is calculated using Equation 3.4 (Manning, Raghavan, & Schütze, 2008).

$$tfidf_{t,d} = tf_{t,d} * idf_t \quad (3.4)$$

TFIDF is one of the most famous techniques that is used to represent documents as numerical weights in the search space. Figure 3.11 illustrates the TFIDF matrix.

	Terms $T=\{T_1, T_2, \dots, T_m\}$			
Documents $D = \{D_1, D_2, \dots, D_n\}$	$tfidf_{1,1}$	$tfidf_{1,2}$	.....	$tfidf_{1,m}$
	$tfidf_{2,1}$	$tfidf_{2,2}$	.....	$tfidf_{2,m}$
	.	.		.
	.	.		.
	$tfidf_{n,1}$	$tfidf_{n,2}$	.....	$tfidf_{n,m}$

Figure 3.11. TFIDF matrix

### 3.1.2 Clustering Phase

This section includes the flow of the proposed Weight-based Firefly Algorithm (WFA) (refer to Figure 3.12). As mentioned previously, each document has a relation with terms, which is term weight, *TFIDF*. Documents that include large term weights have a higher probability to be represented as the center of clusters. As learned in the literature, whenever the term frequency increases in one document and appears in a small number of documents, then, this term will be more discriminative for splitting the documents. The increase of term frequency in one document leads to the increase of the total weight of the document. The total weight of a document is the summation of term weights in that document. The total weight of each document is calculated using Equation 3.5.

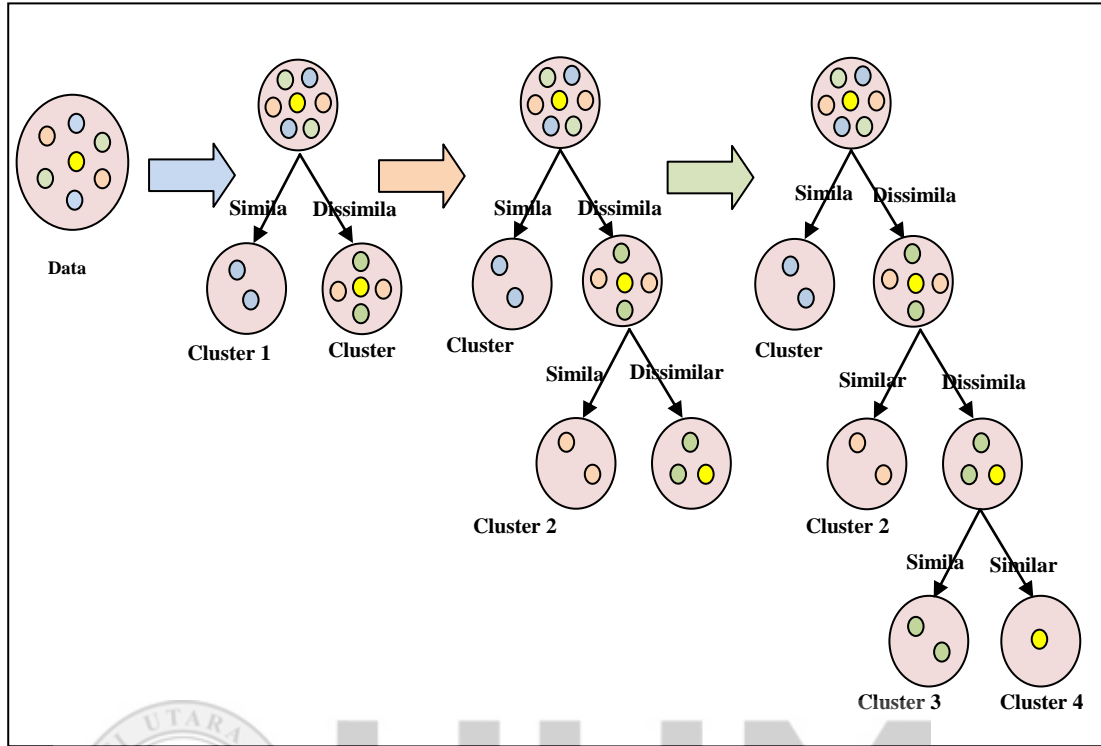


Figure 3.12. Flow of Hierarchical Text clustering using Weight-based Firefly Algorithm (WFA)

$$\text{Total Wiegght } (d_j) = \sum_{i=1}^m tfidf_{t_i, d_j} \quad (3.5)$$

where,  $j$  is the number of documents,  $i$  is the number of the terms. Figure 3.13 shows an example of the total weight matrix.

In the standard firefly algorithm (Yang, 2010b), it has two important factors: light intensity and attractiveness. The brighter firefly has a higher probability to become a center and attract other fireflies. In the proposed WFA algorithm, each document is represented by a single firefly. The total weight of each document represents the initial value of brightness,  $I_0$ , in WFA.

	Documents $D = \{D_1, D_2, \dots, D_n\}$			
Terms $T = \{T_1, T_2, \dots, T_m\}$	$tfidf_{1,1}$	$tfidf_{1,2}$	.....	$tfidf_{1,n}$
	$tfidf_{2,1}$	$tfidf_{2,2}$	.....	$tfidf_{2,n}$
	$\cdot$	$\cdot$		$\cdot$
	$\cdot$	$\cdot$		$\cdot$
	$\cdot$	$\cdot$		$\cdot$
	$tfidf_{m,1}$	$tfidf_{m,2}$	.....	$tfidf_{m,n}$
<b>Total Weight</b>	$\sum_{i=1}^m tfidf_{t_i, d_1}$	$\sum_{i=1}^m tfidf_{t_i, d_2}$	.....	$\sum_{i=1}^m tfidf_{t_i, d_n}$

Figure 3.13. An example of the total weight matrix

The attractive,  $\beta$ , of a firefly as shown in Equation 2.8 varies based on the distance between two fireflies (Yang, 2010b), where,  $\beta_0$  is the initial attractiveness, in this research, it sets to 1,  $\gamma$  is the light absorption coefficient and set to 1,  $r_{ij}$  is the distance between two documents  $i$  and  $j$ . The setting of the initial attractiveness and the light absorption coefficient to 1 make the attractiveness mostly related with the changing of distance between two fireflies and this setting is based on suggestion of Banati and Bajaj (2013), and Yang (2009).

The distance between two documents is calculated using the real position of the documents, where the initial position is the position of the document in the datasets. The distance is calculated using Cartesian distance (Yang, 2010b) as shown in Equation 2.9, where,  $X_i, X_j$  is any two documents in the data set. The Weight-based Firefly Algorithm (WFA) compares between two lights (two documents). The brightest light will attract the less bright, then, the position of the less bright finally will change based on Equation 2.7 (Yang, 2010b), where,  $X_i(t+1)$  is a new position of firefly,  $\alpha$  is a randomization parameter between (0, 1) (Yang, 2010b),  $\epsilon_i$  is a vector

of random numbers drawn from a Gaussian distribution or uniform distribution (Yang & He, 2013; Yang, 2010a). Later, the brightness of the wining firefly will be increased based on  $\beta$  value as shown in Equation 3.6.

$$\text{Light intensity } I(d_j)(t+1) = I(d_j)(t) + \beta \quad (3.6)$$

In Weight-based Firefly Algorithm (WFA), a firefly with the brightest light is used as a centroid (i.e. center of a cluster). Documents that are similar to the centroid will be identified using the most widely applied similarity measurement in text mining, which is the cosine similarity. Formula 3.7 represents the equation of cosine similarity (Luo, Li, & Chung, 2009).

$$\text{Cosine\_Similarity}(C_j) = \sum_{j=1}^m (d_j * O_j) \quad (3.7)$$

where,  $d_j$  is a document in cluster  $C_j$ ,  $O_j$  is the center of cluster  $C_j$ , and  $j$  is the number of terms in the collection. The value of  $d_j$  and  $O_j$  is taken from the Euclidean Normalized database. If the similarity is greater than the pre-defined threshold, the document is assigned into the first cluster, otherwise into another cluster. Later, the documents in the second cluster undergo the same process, which is finding a new centroid by sorting the brightness of fireflies (documents). Figure 3.14 shows the introduced process in Weight-based Firefly Algorithm (WFA).

WFA will evaluate the output cluster by measuring the average distance between each center and documents of clusters using the Euclidean distance function (Hassanzadeh & Meybodi, 2012) which is shown in Equation 3.8.

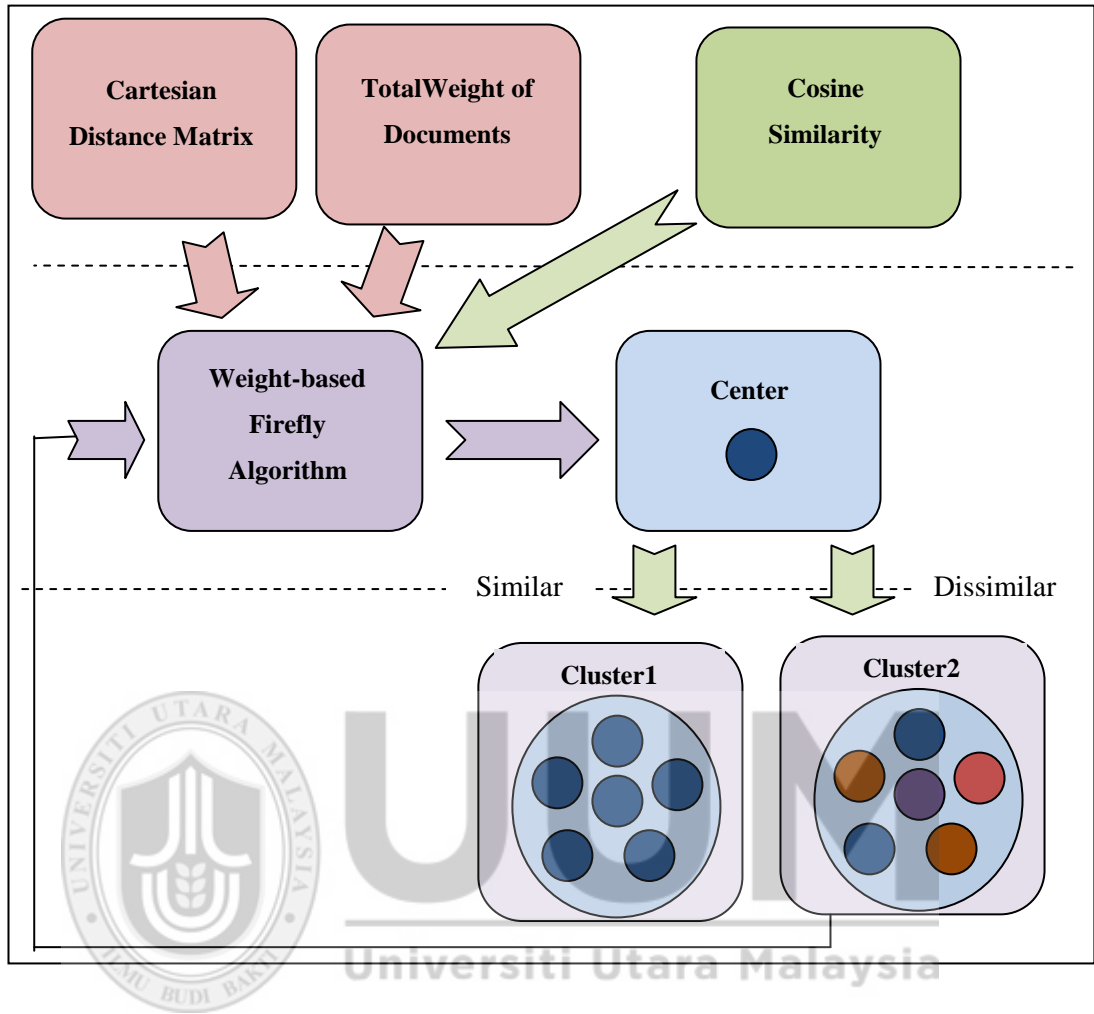


Figure 3.14. The process of Weight-based Firefly Algorithm (WFA)

$$EuclideanDistance(O_j, d_j) = \sqrt{\sum_{i=1}^m (d_{j,i} - O_{j,i})^2} \quad (3.8)$$

where,  $O_j$  is the center of the cluster  $j$ ,  $d_j$  is the documents in the cluster. The value of  $O_j$  and  $d_j$  is derived from TFIDF. A detail of the proposed WFA is presented in Chapter Four.



### 3.1.3 Cluster Refining Phase

This phase includes re-locating algorithm. Re-locating refers to the process of changing the location of a document from one cluster to another. An example of the process is illustrated in Figure 3.15.

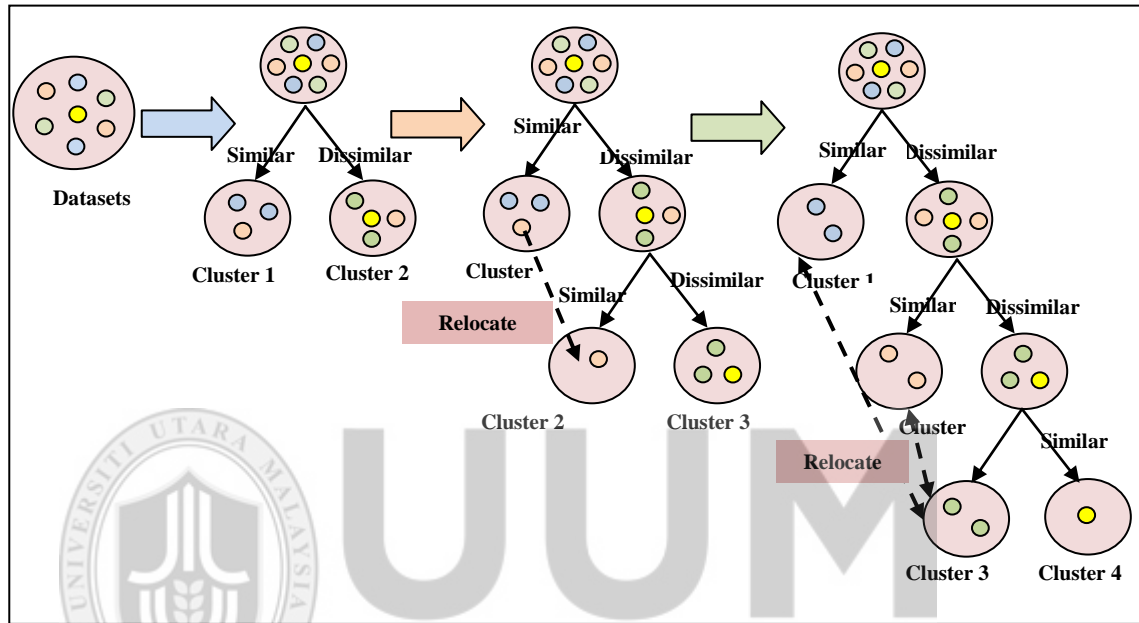
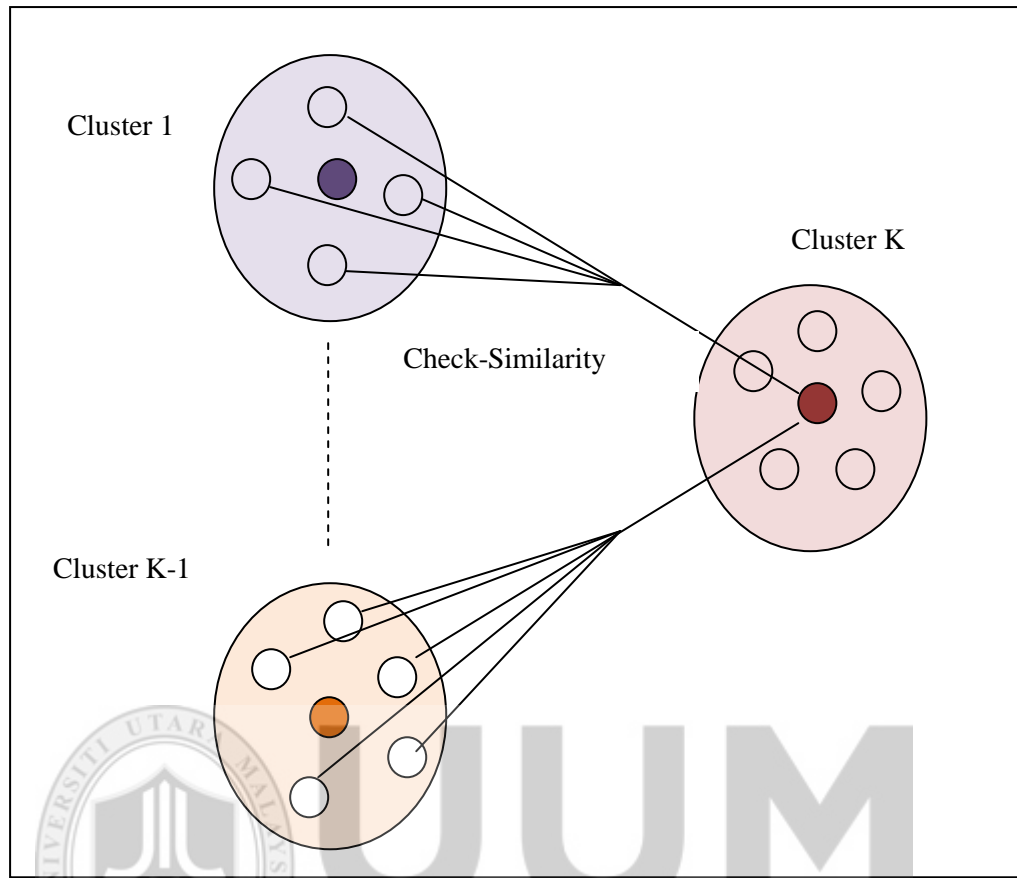


Figure 3.15. Process of document re-locating

The proposed document re-locating starts to operate once the construction of the second cluster is completed (using the Weight-based Firefly Algorithm (WFA)). When the second cluster is constructed, the document re-locating algorithm is invoked to check the similarities between the centroids of the second cluster (newly constructed) and the content (documents) of the first cluster. Whenever a new cluster is created, the process of document re-locating is repeated. Figure 3.16 shows the comparison in document re-locating.

For example, assume  $k=3$ , so Figure 3.16 includes three clusters. Assume Cluster 1 contains five documents (D1, D3, D5, D7 and D9) and the center is D5.



*Figure 3.16.* Comparison between clusters for document re-locating

Further, assume Cluster 2 contains six documents (D2, D8, D10, D12, D13 and D16) and the center is D10. A newly constructed cluster is Cluster 3 that contains six documents (D4, D6, D11, D14, D15 and D17) and D6 is the center. The document re-locating algorithm will check the similarity between D6 (center of Cluster 3) firstly with all documents in Cluster 1 (D1, D3, D7 and D9) excluding the center D5. If the similarity of D6 with any document in Cluster 1 is higher than the similarity of D5 with any document in Cluster 1, then the location of the particular document is changed from Cluster 1 to Cluster 3. Later, the document re-locating algorithm will check the similarity between D6 (center of Cluster 3) and all documents in Cluster 2,

where the same process with Cluster1 is repeated. A detailed explanation on the document re-locating algorithm is presented in Chapter Five.

#### **3.1.4 Cluster Merging Phase**

In the cluster merging phase, this research proposed an enhanced Un-weighted Pair Group Method with Arithmetic Mean (eUPGMA). It includes two steps: merge clusters and refine merged clusters.

The proposed merge clusters step is based on the Un-weighted Pair Group Method with Arithmetic Mean (UPGMA) (Manning, Raghavan, & Schütze, 2008; Yujian & Liye, 2010) and starts to operate after the clustering and refining phases are completed. Figure 3.17 illustrates the structure of the merging similar clusters process introduced in eUPGMA.

The refine merged clusters step chooses clusters that exceed an identified threshold. Experimentally, in this research two threshold (values are utilized;  $(50, n/20)$  and  $(50, n/40)$ ), where  $n$  represents the number of documents in dataset). The first threshold is based on the criterion used by Tan et al. (2011a) and the idea of refine merging clusters is adopted from Picarougne et al. (2007). The first threshold  $(50, n/20)$  is proposed to be used on balanced datasets, while the second threshold  $(50, n/40)$  is used on un-balanced datasets. Once the clusters are selected, a new center for the cluster is identified. This is followed by assigning the un-selected clusters to the nearest centers.



documents in  $C_2$  are represented by the column. The value of the CSim matrix is equal to 1 if the document in  $C_1$  is similar to the document in  $C_2$ , else it equals 0. The similarity between the two documents is based on the threshold.

**Step 3:** If (the number of documents in cluster  $C_1 \geq 2$  and the number of documents in cluster  $C_2 \geq 2$ ) OR If (the number of documents in cluster  $C_1 \geq 3$  and the number of documents in cluster  $C_2 = 1$ ) OR If (the number of documents in cluster  $C_2 \geq 3$  and the number of documents in cluster  $C_1 = 1$ ) then

**Step 4:** Calculate the average similarity between the two clusters as in Equation 3.9.

$$\frac{1}{P_1} \sum_{i=1}^{P_1} \sum_{j=1}^{P_2} \frac{CSim(C_i, C_j)}{P_2} \quad (3.9)$$

where,  $P_1$  is the number of document in the first cluster,  $P_2$  is the number of document in the second cluster,  $C_i$  is the first cluster,  $C_j$  is the second cluster.

**Step 5:** Calculate the merge threshold as in Equation 3.10 below.

$$MergeThreshold (MT) = floor \left( \frac{round(\frac{P_1 * P_2}{2}) - 1}{P_1 * P_2} * 10 \right) / 10 \quad (3.10)$$

**Step 6:** If Equation 3.9 passed the merge threshold in Equation 3.10 as shown in Equation 3.11, then, combine two clusters  $C_1$  and  $C_2$  into one cluster.

$$\frac{1}{P_1} \sum_{i=1}^{P_1} \sum_{j=1}^{P_2} \frac{CSim(C_i, C_j)}{P_2} \geq MergeThreshold (MT) \quad (3.11)$$

**Step 7:** If (the number of documents in cluster  $C_1 \geq 2$  and the number of documents in cluster  $C_2 \geq 1$ ) OR If (the number of documents in cluster  $C_2 \geq 2$  and the number of documents in cluster  $C_1 \geq 1$ )

**Step 8:** Combine  $C_1$  and  $C_2$ , if Equation 3.11 is true using Equation 3.12 to obtain merge threshold.

$$\text{MergeThreshold } (MT) = \frac{\text{round}\left(\frac{P_1 * P_2}{2}\right)}{P_1 * P_2} \quad (3.12)$$

**Step 9:** If (number of documents in cluster  $C_1 \geq 1$  and the number of documents in cluster  $C_2 \geq 1$ )

**Step 10:** Combine  $C_1$  and  $C_2$ , if  $\text{CSim}(C_1, C_2)$  equals to 1.

### 3.2 Implementation of Algorithms

The three proposed algorithms are developed as follows:

- 1- The Weight-based Firefly algorithm (WFA) is implemented on text datasets that have been mentioned in Section 3.1.1.1. The results are extracted in order to compare them with the state-of-the-art methods as presented in Chapter Four.
- 2- The Weight-based Firefly algorithm (WFA) that includes the document re-locating algorithm denoted as  $\text{WFA}_R$  is executed on text datasets. The results are extracted in order to compare them with state-of-the-art methods as presented in Chapter Five.
- 3- As for the Weight-based Firefly algorithm (WFA) that includes the document re-locating algorithm and cluster merging algorithm denoted as  $\text{WFA}_{RM}$ , the results are extracted in order to compare them with the state-of-the-art methods as presented in Chapters Six and Seven.

In addition, Particle Swarm Optimization (PSO), K-means, Bisect K-means, FAK-means, BatK-means, PGSCM and DCPG have been implemented in the simulation program for a fair comparison with the proposed algorithms on the same text datasets. The implementations of the algorithms were carried out using Matlab version R2008a under Windows 8.

### 3.3 Evaluation

This section discusses the evaluation of the proposed algorithms. The evaluation includes two parts: based on performance metrics and Independent T-test. These evaluations are conducted in three parts as shown in the following.

- 1- Comparison between the Weight-based Firefly algorithm (WFA) against Particle Swarm Optimization (PSO), K-means, Bisect K-means and FAK-means as reported in Chapter Four.
- 2- Comparison between WFA<sub>R</sub> against Particle Swarm Optimization (PSO), K-means, Bisect K-means and FAK-means as reported in Chapter Five.
- 3- Comparison between the proposed WFA<sub>RM</sub> against Particle Swarm Optimization (PSO), K-means, Bisect K-means, FAK-means, BatK-means, PGSCM and DCPG as reported in Chapters Six and Seven.

The evaluation is based on clustering performance metrics that are presented in the following section.

### 3.3.1 Performance Metrics

The metrics involved in the evaluation of the clustering performance are of internal, external and relative quality metrics. The average distance between documents and center (ADDC) is used as the internal metrics, while the Davies-Bouldin Index (DBI) and Dunn Index (DI) are as the relative metrics. On the other hand, F-measure, Entropy and Purity are used as the external metrics.

#### 3.3.1.1 Internal and Relative Quality Metrics

**Average Distance between Documents and Centers (ADDC)** (Cui, Potok, & Palathingal, 2005; Forsati, Mahdavi, Shamsfard, & Meybodi, 2013) evaluates the compactness of the clustering solution, where a smaller ADDC value indicates a more compact solution. Equation 3.13 illustrates the ADDC metric.


$$ADDC = \sum_{j=1}^K \frac{\sum_{i=1}^{n_i} Ec(O_i, D_i)}{n_i} \quad (3.13)$$

where,  $K$  is the number of clusters,  $n_i$  is the number of documents in cluster  $i$ ,  $O_i$  is the center of cluster  $i$  and  $d_i$  is the document in cluster  $i$ , and  $Ec$  is the Euclidian distance (Murugesan & Zhang, 2011a, 2011b) that can be calculated by Equation 3.8.

**Davies-Bouldin Index (DBI)** measures the ratio with the cluster and between the average distances for the cluster (Davies & Bouldin, 1979). The DBI is calculated as shown in Equation 3.14.



$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{i \neq j} \frac{\delta_i + \delta_j}{d(i, j)} \quad (3.14)$$

where,  $k$  is the number of clusters,  $\delta_i$ ,  $\delta_j$  are the average distance of all objects in cluster  $i$  and  $j$ ,  $d(i, j)$  is the distance between the center of cluster  $i$  and center of cluster  $j$ . The good cluster algorithm has a lower value of DBI.

**Dunn Index (DI)** measures the ratio between the minimum inter cluster distance and the maximum intra cluster distance (Dunn, 1974). DI is calculated as shown in Equation 3.15.

$$DI = \min_{1 \leq i \leq n} \left\{ \min_{1 \leq j \leq n, i \neq j} \left\{ \frac{d(i, j)}{\max_{1 \leq k \leq n} d(k)} \right\} \right\} \quad (3.15)$$

where,  $d(i, j)$  is the distance between cluster  $i$  and cluster  $j$ .  $d(k)$  is the maximum intra cluster between any two objects in cluster  $k$ . The high value of DI means good cluster quality.

### 3.3.1.2 External Quality Metrics

The goodness of a cluster is measured using three external measures: Entropy, F-Measure, and Purity (Kashef & Kamel, 2010; Murugesan & Zhang, 2011a, 2011b).

**Entropy** is a measure of goodness and randomness (Murugesan & Zhang, 2011a, 2011b; Shannon, 1948). The entropy of output cluster  $C_j$  is in Equation 3.16.

$$H(j) = - \sum_{k=1}^c \frac{|\theta_k \cap C_j|}{|C_j|} \log \frac{|\theta_k \cap C_j|}{|C_j|} \quad (3.16)$$

where,  $C_j$  is the output clustering from the clustering algorithm,  $|C_j|$  represents the number of document in cluster  $C_j$ ,  $\Theta_k$  is known class and  $c$  is the number of known classes,  $|\Theta_k \cap C_j|$  represents the number of document in a class  $\Theta_k$ , in a cluster  $C_j$  and in both class  $\Theta_k$  and cluster  $C_j$  respectively. The entropy value for a clustering algorithm is calculated using Equation 3.17.

$$H = \sum_{j=1}^k \frac{H_j * |C_j|}{N} \quad (3.17)$$

where,  $N$  is the number of documents in the collection.

**F-Measure** measures the accuracy (Murugesan & Zhang, 2011a, 2011b) as it depends on the recall and precision values (Meghabghab & Kandel, 2008). The recall measure formula is shown in Equation 3.18 while Equation 3.19 represents the precision.

$$Recall R(\Theta_k, C_j) = \frac{|\Theta_k \cap C_j|}{|C_j|} \quad (3.18)$$

$$Precision P(\Theta_k, C_j) = \frac{|\Theta_k \cap C_j|}{|\Theta_k|} \quad (3.19)$$

where,  $|\Theta_k|$  represents the number of document in a class  $\Theta_k$ . The F-measure formula is shown in Equation 3.20.

$$F - measure = \frac{2 * R(\Theta_k, C_j) * P(\Theta_k, C_j)}{R(\Theta_k, C_j) + P(\Theta_k, C_j)} \quad (3.20)$$

The total F-measure is the summation average of F-measures for all classes. It depends on the maximum value of F-measure for all of the classes as Equation 3.21 shows the maximum value of F-measure.

$$F(\Theta_k) = \max_{C_j \in \{C_1, \dots, C_k\}} (F(\Theta_k, C_j)) \quad (3.21)$$

The equation for total F-measure is shown in Equation 3.22.

$$Total\ F - measure = \sum_{k=1}^c \frac{|\Theta_k|}{N} * F(\Theta_k) \quad (3.22)$$

**Purity** is a measure of cluster quality (Murugesan & Zhang, 2011a, 2011b). It depends on the maximum number of documents in class  $\Omega_k$  and in cluster  $C_j$  respectively. The formula of purity is shown in Equation 3.23.

$$P(\Theta_k, C_j) = \max_k |\Theta_k \cap C_j| \quad (3.23)$$

The cluster purity is calculated as in Equation 3.24.

$$Purity = \sum_{\Theta_k \in \{\Theta_1, \dots, \Theta_c\}} \frac{P(\Theta_k, C_j)}{N} \quad (3.24)$$

### 3.3.2 Statistical Analysis

The T-test is a statistical hypothesis test which “is a function of the sample data and critical region” (Ross, 2010). There are two types of hypotheses; the null hypothesis that is denoted as  $H_0$  and the alternative hypothesis that is called  $H_1$ . The null hypothesis ( $H_0$ ) tests if the test statistic value (p-value) falls in the critical area or not, if it falls in the critical area, the null hypothesis is rejected, otherwise not, while the alternative hypothesis is accepted if the null hypothesis is rejected. The probability to reject the null hypothesis is based on the identified value  $\alpha$ , named as the significance level of the test. The values of the significance level are  $\alpha=0.10$ ,  $\alpha=0.05$ , and  $\alpha=0.01$ , where,  $\alpha=0.05$  is the most common used value.

The hypotheses for Independent Samples T-test can be expressed in mathematical equivalents.

$H_0: \text{Mean}(WFA_{RM}) = \text{Mean}(\text{any static methods})$

$H_1: \text{Mean}(WFA_{RM}) \neq \text{Mean}(\text{any static methods})$

Where,  $\text{Mean}(WFA_{RM})$  and  $\text{Mean}(\text{any static methods})$  are the means of the population for  $WFA_{RM}$  and any static methods. The null hypothesis is rejected if (P-value  $< 0.05$ ) and the alternative hypothesis is accepted, otherwise, if (P-value  $> 0.05$ ), the null hypothesis is accepted.

### 3.4 Summary

The main objective of this research is to design a hierarchical clustering algorithm for text documents based on Firefly Algorithm. The three main phases in the proposed hierarchical clustering algorithm design includes: clustering using Weight-based Firefly Algorithm, refining using document re-locating algorithm, and merging using eUPGMA algorithm. In the clustering phase, this research proposes an objective function for the firefly algorithm that is based on the total weight of documents. The objective function used in WFA will increase the light of a document (initial total weight) based on the distance between two documents. In this research, the goal of WFA is to determine the center of clusters by identifying the one with the brightest light (i.e. document with the highest total weight).

In the cluster refining phase, this research proposes a document re-locating algorithm, known as  $WFA_R$ , which changes the location of documents between

clusters when a new cluster is constructed. The aim of  $WFA_R$  is to produce more compact clusters which lead to enhancement of the quality performance (i.e. purity).

On the cluster merging phase, this research proposes an enhanced Un-weighted Pair Group Method with Arithmetic Mean (eUPGMA). It includes two steps: merge clusters and refine merged clusters. The goal of eUPGMA is to obtain the optimal clusters.

The proposed Firefly Algorithm is later compared against existing state-of-the-art clustering algorithms, such as Particle Swarm Optimization (PSO), K-means, Bisect K-means, Hybrid Firefly algorithm with K-means, Hybrid Bat algorithm with K-means, General Stochastic Clustering method and Hybrid Genetic algorithm with Particle Swarm Optimization. The evaluation of the proposed algorithm is based on performance metrics, namely internal, external and relative metrics, using text datasets. Furthermore, the evaluation is also based on T-test statistical hypotheses.

## CHAPTER FOUR

### DOCUMENT CLUSTERING

#### 4.1 Weight-based Firefly Algorithm (WFA)

Weight-based Firefly Algorithm (WFA) is designed based on Firefly Algorithm which was introduced to solve the problem of optimization. In WFA, the number of  $k$  clusters is not pre-determined, but will be automatically identified. The proposed WFA operates through two (2) stages: initialization of parameters and data clustering.

##### 4.1.1 Initialization of Parameters

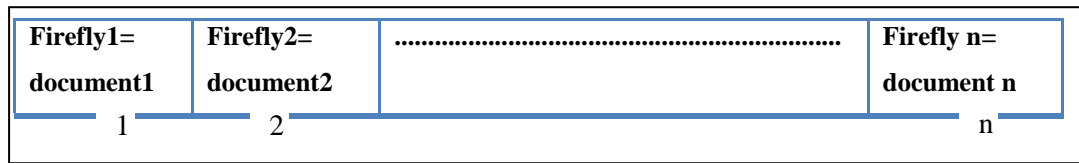
In order to operate WFA, the required input includes the total weight of each document which represents the initial light of a firefly, the initial position for each firefly and a distance matrix that represents the distance between all fireflies.

In WFA, the initial light of a firefly is represented by the total weight of the document using Equations 4.1 and 4.2.

$$\text{InitialLight } I(d_j) = \text{Total Wiegth } (d_j) \quad (4.1)$$

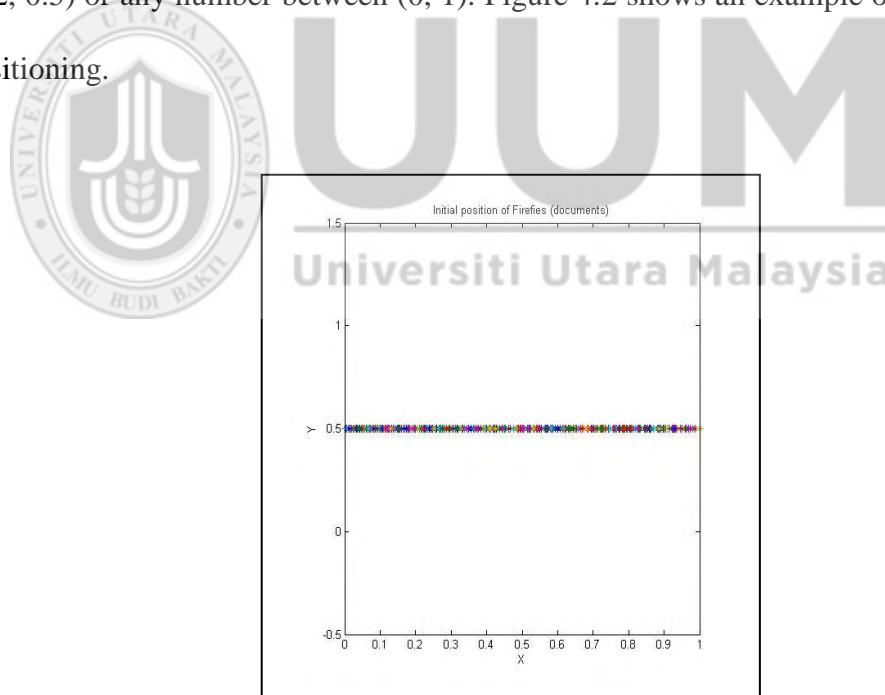
$$\text{Total Wiegth } (d_j) = \sum_{i=1}^m tfidf_{t_i, d_j} \quad (4.2)$$

Upon obtaining the total weight of a document, the search space reduces to one dimension as shown in Figure 4.1.



*Figure 4.1.* One dimension search space

The initial positioning of a firefly in the search space is represented by two coordinates, X and Y. In this study, the value of Y is fixed as (0.5) (because the search space is one dimension), while the value of X can be obtained using the normalization method, where the search space is represented between (0, 1). For example, the firefly that represents a document may be positioned at (0.7, 0.5) or (0.2, 0.5) or any number between (0, 1). Figure 4.2 shows an example of normalized positioning.



*Figure 4.2.* An example of normalized positioning

In this thesis, the initial positioning is normalized in the range of (0, 1) using a sample normalization process which is called Rescaling Method (Yunrong & Liangzhong, 2009). Equation 4.3 illustrates the Rescaling Method.

$$X^o = \frac{(X - \min(X))}{(\max(X) - \min(X))} \quad (4.3)$$

Where,  $X^o$  is the normalized positioning,  $X$  is the current positioning.

The third parameter required for the operation of WFA is the distance between fireflies. It is the distance between two positions which is calculated using the Cartesian distance (Yang, 2010b) as shown in Equation 4.4.

$$Cartesian\_distance(X_j, X_i) = \sqrt{(X_j - X_i)^2} \quad (4.4)$$

where,  $X_j$  is the position of first firefly and  $X_i$  is the position of second firefly.

#### 4.1.2 Data Clustering

The data clustering phase in WFA includes two processes: identification of center and construction of clusters. These two processes are repeated until all data (documents) are clustered. The second process starts after the first center has been identified and it will return to the first process once the first cluster is created.

In the standard Firefly Algorithm (FA) (Yang, 2010b), every firefly needs to be compared (in terms of the brightness) with all other fireflies. Figure 4.3 illustrates an example of the competition between fireflies. Assume, the triangle symbol represents Class 1 and the circle symbol represents Class 2. Further, assume firefly A has the total weight of 20. It will need to compete with all other fireflies. Such a process requires high computational effort (i.e. processing time). So, in the WFA algorithm, the exploration of the firefly is improved by introducing a condition that permits the comparison of brightness if and only if the documents are similar. Figure 4.4 shows



an example of the competition between fireflies in WFA. Firefly A that has a total weight of 20 will only need to compete with five similar fireflies.

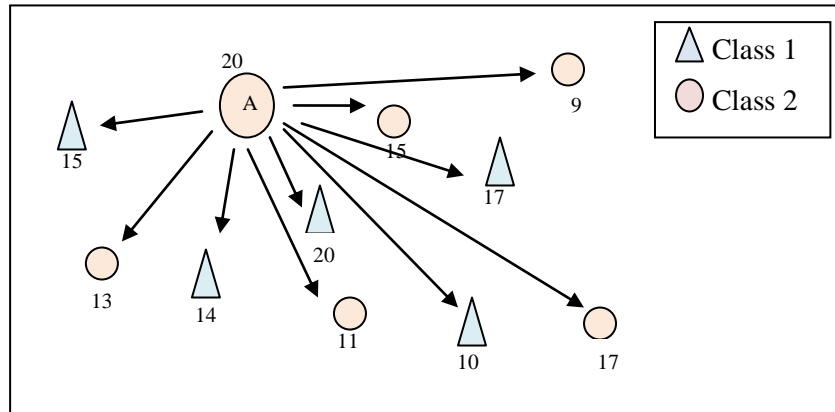


Figure 4.3. An example of competition in standard Firefly Algorithm (FA)

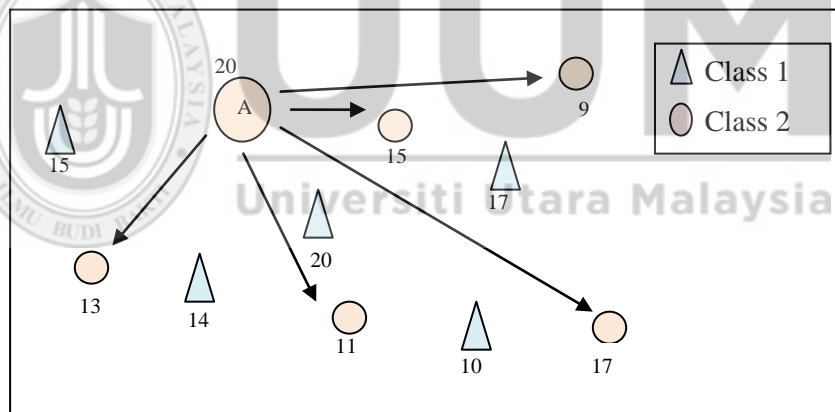


Figure 4.4. An example of competition in Weight-based Firefly Algorithm (WFA)

In detail, the process of center identification in WFA is based on two conditions: the first condition is based on the brightness of fireflies, and the second condition relies on the similarity (i.e. cosine similarity) between two documents. If there exists a document that passed the specified similarity threshold (the setting of similarity threshold is illustrated in Table 4.1), then, the movement of the less bright firefly

towards the brightest firefly is executed using Equation 2.7 (Yang, 2010b), and the light intensity of the brightest firefly is updated (i.e. increased) using Equation 4.8. In Equation 2.7,  $X_i$  is the position of the less bright firefly and  $X_j$  is the position of the brighter one.  $\alpha$  is a randomization parameter between (0, 1) (Yang, 2010b). In this study, experimentally,  $\alpha$  is set to 0.2, while,  $\beta$  (the attractiveness between two fireflies) is obtained using Equation 2.8 (Yang, 2010b), where,  $\beta_0$  is the initial attractiveness and in this algorithm sets to 1,  $Y$  is the absorption coefficient and also sets to 1. The  $r_{ij}$  is the distance between document  $i$  and document  $j$  which is computed by Equation 4.4.

The value of  $\varepsilon_i$  in Equation 2.7 is a vector of random numbers drawn from a Gaussian distribution or uniform distribution (Yang & He, 2013; Yang, 2010a). For example, the simplest form of  $\varepsilon_i$  can be replaced by  $(\text{rand} - 1/2)$  where  $\text{rand}$  refers to the random number uniformly distributed in  $[0 \text{ and } 1]$  (Yang, 2010b). In this study, the  $\varepsilon_i$  value is an adaptive value that is computed using Equations 4.5, 4.6 and 4.7. It is in the range between the minimum *TFIDF* of two documents and the maximum *TFIDF* of the same documents.

$$\varepsilon_i = \text{random}(\text{Min } TFIDF, \text{Max } TFIDF) \quad (4.5)$$

$$\text{Min } TFIDF = \alpha * \text{Min}(TFIDF_i, TFIDF_j) \quad (4.6)$$

$$\text{Max } TFIDF = \alpha * \text{Max}(TFIDF_i, TFIDF_j) \quad (4.7)$$

Finally, the light of the brighter firefly increases based on the value of attractiveness  $\beta$  which depends on the distance between two fireflies as shown in Equation 4.8.

$$\text{light intensity } I(d_j)(t+1) = I(d_j)(t) + \beta \quad (4.8)$$

The competition between fireflies continues until it reaches the predefined number of iteration. Then, the fireflies are ranked based on their brightness, where the brightest firefly is identified as a centroid. Once this is done, clusters can be constructed, where documents that have a high similarity value with the centroid (using cosine similarity) are assigned to the same cluster with the centroid (first cluster). On the other hand, the ones with lower values will be assigned in another cluster (second cluster). Such an approach requires a pre-defined threshold value; in this study, each dataset has a different threshold value (the best setting is chosen experimentally) and are shown in Table 4.1.

Table 4.1

*Parameters setting in WFA.*

Datasets	Similarity threshold in identifying centers	Similarity threshold in constructing clusters
20Newsgroups	0.2	0.15
Reuters	0.2	0.15
TR11	0.3	0.3
TR12	0.4	0.25
TR23	0.3	0.3
TR45	0.3	0.25

The process of clustering construction repeats on the second cluster where the fireflies carrying documents in cluster two is sorted to identify the one with the brightest light. The formulation of cluster two continues by identifying similar

documents and the process is completed once all documents are grouped into clusters.

On the other hand, the cosine similarity is as defined in Equation 4.9 (Luo, Li & Chung, 2009). The value of cosine similarity is in the range between (0, 1); when the value of cosine similarity approaches 1, this means the two documents are identical, and when it approaches 0, this means the two documents are far away and are not identical. Equation 4.9 displays the formula to calculate the Cosine similarity.

$$\text{CosineSimilarity}(d_i, d_j) = \frac{d_i * d_j}{||d_i|| * ||d_j||} \quad (4.9)$$

In this thesis, the Cosine similarity is based on the normalized term frequency (term frequency is normalized to the length of documents), hence Equation 4.9 becomes the following Equation 4.10 (Luo, Li & Chung, 2009).

$$\text{CosineSimilarity}(d_i, d_j) = \sum_{t=1}^m (d_{i,t} * d_{j,t}) \quad (4.10)$$

where, m is the number of terms in the collection,  $d_j$  and  $d_i$  are two different documents. The proposed WFA algorithm is presented in Figure 4.5.

The outcome of a clustering process is the constructed clusters and their centroid. In order to measure the quality of the produced clusters, performance metrics such as the average distance between center and documents in clusters (ADDC), DBI, DI, Purity, F-measure and Entropy, which are explained in the previous chapter in Section 3.3.1, are employed.

**Weight-based Firefly Algorithm (WFA)**

- Step 1:** Generate initial population of firefly randomly  $x^i$  where  $i=1, 2 \dots n$ ,  $n$ =number of fireflies (documents).
- Step 2:** Initial Light Intensity,  $I$ =total weight of document using Equations 4.1 and 4.2.
- Step 3:** Define light absorption coefficient  $\gamma$ , initial  $\gamma=1$ .
- Step 4:** Define the randomization parameter  $\alpha$ ,  $\alpha=0.2$ .
- Step 5:** Define initial attractiveness  $\beta_0 = 1.0$ .
- Step 6:** While  $t < \text{Maximum number of iteration}$  ( $t$ = number of iteration)
- Step 7:** For  $i=1$  to  $N$
- Step 8:** For  $j=1$  to  $N$
- Step 9:** IF (Light  $I_i < \text{Light } I_j$ ) (Light=total weight)
- Step 10:** IF (CosineSimilarity ( $i, j$ )  $\geq$  Threshold (CosineSimilarity using Equation 4.10))
- Step 11:** Calculate distance between  $i, j$  using Equation 4.4.
- Step 12:** Calculate attractiveness using Equation 2.8.
- Step 13:** Calculate random parameter  $\varepsilon_i$  using Equations 4.5, 4.6 and 4.7.
- Step 14:** Move document  $i$  to  $j$  using Equation 2.7.
- Step 15:** Update light intensity using Equation 4.8.
- Step 16:** End For  $j$
- Step 17:** End For  $i$
- Step 18:**  $t=t+1$
- Step 19:** End While
- Step 20:** Rank the Light List to find best document (brightest light) and represent as center.
- Step 21:** Find document similar to center using Equation 4.10 and construct cluster
- Step 22:** Remove produced clusters from Light List.
- Step 23:** Return to Step 20 until remains one document in Light List.
- Step 24:** Output clusters.

*Figure 4.5. Weight-based Firefly Algorithm (WFA) for hierarchical text clustering*

### Example

The following elaboration is on the process of applying the proposed WFA. The dataset that is used for this example includes thirty documents obtained from the 20Newsgroups dataset. It contains of three topics: Comp.sys.mac.hardware, Rec.sport.baseball and Sci.electronic. Figure 4.6 shows an example of TFIDF for the particular dataset.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	acm	adb	appl	articl	clone	clotho	comp	corpor	data	david	disk	engin	face	file	gmt	hardwar	hold
2	0.247566	0.14854	0.260837	0.026597	0.197113	0.198053	0.041318	0.118268	0.1676	0.06704	0.087996	0.058664	0.078845	0.118268	0.000987	0.02361	0.07884
3	0	0	0	0.02451	0	0	0.021758	0	0	0	0	0.162184	0	0	0.001819	0.021758	0
4	0	0	0	0.035313	0	0	0.020898	0	0	0	0	0	0	0	0.001747	0.020898	0
5	0	0	0	0.032909	0	0	0.048689	0	0	0	0.096782	0	0	0	0.001628	0.029214	0
6	0	0	0	0.042657	0	0	0.037867	0	0	0	0	0	0	0	0.003166	0.047334	0
7	0	0	0	0	0	0	0.07875	0	0	0	0	0.111811	0	0	0.001881	0.03375	0
8	0	0	0	0.015203	0.135204	0	0.047235	0	0.07664	0	0.100597	0	0	0	0.001128	0.033739	0
9	0	0	0.056158	0	0	0	0.025417	0	0	0	0	0	0	0	0.002125	0.025417	0
10	0	0	0.053318	0	0	0	0.024131	0	0	0	0	0	0	0	0.002018	0.024131	0
11	0	0	0	0.045913	0	0	0.040757	0	0	0	0	0	0	0.272212	0.002272	0.040757	0
12	0	0	0	0.013705	0	0	0.066913	0	0.138178	0	0.181371	0	0	0	0.001017	0.030415	0
13	0	0	0.063116	0	0	0	0.042849	0	0	0	0	0	0	0	0.000796	0.014283	0
14	0	0	0	0	0	0	0.022593	0	0	0	0	0.11227	0	0	0.001889	0.022593	0
15	0	0	0	0.036836	0	0	0.032699	0	0	0	0	0	0	0	0.002734	0.032699	0
16	0	0	0	0.023583	0	0	0.020935	0	0	0	0	0	0	0	0.00175	0.020935	0
17	0	0	0	0.022982	0	0	0.020401	0	0	0	0	0	0	0	0.001706	0.020401	0
18	0	0	0	0	0	0	0.027756	0	0	0	0	0	0	0	0.002321	0.027756	0
19	0	0	0	0	0	0	0.041505	0	0	0	0	0	0	0	0.001388	0.024903	0
20	0	0	0	0	0	0	0.025618	0	0	0	0	0	0	0	0.002142	0.025618	0
21	0	0	0.199994	0.016994	0	0	0.0528	0.100756	0	0	0	0	0	0	0.001261	0.037715	0
22	0	0	0	0.020939	0	0	0	0	0	0	0	0	0	0	0.001554	0	0
23	0	0	0	0.036522	0	0	0	0	0	0	0	0	0	0	0.002711	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.002887	0	0
25	0	0	0	0.018498	0	0	0	0	0	0.139876	0	0	0	0	0.001373	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.002189	0	0

Figure 4.6. An example of TFIDF for 20Newsgroups

The proposed WFA first calculates the cosine similarity between documents using Equation 4.10. Figure 4.7 shows an example of the cosine similarity table. For example, the value of similarity between document 1 and document 2 can calculate as follows:

$$\text{Cosine Similarity } (d_1, d_2) = [(\text{first term})_{d_1} * (\text{first term})_{d_2}] + [(\text{second term})_{d_1} * (\text{second term})_{d_2}] + \dots + [(\text{m term})_{d_1} * (\text{m term})_{d_2}]$$




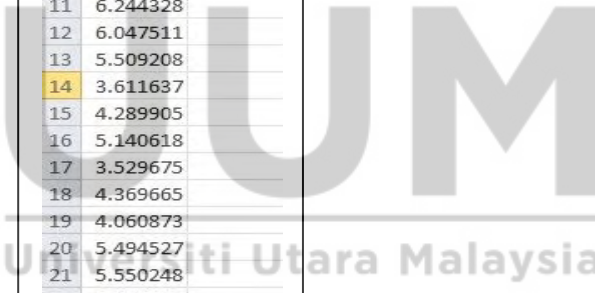
	A	B	C	D	E	F	G	H	I	J	K	Formula Bar	M	N	O
1	1	0.182237	0.159124	0.268781	0.20183	0.359775	0.324977	0.193528	0.16996	0.323266	0.242008	0.232007	0.141921	0.23342	0.1315
2	0.182237	1	0.208961	0.204987	0.192637	0.209191	0.153882	0.343312	0.25822	0.224023	0.145123	0.103561	0.241755	0.206501	0.1358
3	0.159124	0.208961	1	0.173917	0.226496	0.223671	0.202375	0.192709	0.199227	0.256374	0.141439	0.118722	0.17891	0.187323	0.1869
4	0.268781	0.204987	0.173917	1	0.217026	0.441624	0.245816	0.183583	0.189455	0.315721	0.259802	0.252679	0.21285	0.272128	0.203
5	0.20183	0.192637	0.226496	0.217026	1	0.243858	0.228671	0.155193	0.147344	0.286191	0.220997	0.234016	0.137949	0.219627	0.1533
6	0.359775	0.209191	0.223671	0.441624	0.243858	1	0.374572	0.253588	0.267028	0.384465	0.273663	0.333374	0.225412	0.266932	0.1784
7	0.324977	0.153882	0.202375	0.245816	0.228671	0.374572	1	0.157634	0.149662	0.298602	0.30049	0.245545	0.189284	0.213476	0.1389
8	0.193528	0.343312	0.192709	0.183583	0.155193	0.253588	0.157634	1	0.835692	0.228286	0.119665	0.130732	0.25	0.147415	0.141
9	0.16996	0.25822	0.199227	0.189455	0.147344	0.267028	0.149662	0.835692	1	0.21674	0.113613	0.13894	0.153842	0.13996	0.134
10	0.323266	0.224023	0.256374	0.315721	0.286191	0.384465	0.298602	0.228286	0.21674	1	0.223873	0.227368	0.202921	0.308024	0.2155
11	0.242008	0.145123	0.141439	0.259802	0.220997	0.273663	0.30049	0.119665	0.113613	0.223873	1	0.177455	0.137394	0.224516	0.1314
12	0.232007	0.103561	0.118722	0.252679	0.234016	0.333374	0.245545	0.130732	0.13894	0.227368	0.177455	1	0.152629	0.258562	0.1751
13	0.141921	0.241755	0.17891	0.21285	0.137949	0.225412	0.189284	0.25	0.153842	0.202921	0.137394	0.152629	1	0.291853	0.148
14	0.23342	0.206501	0.187323	0.272128	0.219627	0.266932	0.213476	0.147415	0.13996	0.308024	0.224516	0.258562	0.291853	1	0.1986
15	0.131509	0.135881	0.186949	0.20381	0.153395	0.178493	0.138951	0.14157	0.13441	0.215551	0.131421	0.175182	0.14872	0.198692	
16	0.139807	0.143153	0.19937	0.172986	0.186855	0.185046	0.150948	0.13378	0.150829	0.259218	0.124069	0.100237	0.13378	0.139842	0.1239
17	0.163789	0.209365	0.252535	0.17433	0.211845	0.266858	0.184223	0.187694	0.178201	0.249297	0.182406	0.136372	0.207285	0.204888	0.1639
18	0.287588	0.151442	0.156647	0.302397	0.225551	0.466812	0.478703	0.149691	0.226101	0.305505	0.234495	0.356869	0.19354	0.28012	0.1345
19	0.136541	0.179754	0.142437	0.152855	0.132957	0.19518	0.189542	0.125988	0.119616	0.179583	0.153277	0.117999	0.186649	0.216118	0.1124
20	0.297219	0.232881	0.167759	0.227405	0.21877	0.279138	0.210106	0.244218	0.181972	0.277604	0.26635	0.185301	0.23357	0.206812	0.1273
21	0.063688	0.110862	0.137799	0.081722	0.079446	0.080925	0.097078	0.342803	0.325466	0.114013	0.047403	0.034247	0.088028	0.058804	0.0878
22	0.061715	0.284366	0.131103	0.091634	0.098981	0.032525	0.042332	0.106299	0.126153	0.113638	0.089042	0.029867	0.094488	0.170946	0.0875
23	0.026295	0.084813	0.098917	0.086762	0.06326	0.025059	0.030061	0.056614	0.053751	0.075653	0.057585	0.037117	0.088066	0.127462	0.0816
24	0.065641	0.135384	0.160469	0.072195	0.122823	0.068512	0.069681	0.090852	0.105425	0.111514	0.048319	0.037818	0.098703	0.069265	0.0748
25	0.059797	0.321446	0.216121	0.09865	0.087911	0.166206	0.065511	0.219935	0.208812	0.143366	0.035947	0.036173	0.157352	0.082816	0.1192
26	0.047734	0.147362	0.160552	0.088593	0.112921	0.079606	0.08049	0.10534	0.10977	0.076907	0.071329	0.05294	0.098203	0.092553	0.0656

Figure 4.7. An example of cosine similarity table for 20Newsgroups dataset

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	0	1.423907	1.628681	1.557916	1.597	1.367075	1.383844	1.427704	1.461296	1.532147	1.563436	1.588745	1.63627	1.548208	1.71509
2	1.423907	0	1.466609	1.37932	1.433769	1.178304	1.256117	1.169161	1.251958	1.386849	1.430212	1.442169	1.442015	1.376732	1.56397
3	1.628681	1.466609	0	1.596108	1.623533	1.410555	1.449278	1.470411	1.487053	1.582826	1.622433	1.626968	1.665862	1.577898	1.73511
4	1.557916	1.37932	1.596108	0	1.563902	1.281891	1.393988	1.406882	1.424662	1.51855	1.532208	1.544325	1.58016	1.504365	1.6628
5	1.597	1.433769	1.623533	1.563902	0	1.380513	1.425752	1.444485	1.469518	1.550394	1.576931	1.526587	1.63822	1.546824	1.71192
6	1.367075	1.178304	1.410555	1.281891	1.380513	0	1.170678	1.196231	1.210285	1.327747	1.376153	1.350919	1.415772	1.324499	1.51485
7	1.383844	1.256117	1.449278	1.393988	1.425752	1.170678	0	1.266316	1.294831	1.38579	1.347296	1.408616	1.467284	1.382085	1.56501
8	1.427704	1.169161	1.470411	1.406882	1.444485	1.196231	1.266316	0	0.597132	1.396054	1.441521	1.441411	1.432204	1.391941	1.56834
9	1.461296	1.251958	1.487053	1.424662	1.469518	1.210285	1.294831	0.597132	0	1.421994	1.46661	1.456836	1.513818	1.417901	1.59162
10	1.532147	1.386849	1.582826	1.51855	1.550394	1.327747	1.38579	1.396054	1.421994	0	1.548238	1.542302	1.595785	1.501102	1.67137
11	1.563436	1.430212	1.622433	1.532208	1.576931	1.376153	1.347296	1.441521	1.46661	1.548238	0	1.592222	1.631812	1.535516	1.70492
12	1.588745	1.442169	1.626968	1.544325	1.526587	1.350319	1.408616	1.441411	1.456836	1.542302	1.592222	0	1.626208	1.50765	1.67883
13	1.63627	1.442015	1.665862	1.58016	1.63822	1.415772	1.467284	1.432204	1.513818	1.595785	1.631812	1.626208	0	1.498703	1.74365
14	1.548208	1.376732	1.577898	1.504365	1.546824	1.324499	1.382085	1.391941	1.417901	1.501102	1.535516	1.50765	1.498703	0	1.65484
15	1.715096	1.563978	1.735117	1.66283	1.711921	1.514852	1.565015	1.568345	1.591622	1.671372	1.704927	1.678831	1.743695	1.654842	
16	1.69705	1.544482	1.713251	1.649658	1.692793	1.494729	1.545139	1.553285	1.564959	1.63137	1.681897	1.694602	1.728293	1.649718	1.80485
17	1.633557	1.471672	1.656997	1.601194	1.629089	1.391923	1.474897	1.481959	1.506449	1.588206	1.616702	1.635928	1.669495	1.582616	1.74444
18	1.450671	1.282608	1.492387	1.387665	1.437493	1.124764	1.11289	1.293102	1.259471	1.410103	1.43737	1.382152	1.479911	1.38668	1.58647
19	1.678042	1.510474	1.703298	1.645424	1.676257	1.46743	1.507454	1.531909	1.555578	1.63492	1.671368	1.681254	1.706668	1.626001	1.78697
20	1.523741	1.34644	1.5939	1.52811	1.561814	1.342828	1.399708	1.348769	1.414361	1.505101	1.513448	1.558784	1.558813	1.512547	1.68286
21	1.543431	1.367805	1.571077	1.511694	1.545303	1.320276	1.359748	1.219696	1.258054	1.500859	1.522913	1.543851	1.587913	1.496555	1.66238
22	1.567632	1.344978	1.584613	1.532251	1.565238	1.348788	1.403784	1.39498	1.407965	1.521572	1.54381	1.571756	1.605381	1.511021	1.68322
23	1.602875	1.438126	1.627644	1.530696	1.599435	1.387205	1.44112	1.448542	1.473405	1.556804	1.595422	1.604568	1.639884	1.549489	1.71416
24	1.579582	1.412711	1.607573	1.551457	1.574833	1.365074	1.415031	1.429611	1.444748	1.539583	1.57148	1.574657	1.623262	1.533275	1.70035
25	1.509304	1.272472	1.530165	1.472065	1.507778	1.255842	1.338638	1.323346	1.351701	1.462058	1.504619	1.512144	1.549223	1.456437	1.62945

Figure 4.8. An example of Euclidean distance table for 20Newsgroups dataset

Then, the Euclidean distance is calculated; Figure 4.8 shows the Euclidean distance table. The total weight (initial light for each firefly) for each document also determines the initial position of each document (firefly) and the Cartesian distance between these positions. In Figure 4.9, the table includes the information on the total weight (initial light for each firefly) for each document.

	A	B
1	5.795003	
2	3.195146	
3	3.554897	
4	5.303493	
5	4.490492	
6	3.065165	
7	3.351498	
8	3.732925	
9	3.979747	
10	3.228945	
11	6.244328	
12	6.047511	
13	5.509208	
14	3.611637	
15	4.289905	
16	5.140618	
17	3.529675	
18	4.369665	
19	4.060873	
20	5.494527	
21	5.550248	
22	3.836508	
23	3.191373	
24	7.10446	
25	4.022661	
26	8.326935	

*Figure 4.9.* An example of total weight for 20Newsgroups dataset

The initial positioning of each document (firefly) is represented randomly between 1 and 30, which is later normalized between 0 and 1 as shown in Figures 4.10 and 4.11.



### Initial positioning between 1 and 30

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD
1	22	6	3	16	11	30	7	28	17	14	8	5	29	21	25	27	26	19	15	1	23	2	4	18	24	13	9	20	10	12

### Normalized positioning between 0 and 1

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	0.7241	0.1724	0.069	0.5172	0.3448	1	0.2069	0.931	0.5517	0.4483	0.2414	0.1379	0.9655	0.6897	0.8276	0.8966	0.8621	0.62069	0.4828	0	0.759	0.034	0.1034	0.5862	0.7931

Figure 4.10. An example of normalized initial positioning for 20Newsgroups dataset

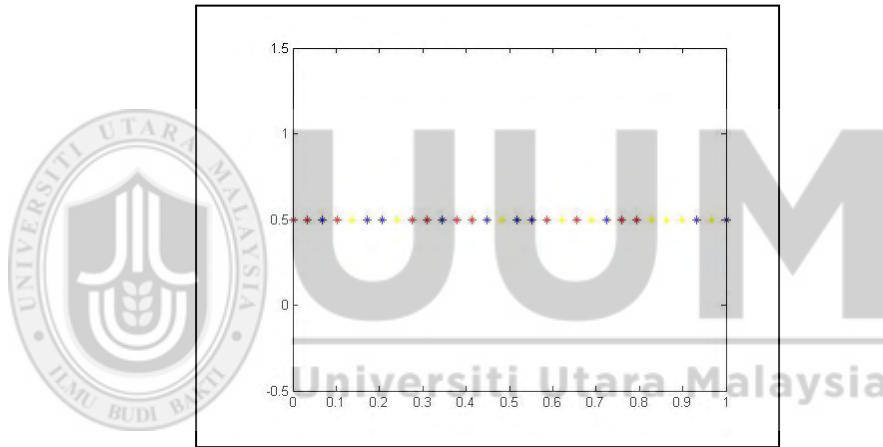


Figure 4.11. Graphical representation of initial document positioning for 20Newsgroups dataset

When WFA operates, document 1 will compete with documents 2 until 30. In this example, the light for document 1 is (5.7950) and it is lesser than the light for document 11 which is (6.2443). The cosine similarity between document 1 and document 11 is (0.2420) and this value exceeds the threshold (in this experiment, it is set to 0.15). Hence, document 1 will be moved towards document 11. The amount of attraction,  $\beta$ , is equal to (0.7921) and is obtained using Equation 2.8 (in Step 12 of the proposed WFA). The initial attraction  $\beta_0$  is 1 and the absorption coefficient  $\gamma$

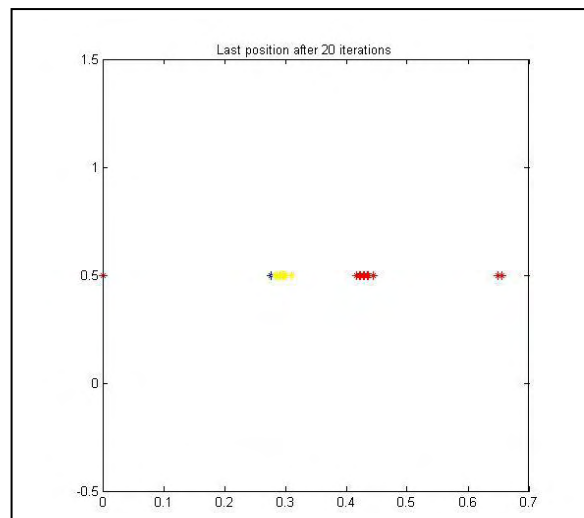
equals to 1. The distance between the position of document 1 and document 11 is equal to (0.4828). The position of document 1 is changed using Equation 2.7, where it becomes (0.3555). In addition, the light of document 11 increases based on Equation 4.8 and it becomes (7.0364).

After 20 iterations, the brightest document will be chosen as the centroid. In this example, document 10 has (348.49), which is the brightest light, and it becomes the centroid for the first cluster. Similar documents in the dataset are grouped with the centroid into this cluster. Cluster 1 later includes 21 documents including the center. The second center has document 25 as the center with seven other documents. The last cluster includes only one document which is the center. A graphical representation of the final position of the documents is illustrated in Figure 4.12.

Cluster1 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 30]

Cluster2 [21, 22, 23, 24, 25, 26, 27, 29]

Cluster3 [28]



*Figure 4.12.* An example of graphical representation of final document positioning for 20Newsgroups dataset

## 4.2 Evaluation

The proposed WFA is later evaluated to study the effectiveness of WFA in automatically producing clusters without prior information on the dataset. A comparison is later made against Particle Swarm Optimization (PSO) (Cui, Potok, & Palathingal, 2005), K-means (Jain, 2010), Hybrid Firefly algorithm with K-means (FAK-means) (Tang, Fong, Yang, & Deb, 2012) and Bisect K-means (Murugesan & Zhang, 2011a, 2011b), which they require a predefined  $k$  number of cluster. Tables 4.2 and 4.3 include the results of the five algorithms: WFA, PSO, K-means, FAK-means and Bisect K-means. Each algorithm was executed thirty times and the average values of the metrics were calculated. Figure 4.13 includes a graphical representation of the results.

As shown in Table 4.2, the WFA algorithm generates the highest average purity and smallest average Entropy and DBI in all iterations compared with PSO, K-means, FAK-means and Bisect K-means, while PSO produces a higher average F-measure, and K-means generates a smaller ADDC and the highest average DI. The purity results of five methods, namely WFA, PSO, K-means, FAK-means and Bisect K-means, are compared and represented graphically as shown in Figure 4.13.a. It is noticed that the purity of WFA is generated the highest value in all iterations, while K-means produced a smaller purity. Furthermore, it is noticed that Bisect K-means has a higher purity than K-means, FAK-means and PSO. As learned from the literature, (Forsati, Mahdavi, Shamsfard, & Meybodi, 2013; Luo, Li, & Chung, 2009; Murugesan & Zhang, 2011a, 2011b), a higher value of purity (approaching to 1) indicates that it is a better clustering solution.

Table 4.2

*External quality metrics of clustering: WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means.*

External Metrics	Algorithms	Iterations				
		1	2	5	10	20
Purity	WFA	<b>0.6150</b> (0.0335)	<b>0.5928</b> (0.0161)	<b>0.5898</b> (0.0258)	<b>0.5934</b> (0.0186)	<b>0.5230</b> (0.0137)
	PSO	0.3823 (0.0432)	0.3861 (0.0643)	0.3726 (0.0433)	0.3731 (0.0403)	0.3772 (0.0516)
	K-means	0.3468 (0.0242)	0.3451 (0.0226)	0.3531 (0.0414)	0.3561 (0.0575)	0.3459 (0.0258)
	FAK-means	0.3658 (0.0133)	0.3701 (0.0150)	0.3738 (0.0138)	0.3719 (0.0147)	0.3731 (0.0132)
	Bisect K-means	0.3759 (0.0300)	0.3806 (0.0397)	0.3872 (0.0446)	0.3939 (0.0601)	0.4031 (0.0818)
External Metrics	Algorithms	Iterations				
		1	2	5	10	20
F-measure	WFA	0.4533 (0.0220)	0.4499 (0.0053)	0.4495 (0.0053)	0.4488 (0.0038)	0.4639 (0.0025)
	PSO	<b>0.4947</b> (0.0188)	<b>0.5062</b> (0.0414)	0.4907 (0.0118)	0.4951 (0.0110)	<b>0.4986</b> (0.0237)
	K-means	0.4910 (0.0213)	0.4935 (0.0138)	<b>0.4975</b> (0.0196)	<b>0.4999</b> (0.0316)	0.4954 (0.0081)
	FAK-means	0.3656 (0.0134)	0.3692 (0.0140)	0.3747 (0.0155)	0.3723 (0.0143)	0.3737 (0.0132)
	Bisect K-means	0.4698 (0.0297)	0.4757 (0.0284)	0.4775 (0.0336)	0.4785 (0.0567)	0.4908 (0.0644)
External Metrics	Algorithms	Iterations				
		1	2	5	10	20
Entropy	WFA	<b>1.1275</b> (0.0724)	<b>1.1966</b> (0.0215)	<b>1.1989</b> (0.0194)	<b>1.1964</b> (0.0137)	<b>1.2328</b> (0.0237)
	PSO	1.5350 (0.0393)	1.5230 (0.0858)	1.5403 (0.0513)	1.5309 (0.0619)	1.5276 (0.0691)
	K-means	1.5782 (0.013)	1.5794 (0.0100)	1.5722 (0.0341)	1.5636 (0.072)	1.5755 (0.0215)
	FAK-means	1.5782 (0.0042)	1.5770 (0.0059)	1.5751 (0.0054)	1.5750 (0.0067)	1.5754 (0.0056)
	Bisect K-means	1.5620 (0.0223)	1.5571 (0.0285)	1.5466 (0.0914)	1.5325 (0.0841)	1.5150 (0.1193)

*Note: the best value is highlighted in 'bold', standard deviation in ().*

Table 4.3

*Internal and relative quality metrics of clustering: WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means.*

Internal & Relative Metrics	Algorithms	Iterations				
		1	2	5	10	20
ADDC	WFA	0.8063 (0.0481)	0.7819 (0.0469)	0.8289 (0.0514)	0.8159 (0.0348)	<b>0.6694</b> (0.0884)
	PSO	1.7630 (0.2124)	1.7775 (0.2738)	1.8053 (0.2751)	1.9379 (0.1862)	1.8736 (0.0430)
	K-means	<b>0.7547</b> (0.2978)	<b>0.7041</b> (0.2861)	<b>0.6783</b> (0.2396)	<b>0.6367</b> (0.2511)	0.7056 (0.3190)
	FAK-means	1.4434 (0.0010)	1.4436 (0.0007)	1.4436 (0.0009)	1.4432 (0.0009)	1.4436 (0.0007)
	Bisect K-means	1.3238 (0.1874)	1.2545 (0.2808)	1.2633 (0.2378)	1.3332 (0.2318)	1.3494 (0.1817)
Internal & Relative Metrics	Algorithms	Iterations				
		1	2	5	10	20
DBI	WFA	<b>1.3452</b> (0.0217)	<b>1.3369</b> (0.0235)	<b>1.3631</b> (0.0358)	<b>1.3527</b> (0.0222)	<b>1.3249</b> (0.0460)
	PSO	1.7069 (0.0261)	1.6100 (0.2168)	1.5559 (0.2303)	1.6092 (0.1732)	1.6472 (0.2269)
	K-means	2.8159 (3.5419)	2.3565 (3.1847)	2.4741 (3.1267)	1.9649 (3.1855)	1.9090 (2.5369)
	FAK-means	14.2277 (0.2063)	14.2834 (0.2025)	14.2549 (0.3551)	14.2637 (0.2862)	14.2158 (0.2801)
	Bisect K-means	8.1636 (3.0869)	7.3146 (3.0355)	6.9485 (3.0605)	7.8287 (2.8864)	7.7189 (2.7763)
Internal & Relative Metrics	Algorithms	Iterations				
		1	2	5	10	20
DI	WFA	0.9312 (0.0195)	0.9236 (0.0129)	0.9208 (0.0185)	0.9273 (0.0074)	0.9040 (0.0364)
	PSO	1.0162 (0.0950)	1.0331 (0.0574)	1.0357 (0.0650)	1.0119 (0.0583)	0.9908 (0.0899)
	K-means	<b>2.3413</b> (2.3505)	<b>2.9315</b> (2.4566)	<b>2.9671</b> (2.5302)	<b>3.7363</b> (2.3653)	<b>3.4078</b> (2.4518)
	FAK-means	0.1380 (0.0026)	0.1372 (0.0027)	0.1377 (0.0039)	0.1374 (0.0035)	0.1382 (0.0035)
	Bisect K-means	0.2393 (0.1483)	0.2876 (0.2299)	0.2998 (0.1887)	0.2715 (0.2178)	0.2698 (0.2008)

*Note: the best value is highlighted in 'bold', standard deviation in ().*

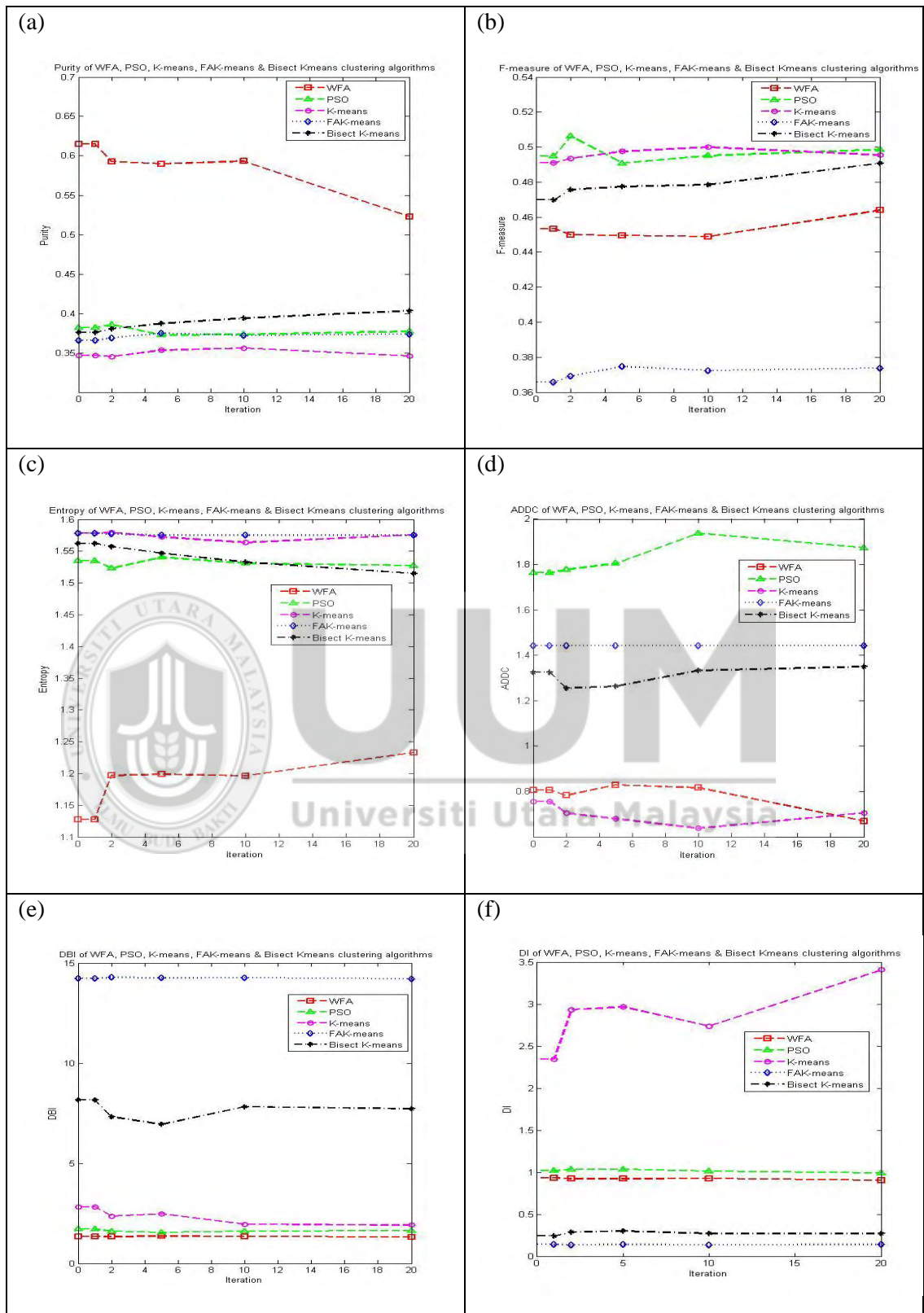


Figure 4.13. Graphical representation of quality metrics of WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means; a) Purity, b) F-measure, c) Entropy, d) ADDC, e) DBI, and f) DI.

Standard Deviation measures how much variation or dispersion from the average exists. A small value of standard deviation indicates that the documents tend to be very close to the mean (center). Based on the observation, the overall standard deviation of WFA algorithm is smaller than PSO, K-means, Bisect K-means in most iterations (refer to iterations 2, 5, 10 and 20) which indicates it is more reliable and more robust.

Table 4.2 also includes the average F-measure of thirty executions for each WFA, PSO, K-means, FAK-means and Bisect K-means. F-measure tries to capture how well the groups of the investigated partition best match the groups of the reference. A high F-measure (near to 1) means perfect clustering (Forsati, Mahdavi, Shamsfard, & Meybodi, 2013; Luo, Li, & Chung, 2009; Murugesan & Zhang, 2011a, 2011b). F-measure is based on two important metrics, precision and recall, which are widely used in information retrieval for evaluation. Precision measures the accuracy of a cluster that produces a specific class, while recall measures the completeness of a specific class. As shown in Table 4.2, the PSO algorithm generates the highest average F-measure in iterations 1, 2 and 20 compared to all the algorithms, while K-means generates the highest average F-measure in iterations 5 and 10. The F-measure of the WFA algorithm is only better than FAK-means in all iterations. In addition, the overall standard deviation of the WFA algorithm is smaller than other algorithms (refer to iterations 2, 5, 10 and 20) which indicates it is more reliable and more robust. Figure 4.13.b presents the F-measure result of five algorithms: WFA, PSO, K-means, FAK-means, and Bisect K-means.

Table 4.2 involves the average Entropy of thirty executions for each WFA, PSO, K-means, FAK-means, and Bisect K-means. Entropy measures the distribution of various classes in each cluster. The smallest Entropy (near to 0) indicates a better clustering solution (Forsati, Mahdavi, Shamsfard, & Meybodi, 2013; Murugesan & Zhang, 2011a, 2011b). As shown in Table 4.2, the average Entropy that is produced by WFA is the smallest than other algorithms in all iterations. On the other hand, PSO produces a smaller Entropy compared to FAK-means, K-means, and Bisect K-means in most iterations. Furthermore, the standard deviation of the WFA algorithm is smaller than PSO, K-means, and Bisect K-means in most iterations (refer to iterations 5 and 10) which indicates it is more reliable and more robust. This result implies that WFA is best to produce clusters with single class than others. Figure 4.13.c illustrates the result of the average Entropy in a graphical representation of five methods: WFA, PSO, K-means, FAK-means and Bisect K-means.

Table 4.3 includes the quality performance results of internal metrics, which are ADDC, DBI and DI for five algorithms: WFA, PSO, K-means, FAK-means and Bisect K-means. All algorithms are implemented in the same environment and are run thirty times for different iterations. The average values of ADDC, DBI and DI are calculated. As shown in Table 4.3, K-means generates a smaller average ADDC in most iterations (refer to iterations 1, 2, 5 and 10) compared against WFA, K-means, FAK-means, and Bisect K-means, while WFA produces a smaller average ADDC in iteration 20 against other methods. Furthermore, the standard deviation of FAK-means is the smallest value against others, followed by the proposed WFA algorithm which is better than PSO, K-means, and Bisect K-means. This result



indicates that WFA obtains more compact clusters. The ADDC values of the five techniques, namely WFA, PSO, K-means, FAK-means, and Bisect K-means, are illustrated in Figure 4.13.d.

In Table 4.3, the average DBI of thirty executions for each WFA, PSO, K-means, FAK-means, and Bisect K-means are reported. DBI measures the validation of how well the output clusters are done (minimum intra distance of clusters and maximum inter distance) using quantities and features inherent to the dataset. Whereby, the summations of the maximum ratio between the average distances of two clusters (not similar) over the distance between the centers of the same two clusters are calculated for all of the produced clusters. The smallest DBI means a more compact clustering solution (Das, Abraham, & Konar, 2009). As shown in Table 4.3, WFA generates the smallest average DBI in all iterations. Furthermore, the standard deviation of WFA is the smallest in all runs, which shows that the solutions are more reliable. Figure 4.13.e illustrates the result of the average DBI in a graphical representation.

Table 4.3 also involves the average DI of thirty executions for each WFA, PSO, K-means, FAK-means, and Bisect K-means. DI measures the ratio of the smallest distance between observations not in the same cluster to the largest intra cluster distance. DI has a value between 0 and  $\infty$  and the largest value of DI means a more compact clustering solution (Das, Abraham, & Konar, 2009). As shown in Table 4.3, WFA generates the largest average of DI against FAK-means and Bisect K-means in all iterations, while K-means produces the best DI value against all other methods, followed by PSO. In addition, it is noticed that the value of standard deviation in WFA is smaller than PSO, K-means, and Bisect K-means in all iterations, which

shows that the solutions in WFA are more reliable. Figure 4.13.f illustrates the result of the average DI in a graphical representation.

Table 4.4 displays the average number of clusters that are automatically generated by WFA and without any prior knowledge about the dataset. As seen in Table 4.4, the number of clusters produced by WFA is higher than the number of clusters in other methods.

Table 4.4

*Average number of clusters of WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means.*

Iterations	Number of clusters of algorithms				Bisect K-means
	WFA	PSO	K-means	FAK-means	
1	19.30 $\approx$ 19	3	3	3	3
2	18.73 $\approx$ 19	3	3	3	3
5	17.57 $\approx$ 18	3	3	3	3
10	17.80 $\approx$ 18	3	3	3	3
20	14.80 $\approx$ 15	3	3	3	3

The obtained results indicate that WFA generates better clusters as it is the best in two of the external metrics, purity and entropy; meanwhile in internal metrics DBI, it obtained the best value. Table 4.5 illustrates the results of different quality performance metrics in five algorithms.

Table 4.5

*Results of quality performance of WFA vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means.*

Algorithms	External Metrics			Internal Metrics		
	Purity	F-measure	Entropy	ADDC	DBI	DI
WFA	✓		✓		✓	
PSO		✓				
K-means				✓		✓
FAK-means						
Bisect K-means						

In the upcoming chapters, the performance of WFA will be improved by integrating the algorithm with document re-locating algorithm and merging algorithm.

### 4.3 Summary

This chapter presents text clustering using one of the swarm intelligence algorithms, known as Weight-based Firefly Algorithm (WFA). Experiments were conducted on the 20Newsgroups dataset and a comparison of the proposed WFA was made against the other algorithms (PSO, K-means, FAK-means, and Bisect K-means).

It is learned that WFA generates better results in Purity, Entropy and DBI metrics. WFA has not only generated a better reading in the measures, but it also does not rely on a pre-determined k number of clusters. Such a result indicates the rival of WFA in text clustering. Furthermore, the result was obtained without initial information on the dataset.

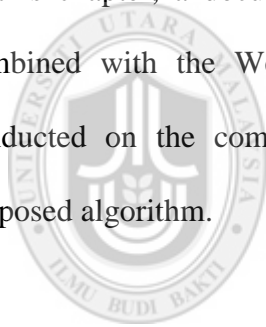
## **CHAPTER FIVE**

### **CLUSTER REFINING**

#### **5.1 Introduction**

The construction of clusters in the proposed WFA (Chapter Four) is based on a static threshold, where documents that are similar to the centroid are assigned to the first cluster. However, there is a possibility that these documents are more similar to one of the upcoming centroids. Such a situation will lead to poor cluster purity. Hence, it is proposed that WFA allows the re-location of an assigned document.

In this chapter, a document re-locating algorithm is introduced. Later, it will be combined with the Weight-based Firefly Algorithm. Empirical studies will be conducted on the combinations to measure the performance evaluation of the proposed algorithm.



**UUM**  
Universiti Utara Malaysia

#### **5.2 Document Re-locating**

Documents re-locating can be introduced when every new cluster in Weight-based Firefly Algorithm (WFA) is constructed starting from the second cluster. It operates by identifying the similarity between the newly identified centroid (the center of the new cluster) and documents that have been assigned in the previous clusters. If the similarity is higher, then the document is moved from the original cluster to the newly created cluster. The pseudo code of documents re-locating is shown in Figure 5.1.

### **Document Re-locating**

**Step 1:** Initial  $m$ =number of clusters.

**Step 2:** If  $m \geq 2$

**Step 3:** For  $K=1$  to  $(m-1)$

**Step 4:** If length (current cluster (K)) $>1$

**Step 5:** For  $Z=1$  to length (current cluster (K))

**Step 6:** If current document (Z) not equal to center (K)

**Step 7:** If similarity(center(m), current document (Z))  $>$  similarity(center(K), current document (Z))

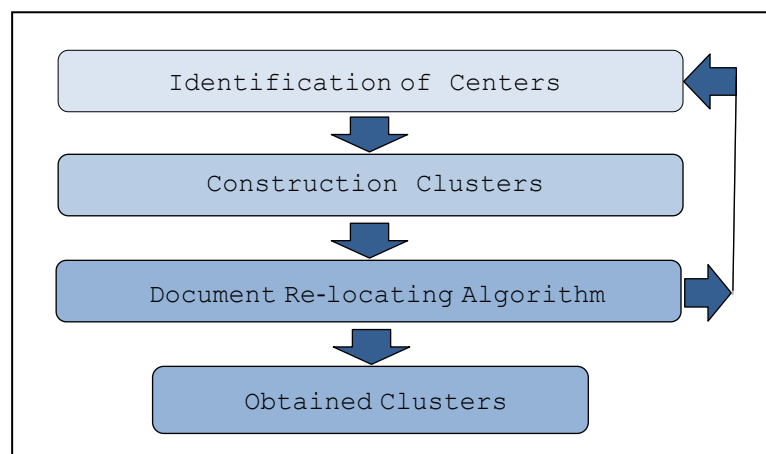
**Step 8:** Move (Z) from current cluster to recent cluster

**Step 9:** End for Z

**Step 10:** End for K

*Figure 5.1.* The pseudo code of Document Re-locating

In a further attempt to improve the solutions produced by WFA, another variant of FA that is termed as Weight-based Firefly Algorithm with Relocate (WFA<sub>R</sub>) is proposed. WFA<sub>R</sub> includes the relocating mechanism as illustrated in Figure 5.2.



*Figure 5.2.* The process of WFA<sub>R</sub>

The steps of WFA<sub>R</sub> are shown in Figure 5.3.

**Weight-based Firefly Algorithm with Relocate (WFA<sub>R</sub>)**

- Step 1:** Generate initial population of firefly randomly  $x^i$  where  $i=1, 2 \dots n$ ,  $n$ =number of fireflies (documents).
- Step 2:** Initial Light Intensity,  $I$ =total weight of document using Equations 4.1 and 4.2.
- Step 3:** Define light absorption coefficient  $\gamma$ , initial  $\gamma=1$ .
- Step 4:** Define the randomization parameter  $\alpha$ ,  $\alpha=0.2$ .
- Step 5:** Define initial attractiveness  $\beta_0 = 1.0$ .
- Step 6:** While  $t < \text{Maximum number of iteration}$  ( $t = \text{number of iteration}$ )
- Step 7:** For  $i=1$  to  $N$
- Step 8:** For  $j=1$  to  $N$
- Step 9:** IF ( $\text{Light } I_i < \text{Light } I_j$ ) ( $\text{Light}=\text{total weight}$ )
- Step 10:** IF ( $\text{CosineSimilarity}(i, j) \geq \text{Threshold}$ ) ( $\text{CosineSimilarity}$  using Equation 4.10)
- Step 11:** Calculate distance between  $i, j$  using Equation 4.4.
- Step 12:** Calculate attractiveness using Equation 2.8.
- Step 13:** Calculate random parameter  $u$  using Equations 4.5, 4.6 and 4.7.
- Step 14:** Move document  $i$  to  $j$  using Equation 2.7.
- Step 15:** Update light intensity using Equation 4.8.
- Step 16:** End For  $j$
- Step 17:** End For  $i$
- Step 18:**  $t=t+1$
- Step 19:** End While
- Step 20:** Rank the Light List to find best document (brightest light) and represent as center.
- Step 21:** Find document similar to center using Equation 4.10 and construct cluster
- Step 22:** Remove produced clusters from Light List.

Figure 5.3. Steps of the WFA<sub>R</sub> algorithm

Figure 5.3 continued

**Step 23:** Initial  $m$ =number of clusters.  
**Step 24:** If  $m \geq 2$   
**Step 25:** For  $K=1$  to  $(m-1)$   
**Step 26:** If length (current cluster (K))>1  
**Step 27:** For  $Z=1$  to length (current cluster (K))  
**Step 28:** If current document (Z) not equal center (K)  
**Step 29:** If similarity(center(m), current document(Z)) > similarity(center(K), current document (Z))  
**Step 30:** Move (Z) from current cluster to recent cluster  
**Step 31:** End for Z  
**Step 32:** End for K  
**Step 33:** Return to Step 20 until remains one document in Light List.  
**Step 34:** Output clusters.

#### Example

The process involved in the proposed WFA<sub>R</sub> is explained using the same example presented in Chapter Four. Upon completing the construction of the second cluster by WFA (in the previous example), then only, the process of relocating documents will be started. Initially, WFA produced the following clusters:

Cluster1 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 30]

Cluster2 [21, 22, 23, 24, 25, 26, 27, 29]

The document re-locating algorithm compares the center of the second cluster (cluster 2) with all documents in cluster 1. If the cosine similarity between the center of cluster 2 and the document in cluster 1 is greater than the cosine similarity between the center of cluster 1 and the document in cluster 1, then this document is

assigned to cluster 2 and is removed from cluster 1. Upon checking all documents, two documents D2 and D30 are assigned to cluster 2 and removed from cluster 1. The newly produced clusters are as shown below.

Cluster1 [1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

Cluster2 [21, 22, 23, 24, 25, 26, 27, 29, 2, 30]

Then,  $WFA_R$  constructs cluster 3, where there exists only a single document in cluster 3 (i.e. 28). Later, the last cluster (i.e. cluster 3) is compared against documents in cluster 1 and documents in cluster 2. Since there is no change, the clusters remain the same.

### 5.3 Evaluation

This section includes two evaluations; the first evaluation is on the comparison between  $WFA_R$  and WFA, and the second between  $WFA_R$  and state-of-the-art methods.

#### 5.3.1 Comparison between $WFA_R$ and WFA

For comparison purposes, the evaluation on Purity, F-measure and Entropy is presented in Table 5.1, while Table 5.2 includes the results on ADDC, DBI and DI in the form of average, best, worst, and standard deviation of the two methods ( $WFA_R$  and WFA). Figure 5.4 includes the graphical representation of the obtained results.



Table 5.1

*External quality metrics: WFA vs. WFA<sub>R</sub>.*

Iterations	Algorithms	Purity Metric			
		Average Purity	Best Purity	Worst Purity	Standard Deviation
1	WFA	0.6150	<b>0.7500</b>	0.5467	0.0335
	WFA <sub>R</sub>	<b>0.8300</b>	<b>0.9467</b>	0.7967	<b>0.0255</b>
2	WFA	0.5928	0.6233	0.5467	0.0161
	WFA <sub>R</sub>	<b>0.8258</b>	0.8700	0.8100	<b>0.0095</b>
5	WFA	0.5898	0.5966	0.4933	0.0258
	WFA <sub>R</sub>	<b>0.8177</b>	0.8233	0.7900	<b>0.0058</b>
10	WFA	0.5934	0.6033	0.4933	0.0186
	WFA <sub>R</sub>	<b>0.8182</b>	0.8267	0.8100	<b>0.0036</b>
20	WFA	0.5230	0.5300	0.4933	0.0137
	WFA <sub>R</sub>	<b>0.7868</b>	0.8100	0.7433	<b>0.0129</b>

Iterations	Algorithms	F-measure Metric			
		Average F-measure	Best F-measure	Worst F-measure	Standard Deviation
1	WFA	0.4533	<b>0.5693</b>	0.4481	<b>0.0220</b>
	WFA <sub>R</sub>	<b>0.5199</b>	0.6625	0.4930	0.0339
2	WFA	0.4499	0.4659	0.4481	<b>0.0053</b>
	WFA <sub>R</sub>	<b>0.5145</b>	0.5423	0.4983	0.0126
5	WFA	0.4495	0.4694	0.4481	<b>0.0053</b>
	WFA <sub>R</sub>	<b>0.5194</b>	0.5657	0.5080	0.0174
10	WFA	0.4488	0.4694	0.4481	<b>0.0038</b>
	WFA <sub>R</sub>	<b>0.5295</b>	<b>0.6589</b>	0.5034	0.0365
20	WFA	0.4639	0.4694	0.4628	<b>0.0025</b>
	WFA <sub>R</sub>	<b>0.6186</b>	<b>0.6589</b>	0.5551	0.0426

Iterations	Algorithms	Entropy Metric			
		Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
1	WFA	1.1275	<b>0.8906</b>	1.2382	<b>0.0724</b>
	WFA <sub>R</sub>	<b>0.6676</b>	<b>0.2806</b>	0.7250	0.0830
2	WFA	1.1966	1.1042	1.2382	<b>0.0215</b>
	WFA <sub>R</sub>	<b>0.6921</b>	0.5299	0.7396	0.0345
5	WFA	1.1989	1.1830	1.2727	0.0194
	WFA <sub>R</sub>	<b>0.7282</b>	0.7008	0.7453	<b>0.0135</b>
10	WFA	1.1964	1.1855	1.2686	<b>0.0137</b>
	WFA <sub>R</sub>	<b>0.7315</b>	0.6587	0.7453	0.0206
20	WFA	1.2328	1.2195	1.2828	<b>0.0237</b>
	WFA <sub>R</sub>	<b>0.7244</b>	0.6587	0.8381	0.0328

*Note: the best value is highlighted in 'bold'.*

Table 5.2

*Internal and relative quality metrics: WFA vs. WFA<sub>R</sub>.*

Iterations	Algorithms	ADDC Metric			
		Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
1	WFA	<b>0.8063</b>	0.6992	0.9629	0.0481
	WFA <sub>R</sub>	1.4282	1.3688	1.4557	<b>0.0233</b>
2	WFA	<b>0.7819</b>	0.7047	0.9232	0.0469
	WFA <sub>R</sub>	1.4228	1.3640	1.4557	<b>0.0156</b>
5	WFA	<b>0.8289</b>	0.7373	0.9232	0.0514
	WFA <sub>R</sub>	1.4171	1.3683	1.4413	<b>0.0130</b>
10	WFA	<b>0.8159</b>	0.7373	0.8578	<b>0.0348</b>
	WFA <sub>R</sub>	1.4085	1.1994	1.4488	0.0408
20	WFA	<b>0.6694</b>	<b>0.6152</b>	0.9122	0.0884
	WFA <sub>R</sub>	1.2222	<b>1.1569</b>	1.3949	<b>0.0744</b>

Iterations	Algorithms	DBI Metric			
		Average DBI	Best DBI	Worst DBI	Standard Deviation
1	WFA	<b>1.3452</b>	<b>1.2832</b>	1.4143	0.0217
	WFA <sub>R</sub>	1.6585	1.6313	1.6740	<b>0.0117</b>
2	WFA	<b>1.3369</b>	1.3198	1.4317	0.0235
	WFA <sub>R</sub>	1.6562	1.6386	1.6828	<b>0.0092</b>
5	WFA	<b>1.3631</b>	1.3079	1.4317	0.0358
	WFA <sub>R</sub>	1.6514	1.6386	1.7012	<b>0.0128</b>
10	WFA	<b>1.3527</b>	1.3079	1.4157	0.0222
	WFA <sub>R</sub>	1.6508	1.5967	1.7104	<b>0.0153</b>
20	WFA	<b>1.3249</b>	<b>1.2971</b>	1.4209	0.0460
	WFA <sub>R</sub>	1.6176	<b>1.5744</b>	1.7104	<b>0.0427</b>

Iterations	Algorithms	DI Metric			
		Average DI	Best DI	Worst DI	Standard Deviation
1	WFA	<b>0.9312</b>	<b>0.9602</b>	0.8582	<b>0.0195</b>
	WFA <sub>R</sub>	0.9017	<b>0.9415</b>	0.8522	0.0290
2	WFA	<b>0.9236</b>	0.9358	0.8828	<b>0.0129</b>
	WFA <sub>R</sub>	0.9167	0.9287	0.8522	0.0215
5	WFA	<b>0.9208</b>	0.9510	0.8828	0.0185
	WFA <sub>R</sub>	0.9064	0.9287	0.8808	<b>0.0170</b>
10	WFA	<b>0.9273</b>	0.9289	0.8879	<b>0.0074</b>
	WFA <sub>R</sub>	0.8946	0.9016	0.8495	0.0118
20	WFA	0.9040	0.9510	0.8490	0.0364
	WFA <sub>R</sub>	<b>0.9156</b>	0.9264	0.8495	<b>0.0202</b>

*Note: the best value is highlighted in 'bold'.*

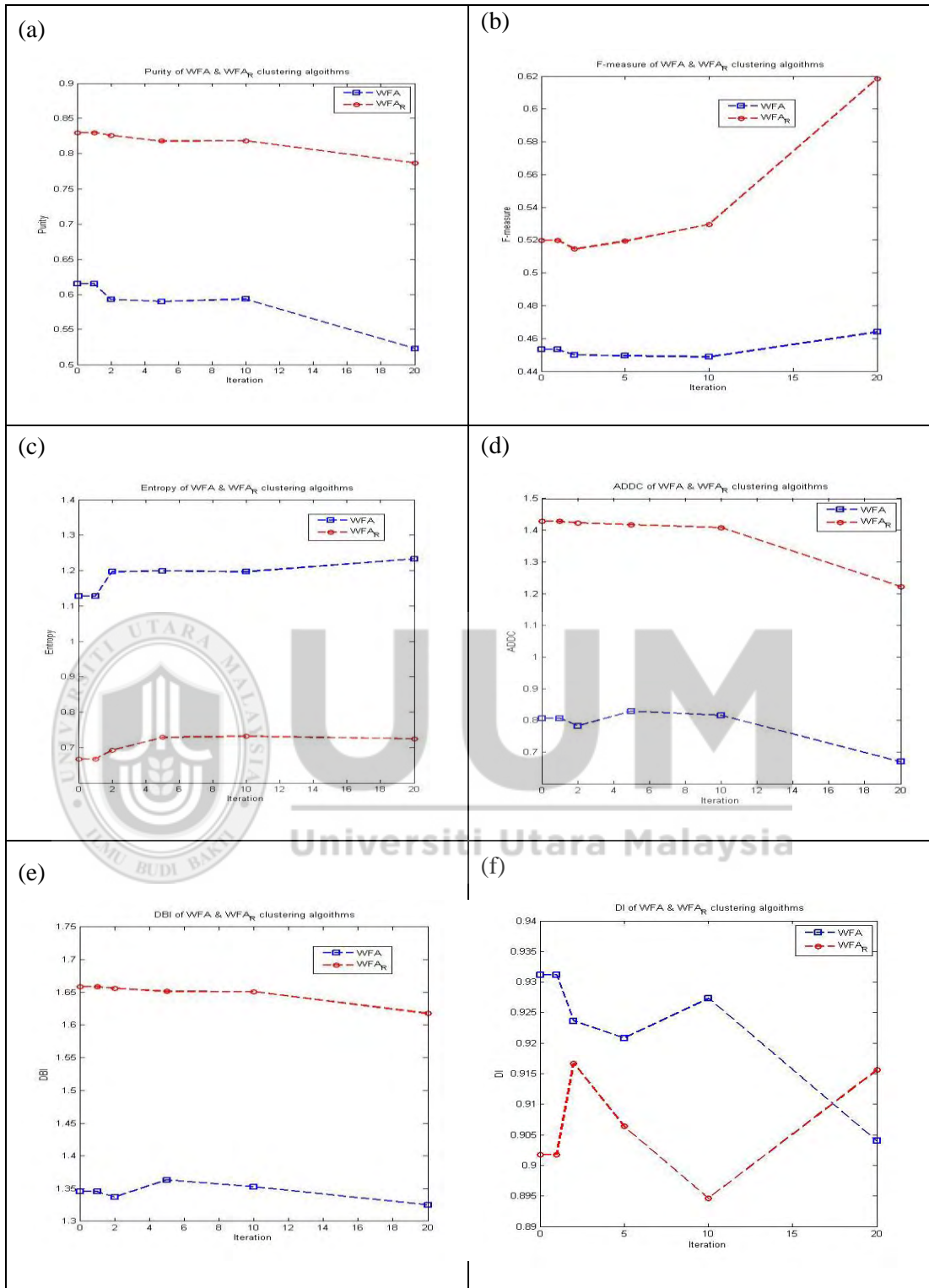


Figure 5.4. Graphical representation of quality metrics between WFA & WFA<sub>R</sub>; a) Purity, b) F-measure, c) Entropy, d) ADDC, e) DBI, and f) DI

As shown in Table 5.1, the WFA<sub>R</sub> algorithm generates the highest average purity in all iterations compared to the WFA algorithm. The best purity value is 0.9467 which is generated by WFA<sub>R</sub> in iteration 1, while WFA generates 0.7500 in the same iteration. In addition, the overall standard deviation of the WFA<sub>R</sub> algorithm is smaller than WFA in all iterations, which indicates it is more reliable and high in robustness. From this result, it can be concluded that the document re-locating algorithm has an impact on clustering. WFA<sub>R</sub> is better than WFA in producing pure subsets of documents. Figure 5.4.a illustrates the result of purity in a graphical representation of these methods. It shows that the purity curve of WFA<sub>R</sub> is higher than the curve of WFA in all iterations.

Table 5.1 illustrates the average F-measure results of two different algorithms, WFA and WFA<sub>R</sub>. As shown in Table 5.1, the WFA<sub>R</sub> algorithm generates a higher average F-measure in all iterations. Furthermore, it is observed that the best F-measure produced is 0.6589 in iterations 10 and 20 by WFA<sub>R</sub>, while WFA generates the best F-measure value is 0.5693 in iteration 1. The previous observations mean that the document re-locating algorithm affects the result of F-measure in WFA, despite the overall standard deviation of WFA is smaller than WFA<sub>R</sub> in all iterations, which indicates it is more reliable and high in robustness. Figure 5.4.b demonstrates the result of average F-measure in a pictorial representation of WFA and WFA<sub>R</sub>. It shows the F-measure curve of WFA<sub>R</sub> increases when the iteration increases, and rises in all iterations compared to the curve of WFA.

The average Entropy of thirty executions for each WFA and WFA<sub>R</sub> algorithm which includes the best, worst Entropy and standard deviation is reported in Table 5.1. As

shown in the table, the average Entropy of WFA<sub>R</sub> is smaller than WFA in all iterations. Additionally, the best Entropy value is 0.2806 which is generated by WFA<sub>R</sub> in iteration 1, while WFA generates 0.8906, better Entropy in the same iteration. Furthermore, the standard deviation of WFA<sub>R</sub> is smaller than WFA only in iteration 5, while WFA is the smaller in the remaining iterations that means the solutions are nearer to the average Entropy and contains less abnormal solutions. This result indicates that WFA<sub>R</sub> is best for producing more pure clusters. Figure 5.4.c shows the result of average Entropy in a graphical representation of WFA and WFA<sub>R</sub>.

Table 5.2 includes the average ADDC of thirty executions for each WFA and WFA<sub>R</sub> algorithm and also includes the best ADDC, worst ADDC and standard deviation. As shown in Table 5.2, the average ADDC that is produced by WFA is smaller than WFA<sub>R</sub> in all iterations. The best ADDC value is 0.6152 which is generated by WFA in iteration 20, while WFA<sub>R</sub> generates 1.1569 in the same iteration. Based on previous observations, WFA outperforms WFA<sub>R</sub>, regardless of the overall standard deviation of WFA<sub>R</sub> that is smaller than WFA, which means the solutions that are generated by WFA<sub>R</sub> are more robust and do not contain abnormal solutions. The ADDC result of WFA and WFA<sub>R</sub> is illustrated in a graphical representation in Figure 5.4.d. It shows that the ADDC of WFA has the smallest value 0.6152 in iteration 20, while WFA<sub>R</sub> is 1.1569 in the same iteration. From this result, it can be concluded that the document re-locating algorithm does not affect the ADDC result in WFA.

Table 5.2 includes the average DBI of thirty executions for each WFA and WFA<sub>R</sub> algorithm and also includes the best, worst DBI and standard deviation. As shown in

Table 5.2, WFA generates the smallest average DBI in all iterations compared against WFA<sub>R</sub>. In spite of the standard deviation of WFA<sub>R</sub> that is the smallest in most runs, this shows that the solutions are more reliable. The best DBI values are 1.2832 and 1.2971 that are generated by WFA in iterations 1 and 20, while WFA<sub>R</sub> generates 1.5744 in iteration 20. From previous results, it can be concluded that the document re-locating algorithm does not make any changes on the DBI value on WFA. Figure 5.4.e illustrates the result of the average DBI in a graphical representation between WFA and WFA<sub>R</sub>.

Table 5.2 reports the average DI of thirty executions for each WFA and WFA<sub>R</sub> algorithm and also includes the best, worst DI and standard deviation. DI measures the ratio of the smallest distance between observations not in same cluster to the largest intra cluster distance. DI has a value between 0 and  $\infty$  and the largest value of DI means a more compact clustering solution. Examining Table 5.2, it is clear that WFA generated the largest average DI in most iterations (refer to iterations 1, 2, 5 and 10), while WFA<sub>R</sub> produced the largest average DI on the last iteration 20. In addition, it is noticed that the value of standard deviation in WFA is the smallest in some iterations (refer to iterations 1, 2 and 10) and larger in the remaining iterations (5 and 20). The best DI value is 0.9602 that is generated by WFA in iteration 1, while WFA<sub>R</sub> generates 0.9415 in the same iteration. These results demonstrate that the document re-locating mechanism enhances the DI metrics in WFA only in the last iteration 20. Figure 5.4.f illustrates the result of the average DI in a graphical representation. It presents that the curve of DI is highest in WFA than WFA<sub>R</sub>, but it falls only in iteration 20 where WFA<sub>R</sub> increases in this iteration.

For an easy comparison between the number of clusters from WFA and WFA<sub>R</sub>, in Table 5.3, it reports the average number of clusters produced by the WFA and WFA<sub>R</sub> algorithms. As seen in Table 5.3, the number of clusters of the two algorithms is equal. It can be concluded that document re-locating mechanism did not affect the number of produced clusters in WFA.

Table 5.3

*Average number of clusters: WFA vs. WFA<sub>R</sub>.*

Iterations	Number of clusters of algorithms	
	WFA	WFA <sub>R</sub>
1	19.30 $\approx$ 19	19.80 $\approx$ 20
2	18.73 $\approx$ 19	19.06 $\approx$ 19
5	17.57 $\approx$ 18	18.03 $\approx$ 18
10	17.80 $\approx$ 18	17.63 $\approx$ 18
20	14.80 $\approx$ 15	14.36 $\approx$ 14

Table 5.4 presents the summary of comparison between WFA and WFA<sub>R</sub>.

Table 5.4

*Summary of quality performance: WFA vs. WFA<sub>R</sub>.*

Performance Metrics	Algorithms	
	WFA	WFA <sub>R</sub>
Purity		✓
F-measure		✓
Entropy		✓
ADDC	✓	
DBI	✓	
DI	✓	
Number of clusters	✓	✓

As depicted in Table 5.4, the document re-locating mechanism improves the results of external metrics: Purity, F-measure and Entropy. However, the mechanism does not affect the number of obtained clusters which are still large and need to be improved.

### 5.3.2 Comparison between WFA<sub>R</sub> and Other Methods

From the previous sections, it is learned that the results of external metrics of the Weight-based Firefly Algorithm with Relocate (WFA<sub>R</sub>) is better than the WFA algorithm. This section includes the comparison between WFA<sub>R</sub> and the selected state-of-the-art methods: Particle Swarm Optimization (PSO) (Cui, Potok, & Palathingal, 2005), K-means (Jain, 2010), Hybrid Firefly algorithm with K-means (FAK-means) (Tang, Fong, Yang, & Deb, 2012), and Bisect K-means (Murugesan & Zhang, 2011a, 2011b). The purpose of this comparison is to investigate the effectiveness of WFA<sub>R</sub> in producing quality clusters, even though it has not been provided with the same support (i.e. number of clusters) as the other methods.

Table 5.5 includes the results of external metrics, while Table 5.6 includes the results on internal and relative metrics. Figure 5.3 includes a graphical representation of quality metrics obtained by WFA<sub>R</sub> and the state-of-the-art methods. All algorithms are implemented in the same environment and are executed thirty times on different iterations, and the average values for each metric are calculated.

As shown in Table 5.5, the WFA<sub>R</sub> algorithm generates the highest average Purity in all iterations compared against PSO, K-means, FAK-means, and Bisect K-means, while PSO produced the smallest purity.



Table 5.5

*External quality metrics: WFA<sub>R</sub> vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means.*

External Metrics	Algorithms	Iterations				
		1	2	5	10	20
Purity	WFA <sub>R</sub>	<b>0.8300</b> (0.0255)	<b>0.8258</b> (0.0095)	<b>0.8177</b> (0.0058)	<b>0.8182</b> (0.0036)	<b>0.7868</b> (0.0129)
	PSO	0.3823 (0.0432)	0.3861 (0.0643)	0.3726 (0.0433)	0.3731 (0.0403)	0.3772 (0.0516)
	K-means	0.3468 (0.0242)	0.3451 (0.0226)	0.3531 (0.0414)	0.3561 (0.0575)	0.3459 (0.0258)
	FAK-means	0.3658 (0.0133)	0.3701 (0.0150)	0.3738 (0.0138)	0.3719 (0.0147)	0.3731 (0.0132)
	Bisect K-means	0.3759 (0.0300)	0.3806 (0.0397)	0.3872 (0.0446)	0.3939 (0.0601)	0.4031 (0.0818)
F-measure	WFA <sub>R</sub>	<b>0.5199</b> (0.0399)	<b>0.5145</b> (0.0126)	<b>0.5194</b> (0.0174)	<b>0.5295</b> (0.0365)	<b>0.6186</b> (0.0426)
	PSO	0.4947 (0.0188)	0.5062 (0.0414)	0.4907 (0.0118)	0.4951 (0.0110)	0.4986 (0.0237)
	K-means	0.4910 (0.0213)	0.4935 (0.0138)	0.4975 (0.0196)	0.4999 (0.0316)	0.4954 (0.0081)
	FAK-means	0.3656 (0.0134)	0.3692 (0.0140)	0.3747 (0.0155)	0.3723 (0.0143)	0.3737 (0.0132)
	Bisect K-means	0.4698 (0.0297)	0.4757 (0.0284)	0.4775 (0.0336)	0.4785 (0.0567)	0.4908 (0.0644)
Entropy	WFA <sub>R</sub>	<b>0.6676</b> (0.0830)	<b>0.6921</b> (0.0345)	<b>0.7282</b> (0.0135)	<b>0.7315</b> (0.0206)	<b>0.7244</b> (0.0328)
	PSO	1.5350 (0.0393)	1.5230 (0.0858)	1.5403 (0.0513)	1.5309 (0.0619)	1.5276 (0.0691)
	K-means	1.5782 (0.013)	1.5794 (0.0100)	1.5722 (0.0341)	1.5636 (0.072)	1.5755 (0.0215)
	FAK-means	1.5782 (0.0042)	1.5770 (0.0059)	1.5751 (0.0054)	1.5750 (0.0067)	1.5754 (0.0056)
	Bisect K-means	1.5620 (0.0223)	1.5571 (0.0285)	1.5466 (0.0914)	1.5325 (0.0841)	1.5150 (0.1193)

*Note: the best value is highlighted in 'bold', standard deviation in ( ).*

Table 5.6

*Internal and Relative quality metrics: WFA<sub>R</sub> vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means.*

Internal & Relative Metrics	Algorithms	Iterations				
		1	2	5	10	20
ADDC	WFA <sub>R</sub>	1.4282 (0.233)	1.4228 (0.0156)	1.4171 (0.0130)	1.4085 (0.0408)	1.2222 (0.0744)
	PSO	1.7630 (0.2124)	1.7775 (0.2738)	1.8053 (0.2751)	1.9379 (0.1862)	1.8736 (0.0430)
	K-means	<b>0.7547</b> (0.2978)	<b>0.7041</b> (0.2861)	<b>0.6783</b> (0.2396)	<b>0.6367</b> (0.2511)	<b>0.7056</b> (0.3190)
	FAK-means	1.4434 (0.0010)	1.4436 (0.0007)	1.4436 (0.0009)	1.4432 (0.0009)	1.4436 (0.0007)
	Bisect K-means	1.3238 (0.1874)	1.2545 (0.2808)	1.2633 (0.2378)	1.3332 (0.2318)	1.3494 (0.1817)
DBI	WFA <sub>R</sub>	<b>1.6585</b> (0.0117)	1.6562 (0.0092)	1.6514 (0.0128)	1.6508 (0.0153)	<b>1.6176</b> (0.0427)
	PSO	1.7069 (0.0261)	<b>1.6100</b> (0.2168)	<b>1.5559</b> (0.2303)	<b>1.6092</b> (0.1732)	1.6472 (0.2269)
	K-means	2.8159 (3.5419)	2.3565 (3.1847)	2.4741 (3.1267)	1.9649 (3.1855)	1.9090 (2.5369)
	FAK-means	14.2277 (0.2063)	14.2834 (0.2025)	14.2549 (0.3551)	14.2637 (0.2862)	14.2158 (0.2801)
	Bisect K-means	8.1636 (3.0869)	7.3146 (3.0355)	6.9485 (3.0605)	7.8287 (2.8864)	7.7189 (2.7763)
DI	WFA <sub>R</sub>	0.9017 (0.0290)	0.9167 (0.0215)	0.9064 (0.0170)	0.8946 (0.0118)	0.9156 (0.0202)
	PSO	1.0162 (0.0950)	1.0331 (0.0574)	1.0357 (0.0650)	1.0119 (0.0583)	0.9908 (0.0899)
	K-means	<b>2.3413</b> (2.3505)	<b>2.9315</b> (2.4566)	<b>2.9671</b> (2.5302)	<b>3.7363</b> (2.3653)	<b>3.4078</b> (2.4518)
	FAK-means	0.1380 (0.0026)	0.1372 (0.0027)	0.1377 (0.0039)	0.1374 (0.0035)	0.1382 (0.0035)
	Bisect K-means	0.2393 (0.1483)	0.2876 (0.2299)	0.2998 (0.1887)	0.2715 (0.2178)	0.2698 (0.2008)

*Note: the best value is highlighted in 'bold', standard deviation in (.).*

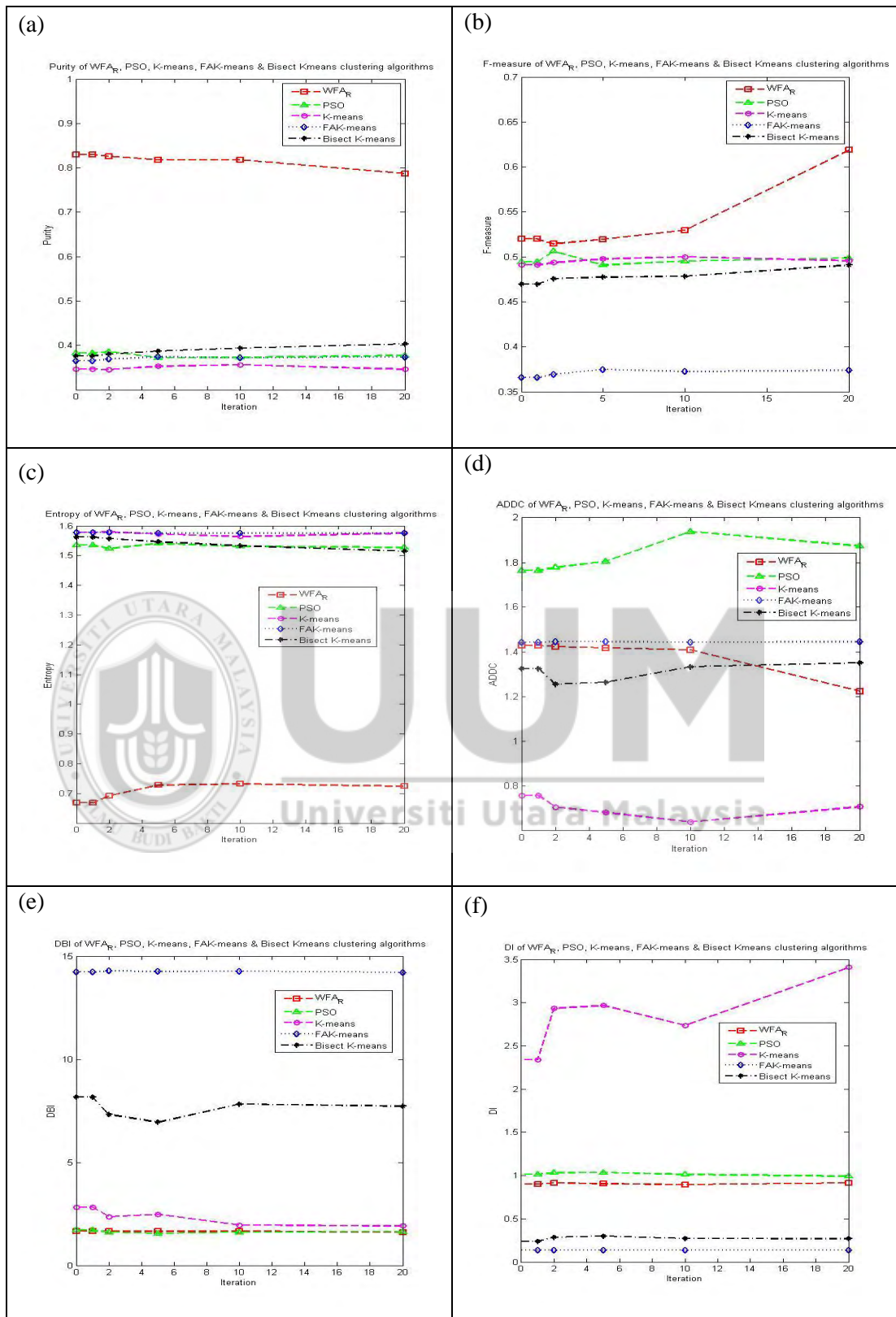


Figure 5.5. Graphical representation of quality metrics of WFA<sub>R</sub> vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means; a) Purity, b) F-measure, c) Entropy, d) ADDC, e) DBI, and f) DI.

Furthermore, it is noticed that Bisect K-means has a higher purity than K-means, FAK-means, and PSO. In addition, the overall standard deviation of the WFA<sub>R</sub> algorithm is smaller than others in most iterations (refer to iterations 2, 5, 10 and 20), which indicates it is more reliable and high in robustness. From these results, it can be concluded that the document re-locating algorithm highly affect the result of performance (purity) in WFA<sub>R</sub> which is the best than other methods. Figure 5.5.a shows the purity results of five methods: namely WFA<sub>R</sub>, PSO, K-means, FAK-means, and Bisect K-means, in graphical representation.

For the F-measure metrics as can be seen in Table 5.5, WFA<sub>R</sub> has a higher F-measure result compared to the other methods, while FAK-means has a smaller F-measure result compared to the other methods. For the overall standard deviation, the WFA<sub>R</sub> algorithm's value is smaller only than Bisect K-means in iterations 2, 5, 10 and 20. It can be concluded that the document re-locating algorithm highly affects the result of performance (F-measure) in WFA<sub>R</sub>, which is the best than other methods. Figure 5.5.b presents the F-measure result of five algorithms: WFA<sub>R</sub>, PSO, K-means, FAK-means, and Bisect K-means.

Table 5.5 reports the Entropy result of five methods: WFA<sub>R</sub>, PSO, K-means, FAK-means, and Bisect K-means. It is observed that the Entropy of WFA<sub>R</sub> is smaller than FAK-means, K-means, Bisect K-means, and PSO. The best Entropy values are 0.6676 and 0.6921 that are generated by WFA<sub>R</sub> in iterations 1 and 2. On the other hand, Bisect K-means is the best method after the proposed method which produced 1.5150 in iteration 20 that it is a smaller Entropy value compared than FAK-means, K-means, and PSO. Furthermore, the standard deviation of WFA<sub>R</sub> is smaller than

PSO, K-means, and Bisect K-means only in iterations 5 and 10 that means the solutions are more nearer to the average Entropy and contains less abnormal solutions. This result indicates that WFA<sub>R</sub> is best for producing more pure clusters. Figure 5.5.c shows the Entropy result in a graphical representation of five methods: WFA<sub>R</sub>, PSO, K-means, FAK-means, and Bisect K-means.

The ADDC values of the five techniques, WFA<sub>R</sub>, PSO, K-means, FAK-means, and Bisect K-means, are reported in Table 5.6 and presented in a graphical representation in Figure 5.5.d. The plotted graph shows that the curve of WFA<sub>R</sub> is smaller than FAK-means and PSO in all iterations and is better than Bisect K-means only in iteration 20. Whereas, K-means generates a smaller average ADDC value in all iterations. The standard deviation of WFA<sub>R</sub> is smaller than PSO, K-means, and Bisect K-means in most iterations (refer to iterations 2, 5 and 10), while FAK-means has a smaller standard deviation which is better than WFA<sub>R</sub> in generating abnormal solutions.

For the DBI metric, Table 5.6 reports the result of the five methods: WFA<sub>R</sub>, PSO, K-means, FAK-means, and Bisect K-means, and Figure 5.5.e presents a graphical representation of these methods. It shows that the DBI curve of the proposed WFA<sub>R</sub> algorithm is lower than K-means, Bisect K-means, and FAK-means in all iterations, excluding PSO, which is better than WFA<sub>R</sub> in generating a smaller DBI value in most iteration (refer to iterations 2, 5 and 10). In addition, it is noticed that the value of standard deviation in WFA<sub>R</sub> is smaller than other comparative methods. This result demonstrates that the document re-locating mechanism affects the DBI metrics in WFA<sub>R</sub>.

In Table 5.6 and Figure 5.5.f, the DI results of five methods are reported: WFA<sub>R</sub>, PSO, K-means, FAK-means, and Bisect K-means. The result of DI in K-means is better than other methods, followed by PSO and the proposed method, WFA<sub>R</sub> which is better than FAK-means and Bisect K-means in producing a higher DI. Furthermore, the standard deviation of WFA<sub>R</sub> is smaller than PSO, K-means, and Bisect K-means, while FAK-means's value is smaller than the proposed WFA<sub>R</sub>. This result indicates that the documents re-locating mechanism does not affect the DI metrics in WFA<sub>R</sub>.

Table 5.7 displays the average number of clusters that is automatically generated by WFA<sub>R</sub> and without any prior knowledge about the dataset. As can be seen in Table 5.7, the number of clusters obtained by WFA<sub>R</sub> is higher than the number of clusters in other techniques. This result indicates that the document re-locating algorithm does not affect the number of clusters.

Table 5.7

*Average number of clusters: WFA<sub>R</sub> vs. PSO vs. K-means vs. FA K-means vs. Bisect K-means.*

Iterations	Number of clusters of algorithms				
	WFA <sub>R</sub>	PSO	K-means	FAK-means	Bisect K-means
1	19.80 $\approx$ 20	3	3	3	3
2	19.06 $\approx$ 19	3	3	3	3
5	18.03 $\approx$ 18	3	3	3	3
10	17.63 $\approx$ 18	3	3	3	3
20	14.36 $\approx$ 14	3	3	3	3

The previous results indicate that  $WFA_R$  generates the best quality results in external performance metrics, Purity, F-measure, and Entropy, and relative metric, DBI. Table 5.8 illustrates the results of different quality performance metrics in five methods.

Table 5.8

*Summary of quality performance:  $WFA_R$  vs. PSO vs. K-means vs. FAK-means vs. Bisect K-means.*

Algorithms	External Metrics			Internal & Relative Metrics		
	Purity	F-measure	Entropy	ADDC	DBI	DI
$WFA_R$	✓	✓	✓		✓	
PSO					✓	
K-means				✓		✓
FAK-means						
Bisect K-means						

In the next chapters, this study will try to enhance the quality performance and the number of produced clusters of  $WFA_R$  by integrating it with merging algorithm.

## 5.4 Summary

This chapter presents a new mechanism to change the location of documents from existing clusters to the newly created cluster. The proposed document re-locating mechanism is introduced into the Weight-based Firefly Algorithm (WFA), presented in Chapter Four, and is termed as  $WFA_R$ . Two experiments are conducted to study the effect of  $WFA_R$  on the obtained clustering. First, is the experiment between WFA and  $WFA_R$ , while the second experiment compares between  $WFA_R$  and the state-of-

the-art methods: PSO, K-means, FAK-means, and Bisect K-means. These comparisons are undertaken using external performance metrics, Purity, F-measure and Entropy, and internal and relative metrics, ADDC, DBI and DI. The results indicate that  $WFA_R$  produced better external metrics (Purity, F-measure and Entropy) compared to WFA. Furthermore,  $WFA_R$  also produces better results in external performance metrics: Purity, F-measure and Entropy, and also in relative metrics: DBI when compared against the state-of-the-art methods; PSO, K-means, FAK-means, and Bisect K-means.





## CHAPTER SIX

### CLUSTER MERGING

#### 6.1 Introduction

This chapter proposes a cluster merging algorithm for text clustering. The algorithm is integrated in  $WFA_R$  (as in Chapter Five) and is known as  $WFA_{RM}$ . The use of the merging algorithm is to minimize the number of clusters produced by  $WFA_R$  which is not near the optimal number of clusters. In the undertaken experiments on the 20Newsgroups dataset, a comparison has been made between  $WFA_{RM}$  and  $WFA_R$ .

The proposed  $WFA_{RM}$  works automatically without any prior knowledge or any information about the datasets. Figure 6.1 displays the process in  $WFA_{RM}$  which includes three phases: clustering using Weight-based Firefly Algorithm (WFA) as in Chapter Four, documents re-locating as mentioned theoretically and experimentally in Chapter Five, and the last phase which is cluster merging.

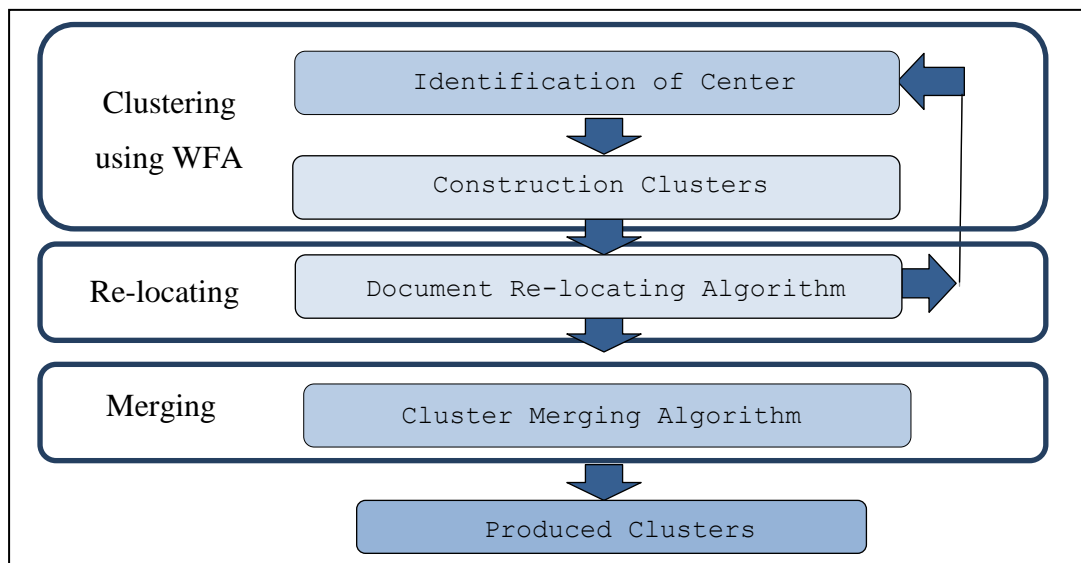


Figure 6.1. Process in  $WFA_{RM}$

## 6.2 Cluster Merging Algorithm

The merging algorithm can be introduced once clusters are constructed. It is an extension of the un-weighted pair group method with arithmetic mean (UPGMA) (Murugesan & Zhang, 2011a, 2011b) to produce less numbers of cluster. The cluster merging algorithm consists of two steps, merge clusters and refine merged clusters, as shown in Figure 6.2.

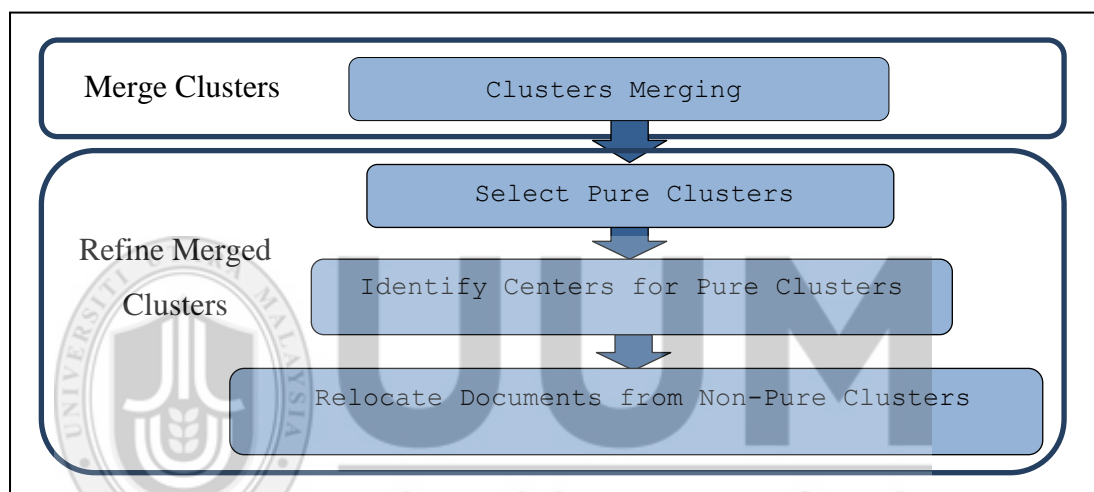


Figure 6.2. Process of cluster merging Algorithm (eUPGMA)

The merge clusters step combines similar clusters, while the refine merged clusters step includes three sub steps: select pure clusters, identify the centers of pure clusters, and relocate documents from non-pure clusters.

### 6.2.1 Merge Clusters

The process of merging similar clusters is illustrated in the following steps:

**Step 1:** Check to merge the first cluster in the output clusters with the remaining clusters in the output clusters, if no merge, eliminate the first cluster from the output clusters (not included in the merge process), then the second cluster

becomes the first cluster. The process of Steps 2-11 continues until the last cluster becomes the first cluster, so the merge process is stopped.

**Step 2:** Suppose that  $C_1$  and  $C_2$  are two clusters that want to merge, and suppose that  $P_1$  and  $P_2$  are the numbers of documents in two clusters respectively. Suppose that  $CSim$  is the Cosine similarity matrix between two clusters  $C_1$  and  $C_2$ . The documents in  $C_1$  are represented by the row and the documents in  $C_2$  are represented by the column. The value of the  $CSim$  matrix is equal to 1 if the document in  $C_1$  is similar to the document in  $C_2$ , else it equals 0. The similarity between two documents is based on threshold.

**Step 3:** If (the number of documents in cluster  $C_1 \geq 2$  and the number of documents in cluster  $C_2 \geq 2$ ) OR If (the number of documents in cluster  $C_1 \geq 3$  and the number of documents in cluster  $C_2 = 1$ ) OR If (the number of documents in cluster  $C_2 \geq 3$  and the number of documents in cluster  $C_1 = 1$ ) then

**Step 4:** Calculated the average similarity between two clusters as in Equation 6.1.

$$\frac{1}{P_1} \sum_{i=1}^{P_1} \sum_{j=1}^{P_2} \frac{CSim(C_i, C_j)}{P_2} \quad (6.1)$$

where,  $P_1$  is the number of document in the first cluster,  $P_2$  is the number of document in the second cluster,  $C_i$  is the first cluster,  $C_j$  is the second cluster.

**Step 5:** Calculate the merge threshold as in Equation 6.2 below.

$$MergeThreshold (MT) = floor \left( \frac{round \left( \frac{P_1 * P_2}{2} \right) - 1}{P_1 * P_2} * 10 \right) / 10 \quad (6.2)$$

**Step 6:** If Equation 6.1 passed the merge threshold in Equation 6.2 as shown in Equation 6.3, then, combine two clusters  $C_1$  and  $C_2$  into one cluster.

$$\frac{1}{P_1} \sum_{i=1}^{P_1} \sum_{j=1}^{P_2} \frac{CSim(C_i, C_j)}{P_2} \geq MergeThreshold(MT) \quad (6.3)$$

**Step 7:** If (the number of documents in cluster  $C_1 \geq 2$  and the number of documents in cluster  $C_2 \geq 1$ ) OR If (the number of documents in cluster  $C_2 \geq 2$  and the number of documents in cluster  $C_1 \geq 1$ )

**Step 8:** Combine  $C_1$  and  $C_2$ , if Equation 6.3 is true using Equation 6.4 to obtain merge threshold.

$$MergeThreshold(MT) = \frac{round(\frac{P_1 * P_2}{2})}{P_1 * P_2} \quad (6.4)$$

**Step 9:** If (the number of documents in cluster  $C_1 \geq 1$  and the number of documents in cluster  $C_2 \geq 1$ )

**Step 10:** Combine  $C_1$  and  $C_2$ , if  $CSim(C_1, C_2)$  equals to 1.

### 6.2.2 Refine Merged Clusters

Once the clusters are merged, there is a need to ensure that the obtained clusters contain pure members. This is implemented by checking the clusters' size by identifying the threshold, and in this research it is set to (50,  $n/20$ ). This threshold is based on the criterion used by Tan, Ting, and Teng (2011a), and the idea of the selected clusters is adopted from Picarougne, Azzag, Venturini, and Guinot (2007). The pseudo code of selecting pure clusters is illustrated in the following in Figure 6.3.

Once the pure clusters are identified, a new center for the selected pure clusters needs to be defined. The center is obtained by the sum of all TFIDF values of

documents in the specific cluster and divided by the number of documents in the cluster. The pseudo code for identifying centers is illustrated in Figure 6.4.

**Selecting pure clusters**

**Step 1:** Set selected threshold equal min (50, n/20).

**Step 2:** For i= 1 to number of clusters

**Step 3:** If length (Ci) >= selected threshold

**Step 4:** Save Ci in selected clusters.

**Step 5:** Else Save Ci in non-selected clusters.

**Step 6:** End.

*Figure 6.3. Pseudo code for selecting pure clusters*

**Identifying centers for pure clusters**

**Step 1:** For i= 1 to k (number of selected clusters)

**Step 2:** Calculate the center for each cluster as shown in equation 6.5.

$$Center(C_k) = \frac{\sum_{j=1}^{NC_k} TFIDF_{D_j}}{NC_k} \quad (6.5)$$

**Step 3:** End.

*Figure 6.4. Pseudo code of identifying centers for pure clusters*

Later, all documents in the non-pure clusters need to be relocated into the pure clusters. This is done by measuring the distance between the documents and the newly identified centers. Figure 6.5 illustrates the proposed pseudo code of relocating non-pure clusters.

**Relocating non-pure clusters**

**Step 1:** For  $i= 1$  to (number of documents in non-pure clusters)

**Step 2:** Find minimum distance between document  $D_i$  and center of  $C_1$  using Equation 6.6 as shown below.

$$mindistance(D_i, Center_{C_1}) = \sum_{j=1}^m (D_{ij} - Center_{C_1})^2 \quad (6.6)$$

**Step 3:** Assign  $D_i=1$

**Step 4:** For  $k= 2$  to (number of selected pure clusters)

**Step 5:** Find minimum distance between document  $D_i$  and center of  $C_k$  using Equation 6.7 as shown below.

$$mindistance2(D_i, Center_{C_k}) = \sum_{j=1}^m (D_{ij} - Center_{C_k})^2 \quad (6.7)$$

**Step 6:** If ( $mindistance \geq mindistance2$ )

**Step 7:** Assign  $D_i=k$

**Step 8:**  $mindistance = mindistance2$

**Step 9:** End For

**Step 10:** Assign  $D_i$  to  $C_k$

**Step 11:** End For

*Figure 6.5. Pseudo code of relocating non-pure clusters*

The following example (based on the example presented in Chapter Five) demonstrates the proposed cluster merging algorithm. The produced clusters by  $WFA_R$  are as follows.

Cluster1 [1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

Cluster2 [21, 22, 23, 24, 25, 26, 27, 29, 2, 30]

Cluster3 [28]

	Cluster 2 (10 documents)											Result of average similarity
		D21	D22	D23	D24	D25	D26	D27	D29	D2	D30	
Cluster 1 (19 documents)	D1	0	0	0	0	0	0	0	0	1	0	= 1/10
	D3	0	0	0	1	1	1	0	1	1	1	= 6/10
	D4	0	0	0	0	0	0	0	0	1	0	= 1/10
	D5	0	0	0	0	0	0	0	0	1	0	= 1/10
	D6	0	0	0	0	1	0	0	0	1	0	= 2/10
	D7	0	0	0	0	0	0	0	0	1	0	= 1/10
	D8	1	0	0	0	1	0	0	0	1	1	= 4/10
	D9	1	0	0	0	1	0	0	0	1	1	= 4/10
	D10	0	0	0	0	0	0	0	0	1	1	= 2/10
	D11	0	0	0	0	0	0	0	0	0	0	= 0
	D12	0	0	0	0	0	0	0	0	0	0	= 0
	D13	0	0	0	0	1	0	0	0	1	0	= 2/10
	D14	0	1	0	0	0	0	0	0	1	0	= 2/10
	D15	0	0	0	0	0	0	0	0	0	1	= 1/10
	D16	0	0	0	0	0	0	0	0	0	1	= 1/10
	D17	0	0	0	0	1	0	0	0	1	1	= 3/10
	D18	0	0	0	0	0	0	0	0	1	0	= 1/10
	D19	0	0	0	0	0	0	0	0	1	0	= 1/10
	D20	0	0	0	0	0	0	0	0	1	0	= 1/10
												= 34/10 = 3.4/19 = 0.178

Figure 6.6. Cosine similarity matrix between cluster1 and cluster2

Where, the number of documents in Cluster 1 is  $P_1=19$ , the number of documents in Cluster 2 is  $P_2=10$ , the number of documents in Cluster 3 is  $P_3=1$ , and the total number of documents is 30. In the proposed  $WFA_{RM}$ , the cosine similarity between Cluster 1 and Cluster 2 (CSim) is checked as illustrated in Figure 6.6.

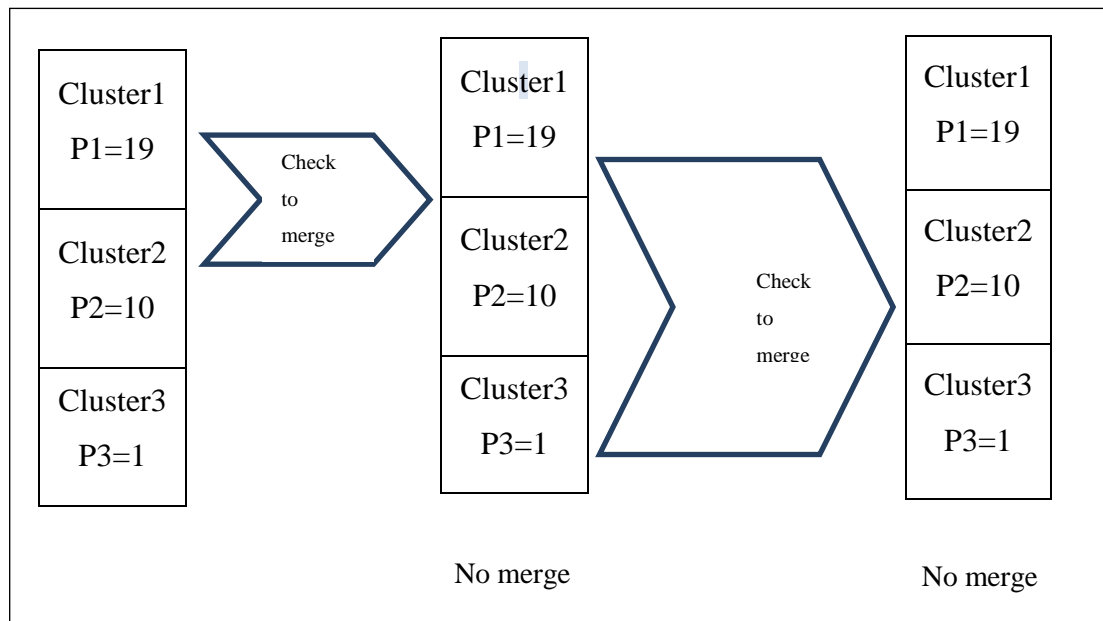
The similarity between document D1 in Cluster 1 and D2 in Cluster 2 exceeds the similarity threshold; hence, the cosine similarity matrix (CSim matrix) is set to 1. The average similarity is calculated using Equation 6.1, as shown in Figure 6.6, where in this example, it is equal to 0.178.

The merge threshold value is calculated using Equation 6.2, and it is noted to be 0.4. In detail, the calculation is as follows.

$$\begin{aligned}
 \text{Merge Threshold} &= \text{Floor} [(((\text{round} ((19*10)/2)) - 1) / 190) * 10)] / 10 \\
 &= \text{Floor} [(((95-1)/190) * 10) / 10] \\
 &= \text{Floor} [0.4947 * 10] / 10 \\
 &= 4/10 = 0.4
 \end{aligned}$$

The value of average similarity, which is 0.178, is smaller than the merge threshold value of 0.4. Thus, the two clusters, Cluster 1 and Cluster 2, cannot be merged. The same process is repeated for Cluster 3. Figure 6.7 presents the produced clusters at the end of the merging process.





*Figure 6.7. Results of merging clusters for 20Newsgroups dataset*

The process of selecting pure clusters starts after the merging cluster has completed. The produced clusters from the merging clusters step are Cluster 1, Cluster 2 and Cluster 3. Where, the number of documents in each cluster is  $P_1=19$ ,  $P_2=10$ ,  $P_3=1$ . The selected threshold is  $(50, 30/20)$  which equal to  $(50, 1.5)$ , and this means the cluster which contains documents more than 1.5 will be identified as the selected cluster, and the ones less than 1.5 will be known as non-selected cluster. The selected pure clusters are Cluster 1 and Cluster 2, while, non-pure cluster is Cluster 3.

The third step is to identify the centers for the pure clusters. Figure 6.8 presents the example of TFIDF of the documents in the first cluster, Cluster 1, and presents the calculation of the center.

	<b>Term<sub>1</sub></b>	<b>Term<sub>2</sub></b>	<b>...</b>	<b>Term<sub>m=549</sub></b>
D1	0.5	0.1	...	0.7
D2	0.2	0.2	...	0.6
.	.	.		.
.	.	.		.
D <sub>19</sub>	0.5	0.3	...	0.4
Center	$\frac{Sum(D_1:D_{19})}{19}$	$\frac{Sum(D_1:D_{19})}{19}$	...	$\frac{Sum(D_1:D_{19})}{19}$

Figure 6.8. An example of TFIDF of documents in Cluster1 and center calculation

Once the centers are identified, the process of the relocating mechanism of non-pure clusters can be performed. The non-pure cluster is Cluster 3 which includes only one document, D28. In this example, D28 is assigned to Cluster 2. Figure 6.9 presents an example of TFIDF of D28 in Cluster 3, and Figure 6.10 presents an example of the centers of selected pure clusters.

	<b>Term<sub>1</sub></b>	<b>Term<sub>2</sub></b>	<b>Term<sub>3</sub></b>	<b>Term<sub>4</sub></b>
D28	0.2	0.3	0.4	0.7

Figure 6.9. An example of TFIDF of document 28 in Cluster 3

	<b>Term<sub>1</sub></b>	<b>Term<sub>2</sub></b>	<b>Term<sub>2</sub></b>	<b>Term<sub>4</sub></b>
Center Cluster1	0.5	0.3	0.2	0.6
Center Cluster2	0.3	0.2	0.4	0.6

Figure 6.10. An example of the centers of selected pure clusters

This is achieved by calculating the distance between document  $D_{28}$  and the center of Cluster 1 and Cluster 2. The calculation process is illustrated in Figure 6.11. As can be seen in Figure 6.11, the minimum distance is (0.0300) between document  $D_{28}$  and the center of Cluster 2, so  $D_{28}$  is relocated to Cluster 2.

D28	0.2	0.3	0.4	0.7
Center Cluster 1	0.5	0.3	0.2	0.6
Mindistance (D <sub>1</sub> , Center Cluster 1)	0.3	0.0	0.2	0.1
Mindistance (D <sub>1</sub> , Center Cluster 1) <sup>2</sup>	0.09	0.0	0.04	0.01
Mindistance (D <sub>1</sub> , Center Cluster 1) <sup>2</sup>	0.1400			
D28	0.2	0.3	0.4	0.7
Center Cluster 2	0.3	0.2	0.4	0.6
Mindistance (D <sub>1</sub> , Center Cluster 2)	0.1	0.1	0.0	0.1
Mindistance (D <sub>1</sub> , Center Cluster 2) <sup>2</sup>	0.01	0.01	0.0	0.01
Mindistance (D <sub>1</sub> , Center Cluster 2) <sup>2</sup>	0.0300			

Figure 6.11. Calculation of minimum distance between centers of pure clusters and members of non-pure cluster

Finally, the following clusters are obtained:

Cluster1 [1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

Cluster2 [21, 22, 23, 24, 25, 26, 27, 29, 2, 30, 28]

### 6.3 Evaluation

The evaluation of  $WFA_{RM}$  clustering is performed in three parts: comparison between  $WFA_R$  and  $WFA_{RM}$ , comparison between  $WFA_{RM}$  against state-of-the-art methods (static methods), and comparison between  $WFA_{RM}$  and state-of-the-art methods (dynamic methods).

### 6.3.1 Comparison between $WFA_{RM}$ and $WFA_R$

The comparison between  $WFA_{RM}$  and  $WFA_R$  consists of three parts of evaluation: number of clusters, performance metrics, and paired samples T-test.

#### 6.3.1.1 Number of Clusters between $WFA_{RM}$ and $WFA_R$

In Table 6.1, it reports the average number of clusters produced by the  $WFA_R$  and  $WFA_{RM}$  algorithms. As seen in the table, the number of clusters obtained by  $WFA_{RM}$  in iteration 20 equals the cluster number of the 20Newsgroups dataset (i.e. 3). It can be concluded that the cluster merging highly affects the number of produced clusters in  $WFA_R$ . Figure 6.12 shows the number of produced clusters in the two algorithms.

Table 6.1

*Average number of clusters of  $WFA_R$  &  $WFA_{RM}$ .*

Iterations	Number of clusters of algorithms	
	$WFA_R$	$WFA_{RM}$
1	19.80 $\approx$ 20	<b>3.9 <math>\approx</math> 4</b>
2	19.06 $\approx$ 19	<b>3.93 <math>\approx</math> 4</b>
5	18.03 $\approx$ 18	<b>4.63 <math>\approx</math> 5</b>
10	17.63 $\approx$ 18	<b>4.7 <math>\approx</math> 5</b>
20	14.36 $\approx$ 14	<b>3</b>

*Note: the best value is highlighted in 'bold'*

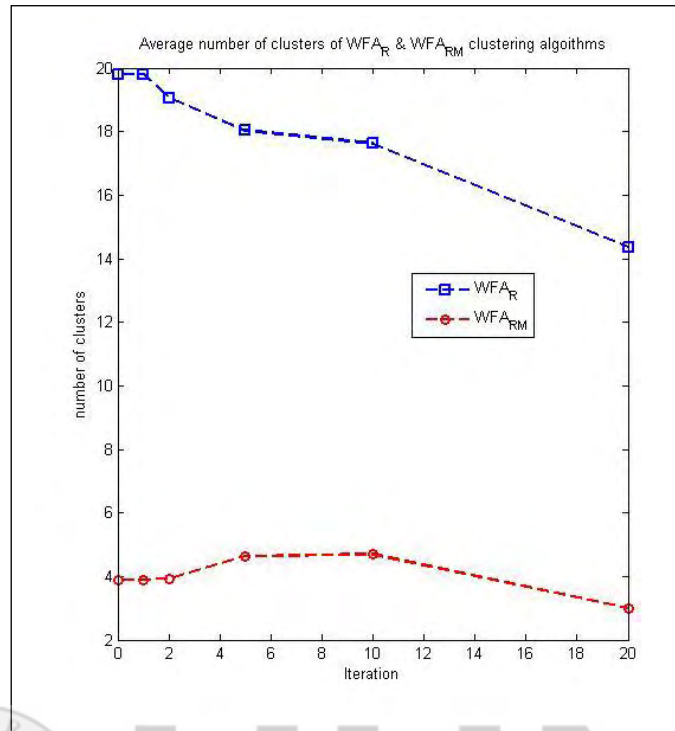


Figure 6.12. Number of produced clusters by WFA<sub>R</sub> and WFA<sub>RM</sub>

### 6.3.1.2 Performance Metrics between WFA<sub>RM</sub> and WFA<sub>R</sub>

The produced clusters by WFA<sub>RM</sub> are evaluated using the Purity, F-measure, Entropy, ADDC, DBI and DI metrics. Tables 6.2 and Table 6.3 include the average, best, worst and standard deviation for the metrics. Additionally, Figure 6.13 presents the graphical representation of metrics between WFA<sub>RM</sub> and WFA<sub>R</sub>.

As shown in Table 6.2, the WFA<sub>RM</sub> algorithm generates the highest average purity in the last iteration (refer to iteration 20) compared to the WFA<sub>R</sub> algorithm. The best purity value is 0.7948 that is generated by WFA<sub>RM</sub> in iteration 20, while WFA<sub>R</sub> generates 0.7868 in the same iteration. In addition, the overall standard deviation of the WFA<sub>RM</sub> algorithm in iteration 20 is nearly similar to WFA<sub>R</sub>, which indicates the solution that is generated by WFA<sub>RM</sub> is more reliable and high in robustness.

Table 6.2

*External quality metrics of clustering and standard deviation: WFA<sub>R</sub> vs. WFA<sub>RM</sub>.*

Iterations	Algorithms	Purity Metric			
		Average Purity	Best Purity	Worst Purity	Standard Deviation
1	WFA <sub>R</sub>	<b>0.8300</b>	0.9467	0.7967	<b>0.0255</b>
	WFA <sub>RM</sub>	0.7758	0.9133	0.7233	0.0319
2	WFA <sub>R</sub>	<b>0.8258</b>	0.8700	0.8100	0.0095
	WFA <sub>RM</sub>	0.7547	0.7733	0.7300	<b>0.0080</b>
5	WFA <sub>R</sub>	<b>0.8177</b>	0.8233	0.7900	<b>0.0058</b>
	WFA <sub>RM</sub>	0.7881	0.8100	0.7300	0.0165
10	WFA <sub>R</sub>	<b>0.8182</b>	0.8267	0.8100	<b>0.0036</b>
	WFA <sub>RM</sub>	0.7898	0.8067	0.7300	0.0134
20	WFA <sub>R</sub>	0.7868	0.8100	0.7433	<b>0.0129</b>
	WFA <sub>RM</sub>	<b>0.7948</b>	0.8100	0.7800	0.0132

Iterations	Algorithms	F-measure Metric			
		Average F-measure	Best F-measure	Worst F-measure	Standard Deviation
1	WFA <sub>R</sub>	0.5199	0.6625	0.4930	<b>0.0339</b>
	WFA <sub>RM</sub>	<b>0.7052</b>	0.9110	0.6474	0.0606
2	WFA <sub>R</sub>	0.5145	0.5423	0.4983	<b>0.0126</b>
	WFA <sub>RM</sub>	<b>0.6765</b>	0.7681	0.6545	0.0182
5	WFA <sub>R</sub>	0.5194	0.5657	0.5080	<b>0.0174</b>
	WFA <sub>RM</sub>	<b>0.6676</b>	0.8143	0.6473	0.0463
10	WFA <sub>R</sub>	0.5295	0.6589	0.5034	0.0365
	WFA <sub>RM</sub>	<b>0.6575</b>	0.8116	0.6473	<b>0.0300</b>
20	WFA <sub>R</sub>	0.6186	0.6589	0.5551	0.0426
	WFA <sub>RM</sub>	<b>0.7997</b>	0.8143	0.7842	<b>0.0129</b>

Iterations	Algorithms	Entropy Metric			
		Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
1	WFA <sub>R</sub>	<b>0.6676</b>	0.2806	0.7250	0.0830
	WFA <sub>RM</sub>	0.8089	0.4412	0.8916	<b>0.0775</b>
2	WFA <sub>R</sub>	<b>0.6921</b>	0.5299	0.7396	0.0345
	WFA <sub>RM</sub>	0.8584	0.7986	0.9063	<b>0.0178</b>
5	WFA <sub>R</sub>	<b>0.7282</b>	0.7008	0.7453	<b>0.0135</b>
	WFA <sub>RM</sub>	0.8206	0.7642	0.8963	0.0246
10	WFA <sub>R</sub>	<b>0.7315</b>	0.6587	0.7453	<b>0.0206</b>
	WFA <sub>RM</sub>	0.8183	0.7472	0.8963	0.0232
20	WFA <sub>R</sub>	<b>0.7244</b>	0.6587	0.8381	0.0326
	WFA <sub>RM</sub>	0.7827	0.7472	0.8130	<b>0.0267</b>

*Note: the best value is highlighted in 'bold'*

Table 6.3

*Internal and relative quality metrics of clustering and standard deviation: WFA<sub>R</sub> vs. WFA<sub>RM</sub>.*

Iterations	Algorithms	ADDC Metric			
		Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
1	WFA <sub>R</sub>	1.4282	1.3688	1.4557	0.0233
	WFA <sub>RM</sub>	<b>1.4158</b>	1.3914	1.4366	<b>0.0138</b>
2	WFA <sub>R</sub>	1.4228	1.3640	1.4557	0.0156
	WFA <sub>RM</sub>	<b>1.4058</b>	1.3980	1.4327	<b>0.0067</b>
5	WFA <sub>R</sub>	1.4171	1.3683	1.4413	0.0130
	WFA <sub>RM</sub>	<b>1.4153</b>	1.3918	1.4304	<b>0.0083</b>
10	WFA <sub>R</sub>	<b>1.4085</b>	1.1994	1.4488	0.0408
	WFA <sub>RM</sub>	1.4180	1.4005	1.4304	<b>0.0072</b>
20	WFA <sub>R</sub>	<b>1.2222</b>	1.1569	1.3949	0.0744
	WFA <sub>RM</sub>	1.4263	1.4229	1.4273	<b>0.0018</b>

Iterations	Algorithms	DBI Metric			
		Average DBI	Best DBI	Worst DBI	Standard Deviation
1	WFA <sub>R</sub>	<b>1.6585</b>	1.6313	1.6740	<b>0.0117</b>
	WFA <sub>RM</sub>	7.3303	6.6973	8.1854	0.6042
2	WFA <sub>R</sub>	<b>1.6562</b>	1.6386	1.6828	<b>0.0092</b>
	WFA <sub>RM</sub>	7.1238	6.6986	8.1854	0.2775
5	WFA <sub>R</sub>	<b>1.6514</b>	1.6386	1.7012	<b>0.0128</b>
	WFA <sub>RM</sub>	7.3802	6.8186	9.2199	0.6246
10	WFA <sub>R</sub>	<b>1.6508</b>	1.5967	1.7104	<b>0.0153</b>
	WFA <sub>RM</sub>	7.3298	6.8264	9.4829	0.4761
20	WFA <sub>R</sub>	<b>1.6176</b>	1.5744	1.7104	<b>0.0427</b>
	WFA <sub>RM</sub>	9.2225	9.0828	9.4829	0.1518

Iterations	Algorithms	DI Metric			
		Average DI	Best DI	Worst DI	Standard Deviation
1	WFA <sub>R</sub>	<b>0.9017</b>	0.9415	0.8522	0.0290
	WFA <sub>RM</sub>	0.2330	0.2489	0.2248	<b>0.0070</b>
2	WFA <sub>R</sub>	<b>0.9167</b>	0.9287	0.8522	0.0215
	WFA <sub>RM</sub>	0.2328	0.2341	0.2297	<b>0.0011</b>
5	WFA <sub>R</sub>	<b>0.9064</b>	0.9287	0.8808	0.0170
	WFA <sub>RM</sub>	0.2283	0.2339	0.2066	<b>0.0071</b>
10	WFA <sub>R</sub>	<b>0.8946</b>	0.9016	0.8495	0.0118
	WFA <sub>RM</sub>	0.2292	0.2339	0.1980	<b>0.0060</b>
20	WFA <sub>R</sub>	<b>0.9156</b>	0.9264	0.8495	0.0202
	WFA <sub>RM</sub>	0.2058	0.2087	0.1980	<b>0.0044</b>

*Note: the best value is highlighted in 'bold'.*

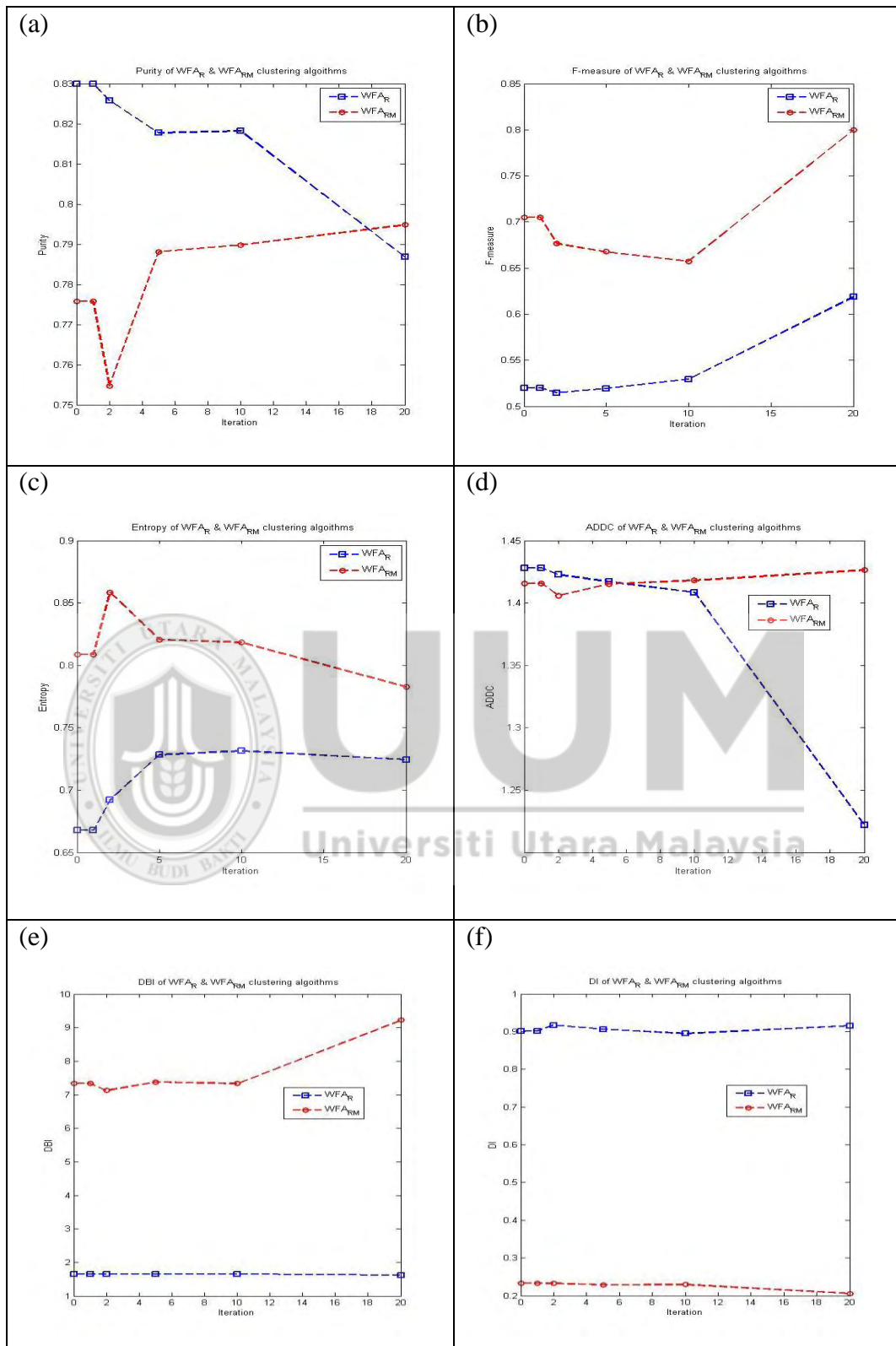


Figure 6.13. Graphical representation of quality metrics:  $WFA_R$  vs.  $WFA_{RM}$ , (a) Purity, (b) F-measure, (c) Entropy, (d) ADDC, (e) DBI, and (f) DI



From these results, it can be concluded that the cluster merging algorithm highly affects the result of performance (purity) in  $WFA_R$ . Figure 6.13.a illustrates the result of purity in a graphical representation of these two methods,  $WFA_{RM}$  and  $WFA_R$ . It shows that the purity curve of  $WFA_{RM}$  increases when the iteration increases, in contrast with  $WFA_R$  which decrease when the iteration increases.

Table 6.2 presents the average F-measure results of the two different methods,  $WFA_{RM}$  and  $WFA_R$ . As shown in Table 6.2,  $WFA_{RM}$  generates the highest average F-measure in all iterations. Furthermore, it is observed that the best F-measure produced is 0.7997 in iteration 20 by  $WFA_{RM}$  when the number of clusters equals 3, while  $WFA_R$  generates the best F-measure value of 0.6186 in iteration 20 when the number of clusters equals 14.36. The previous observations mean that the cluster merging algorithm highly affects the result of F-measure in  $WFA_R$ . Moreover, the overall standard deviation of  $WFA_{RM}$  is smaller than  $WFA_R$  in the last two iterations (refer to iterations 10 and 20), which indicates that the solution generated by  $WFA_{RM}$  is highly reliable and high in robustness. Figure 6.13.b demonstrates the result of the average F-measure in a pictorial representation of  $WFA_{RM}$  and  $WFA_R$ . It shows the F-measure curve of  $WFA_{RM}$  increases when the iteration increases, and rises in all iterations as compared to the curve of  $WFA_R$ .

In Table 6.2, it reports the average Entropy value, best Entropy value, worst Entropy value and standard deviation of thirty executions for each  $WFA_{RM}$  and  $WFA_R$  method. As shown in the table, the average Entropy of  $WFA_R$  is smaller than  $WFA_{RM}$  in all iterations, where, the best average Entropy value is 0.7244 generated by  $WFA_R$  in iteration 20, while  $WFA_{RM}$  generates 0.7827 in the same iteration. This

result indicates that  $WFA_R$  is better than  $WFA_{RM}$  in Entropy value, despite the number of produced clusters by  $WFA_{RM}$  which is near optimal clusters (3). The standard deviation of  $WFA_{RM}$  is smaller than  $WFA_R$  in iterations 1, 2 and 20, while  $WFA_R$  is smaller in the remaining iterations. This result means the solutions are nearer to the average Entropy and contain less abnormal solutions. It indicates that  $WFA_{RM}$  is best for producing more pure clusters. Figure 6.13.c shows the result of the average Entropy in a graphical representation of  $WFA_{RM}$  and  $WFA_R$ .

Table 6.3 includes the average ADDC of thirty executions for each  $WFA_{RM}$  and  $WFA_R$  algorithm, and also includes the best ADDC, worst ADDC and standard deviation. As shown in Table 6.3, the average ADDC produced by  $WFA_{RM}$  is smaller than  $WFA_R$  in most iterations (refer to iterations 1, 2, and 5). The best average ADDC value is 1.2222 generated by  $WFA_R$  in iterations 20 with the number of clusters of 14.36, while  $WFA_{RM}$  generates 1.4263 in the same iteration with the number of clusters of 3. The overall standard deviation of  $WFA_{RM}$  is smaller than  $WFA_R$  in all iterations, which means the solutions generated by  $WFA_{RM}$  are more robust and do not contain abnormal solutions. Based on previous observations,  $WFA_{RM}$  outperforms  $WFA_R$  in ADDC value. Figure 6.13.d shows the result of the average ADDC in a graphical representation of  $WFA_{RM}$  and  $WFA_R$ .

Table 6.3 presents the average DBI of thirty executions for each  $WFA_{RM}$  and  $WFA_R$  algorithm, and also includes the best, worst DBI and standard deviation. As shown in table,  $WFA_R$  generates a higher average DBI in all iterations compared to  $WFA_{RM}$ . This result occurs because the number of clusters in  $WFA_{RM}$  is smaller than  $WFA_R$ . The standard deviation of  $WFA_R$  is smaller in most runs. From previous results, it

can be concluded that cluster merging algorithm makes changes on the DBI value on  $WFA_R$ . Figure 6.13.e shows the result of the average DBI in a graphical representation of  $WFA_{RM}$  and  $WFA_R$ .

Table 6.3 includes the average DI of thirty executions for each  $WFA_{RM}$  and  $WFA_R$  algorithm, and also includes the best, worst DI, and standard deviation. DI measures the ratio of the smallest distance between observations not in the same cluster to the largest intra cluster distance. The largest value of DI means a more compact clustering solution. Examining Table 6.3, it is clear that  $WFA_R$  generates the largest average DI in all iterations, despite the value of standard deviation of  $WFA_{RM}$  is smaller in all iterations. The DI metric is highly effective with the number of clusters. As it can be seen, when the number of clusters in  $WFA_R$  is 14.36 in iteration 20, the DI value is 0.9156 while the number of clusters in  $WFA_{RM}$  is 3 in the same iteration with a DI value of 0.2058. These results demonstrate that the cluster merging algorithm makes changes on the DI metrics in  $WFA_R$ . Figure 6.13.f illustrates the result of the average DI in a graphical representation of  $WFA_{RM}$  and  $WFA_R$ .

From previous discussion, it can be concluded that the cluster merging algorithm enhances the external metrics, namely purity, F-measure and ADDC, in producing clusters in  $WFA_R$ , and also enhances the number of obtained clusters where it gets near optimal cluster 3. Table 6.4 presents the summary of comparison between  $WFA_{RM}$  and  $WFA_R$  algorithms.

Table 6.4

*Quality performance of WFA<sub>R</sub> & WFA<sub>RM</sub> algorithms.*

Performance Metrics	Algorithms	
	WFA <sub>R</sub>	WFA <sub>RM</sub>
Purity		✓
F-measure		✓
Entropy	✓	
ADDC		✓
DBI	✓	
DI	✓	
Number of clusters		✓

### 6.3.1.3 Paired Samples T-test between WFA<sub>RM</sub> and WFA<sub>R</sub>

The statistical analysis of paired samples T-test is performed on the differences between the pairs of WFA<sub>RM</sub> and WFA<sub>R</sub>. The null hypotheses,  $H_0$ , refers to no difference between the mean result of the WFA<sub>RM</sub> and WFA<sub>R</sub> algorithms, while the alternative hypotheses,  $H_1$ , means that there is a difference between the mean result of the WFA<sub>RM</sub> and WFA<sub>R</sub> algorithms.

$H_0$ : There is no difference between the mean of two algorithms.

$H_1$ : There is a difference between the mean of two algorithms.

Table 6.5 illustrates the p-value using the samples of purity, F-measure, Entropy, ADDC, DBI and DI metrics of iteration 20 for WFA<sub>RM</sub> and WFA<sub>R</sub>. As can be seen in Table 6.5, the p-value between WFA<sub>RM</sub> and WFA<sub>R</sub> is less than (0.05). This means that the cluster merging algorithm has an effect on the evaluated metrics. Hence, the null hypothesis is rejected and this concludes that there is sufficient evidence to

accept the alternative hypotheses that there is a difference between the mean of the  $WFA_{RM}$  and  $WFA_R$  algorithms.

Table 6.5

*The P-value between  $WFA_R$  &  $WFA_{RM}$  algorithms.*

Metrics	P-value between $WFA_{RM}$ & $WFA_R$
Purity	0.02852000
F-measure	2.7299E-20
Entropy	9.16548E-9
ADDC	3.1431E-15
DBI	1.2419E-51
DI	2.1466E-46

### 6.3.2 Comparison between $WFA_{RM}$ and Static Methods

In this section, a comparison is made between the  $WFA_{RM}$  algorithm and several state-of-the-art methods; Particle Swarm Optimization (PSO) (Cui, Potok, & Palathingal, 2005), K-means (Jain, 2010), Bisect K-means (Murugesan & Zhang, 2011a, 2011b), Hybrid Firefly algorithm with K-means (FAK-means) (Tang, Fong, Yang, & Deb, 2012), and BatK-means (Tang, Fong, Yang, & Deb, 2012). The comparison consists of three parts of evaluation: number of clusters, performance metrics, and independent samples T-test.

#### 6.3.2.1 Number of Clusters between $WFA_{RM}$ and Static Methods

Table 6.6 displays the average number of clusters that has been automatically produced by  $WFA_{RM}$ . As seen in Table 6.6, the number of clusters produced by  $WFA_{RM}$  in iteration 20 is equal to the number of clusters in other algorithms.

Table 6.6

*Average number of clusters: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means.*

Iterations	Number of clusters of algorithms					
	WFA <sub>RM</sub>	PSO	K-means	Bisect K-means	FAK- means	BatK- means
1	<b>3.9 <math>\approx</math> 4</b>	3	3	3	3	3
2	<b>3.93 <math>\approx</math> 4</b>	3	3	3	3	3
5	<b>4.63 <math>\approx</math> 5</b>	3	3	3	3	3
10	<b>4.7 <math>\approx</math> 5</b>	3	3	3	3	3
20	<b>3</b>	3	3	3	3	3

### 6.3.2.2 Performance Metrics between WFA<sub>RM</sub> and Static Methods

In this section, WFA<sub>RM</sub> is compared against two types of static methods: single methods such as PSO, K-means, and Bisect K-means, and also hybrid methods such as Hybrid Firefly algorithm with K-means (FAK-means) and Hybrid Bat algorithm with K-means (BatK-means). Table 6.7 includes the quality performance results of external metrics, which are Purity, F-measure and Entropy, for six algorithms: WFA<sub>RM</sub>, PSO, K-means, Bisect K-means, FAK-means, and BatK-means. Table 6.8 includes the quality performance results of internal and relative metrics, which are ADDC, DBI and DI, for the mentioned algorithms. A graphical representation of the results is shown in Figure 6.14.

As shown in Table 6.7, the WFA<sub>RM</sub> algorithm generates the highest average purity in all iterations compared to other algorithms. The best purity value is 0.7948 generated by WFA<sub>RM</sub> in iteration 20, while BatK-means generates 0.6650 in the same iteration and K-means generates the worst value of 0.3459. This is because K-means is

trapped in local optima. In addition, the overall standard deviation of the WFA<sub>RM</sub> algorithm in iteration 20 is smaller than the other methods. This result indicates that the solution generated by WFA<sub>RM</sub> is more reliable and high in robustness. From these results, it can be concluded that the cluster merging algorithm highly affects the result of performance (purity) in WFA<sub>RM</sub>. Figure 6.14.a illustrates the purity result in a graphical representation of the following methods: WFA<sub>RM</sub>, PSO, K-means, Bisect K-means, FAK-means, and BatK-means.

For the F-measure metric, as seen in Table 6.7, WFA<sub>RM</sub> generates the highest average F-measure in all iterations. Besides that, it can be observed that the best F-measure produced is 0.7997 in iteration 20 by WFA<sub>RM</sub> when the value of precision and recall is higher (0.8047, 0.7948) respectively, and when the number of generated clusters is equal to 3, while BatK-means generates 0.6649 in the same iteration with a static number of clusters. The previous observations mean that WFA<sub>RM</sub> is better than other methods. Figure 6.14.b demonstrates the result of the average F-measure in a pictorial representation of WFA<sub>RM</sub>, PSO, K-means, Bisect K-means, FAK-means, and BatK-means.

In Table 6.7, it reports the average Entropy value and standard deviation of thirty executions for WFA<sub>RM</sub>, PSO, K-means, Bisect K-means, FAK-means, and BatK-means. As shown in Table 6.7, the average Entropy of the WFA<sub>RM</sub> is smaller than other methods in all iterations, where, the best average Entropy value is 0.7827 generated by WFA<sub>RM</sub> in iteration 20, while BatK-means generates 1.0162 in the same iteration.

Table 6.7

External quality metrics of clustering:  $WFA_{RM}$  vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means.

External Metrics	Algorithms	Iterations					External Metrics	Iterations				
		1	2	5	10	20		1	2	5	10	20
Purity	WFA <sub>RM</sub>	<b>0.7758</b> (0.0319)	<b>0.7547</b> (0.0080)	<b>0.7881</b> (0.0165)	<b>0.7898</b> (0.0134)	<b>0.7948</b> (0.0132)	Entropy	<b>0.8089</b> (0.0775)	<b>0.8584</b> (0.0178)	<b>0.8206</b> (0.0246)	<b>0.8183</b> (0.0232)	<b>0.7827</b> (0.0267)
	PSO	0.3823 (0.0432)	0.3861 (0.0643)	0.3726 (0.0433)	0.3731 (0.0403)	0.3772 (0.0516)		1.5350 (0.0393)	1.5230 (0.0858)	1.5403 (0.0513)	1.5309 (0.0619)	1.5276 (0.0691)
	K-means	0.3468 (0.0242)	0.3451 (0.0226)	0.3531 (0.0414)	0.3561 (0.0575)	0.3459 (0.0258)		1.5782 (0.013)	1.5794 (0.0100)	1.5722 (0.0341)	1.5636 (0.072)	1.5755 (0.0215)
	Bisect	0.3759 (0.0300)	0.3806 (0.0397)	0.3872 (0.0446)	0.3939 (0.0601)	0.4031 (0.0818)		1.5620 (0.0223)	1.5571 (0.0285)	1.5466 (0.0914)	1.5325 (0.0841)	1.5150 (0.1193)
	K-means	0.3658 (0.0133)	0.3701 (0.0150)	0.3738 (0.0138)	0.3719 (0.0147)	0.3731 (0.0132)		1.5782 (0.0042)	1.5770 (0.0059)	1.5751 (0.0054)	1.5750 (0.0067)	1.5754 (0.0056)
	FAK-means	0.3822 (0.0158)	0.4239 (0.0267)	0.5161 (0.0703)	0.6283 (0.0777)	0.6650 (0.0811)		1.5715 (0.0066)	1.5390 (0.2002)	1.3942 (0.1274)	1.1216 (0.1916)	1.0162 (0.1819)
	BatK-means											
External Metrics	Algorithms	Iteration 1		Iteration 2		Iteration 5		Iteration 10		Iteration 20		
F-measure	WFA <sub>RM</sub>	<b>0.7052</b> (0.0606) [0.6464, 0.7758]		<b>0.6765</b> (0.0182) [0.6130, 0.7547]		<b>0.6676</b> (0.0046) [0.5791, 0.7881]		<b>0.6575</b> (0.0300) [0.5632, 0.7898]		<b>0.7997</b> (0.0129) [0.8047, 0.7948]		
	PSO	0.4947 (0.0188) [0.7007, 0.3823]		0.5062 (0.0414) [0.7348, 0.3861]		0.4907 (0.0118) [0.7184, 0.3726]		0.4951 (0.0110) [0.7357, 0.3731]		0.4986 (0.0237) [0.2493, 0.3772]		
	K-means	0.4910 (0.0213) [0.8405, 0.3468]		0.4935 (0.0138) [0.8658, 0.3451]		0.4975 (0.0196) [0.8417, 0.3531]		0.4999 (0.0316) [0.8385, 0.3561]		0.4954 (0.0081) [0.8725, 0.3459]		
	Bisect K-means	0.4698 (0.0297) [0.6262, 0.3759]		0.4757 (0.0284) [0.6342, 0.3806]		0.4775 (0.0336) [0.6227, 0.3872]		0.4785 (0.0567) [0.6094, 0.3939]		0.4908 (0.0644) [0.6273, 0.4031]		
	FAK-means	0.3656 (0.0134) [0.3654, 0.3658]		0.3692 (0.0140) [0.3683, 0.3701]		0.3747 (0.0155) [0.3756, 0.3738]		0.3723 (0.0143) [0.3727, 0.3719]		0.3737 (0.0132) [0.3743, 0.3731]		
	BatK-means	0.3813 (0.0164) [0.3804, 0.3822]		0.4227 (0.0245) [0.4215, 0.4239]		0.5126 (0.0720) [0.5092, 0.5161]		0.6345 (0.0765) [0.6408, 0.6283]		0.6649 (0.0768) [0.6648, 0.6650]		

Note: the best value is highlighted in 'bold', standard deviation in ( ), average precision and average recall in [ ].



Table 6.8

*Internal and relative quality metrics of clustering and standard deviation: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means.*

Internal & Relative Metrics	Algorithms	Iterations				
		1	2	5	10	20
ADDC	WFA <sub>RM</sub>	1.4158 (0.0138)	1.4058 (0.0067)	1.4153 (0.0083)	1.4180 (0.0072)	1.4263 (0.0018)
	PSO	1.7630 (0.2124)	1.7775 (0.2738)	1.8053 (0.2751)	1.9379 (0.1862)	1.8736 (0.0430)
	K-means	<b>0.7547</b> (0.2978)	<b>0.7041</b> (0.2861)	<b>0.6783</b> (0.2396)	<b>0.6367</b> (0.2511)	<b>0.7056</b> (0.3190)
	Bisect K-means	1.3238 (0.1874)	1.2545 (0.2808)	1.2633 (0.2378)	1.3332 (0.2318)	1.3494 (0.1817)
	FAK-means	1.4434 (0.0010)	1.4436 (0.0007)	1.4436 (0.0009)	1.4432 (0.0009)	1.4436 (0.0007)
	BatK-means	1.4428 (0.0035)	1.4505 (0.0044)	1.4489 (0.0099)	1.4459 (0.0144)	1.4563 (0.0171)
Internal & Relative Metrics	Algorithms	Iterations				
		1	2	5	10	20
DBI	WFA <sub>RM</sub>	7.3303 (0.6042)	7.1238 (0.2775)	7.3802 (0.6346)	7.3298 (0.4761)	9.2225 (0.1518)
	PSO	<b>1.7069</b> (0.0261)	<b>1.6100</b> (0.2168)	<b>1.5559</b> (0.2303)	<b>1.6092</b> (0.1732)	<b>1.6472</b> (0.2269)
	K-means	2.8159 (3.5419)	2.3565 (3.1847)	2.4741 (3.1267)	1.9649 (3.1855)	1.9090 (2.5369)
	Bisect K-means	8.1636 (3.0869)	7.3146 (3.0355)	6.9485 (3.0605)	7.8287 (2.8864)	7.7189 (2.7763)
	FAK-means	14.2277 (0.2063)	14.2834 (0.2025)	14.2549 (0.3551)	14.2637 (0.2862)	14.2158 (0.2801)
	BatK-means	14.2657 (0.3657)	12.3838 (0.4195)	11.3361 (1.2888)	10.7060 (1.4489)	10.9907 (1.5131)
Internal & Relative Metrics	Algorithms	Iterations				
		1	2	5	10	20
DI	WFA <sub>RM</sub>	0.2330 (0.0070)	0.2328 (0.0011)	0.2283 (0.0071)	0.2292 (0.0060)	0.2058 (0.0044)
	PSO	1.0162 (0.0950)	1.0331 (0.0574)	1.0357 (0.0650)	1.0119 (0.0583)	0.9908 (0.0899)
	K-means	<b>2.3413</b> (2.3505)	<b>2.9315</b> (2.4566)	<b>2.9671</b> (2.5302)	<b>3.7363</b> (2.3653)	<b>3.4078</b> (2.4518)
	Bisect K-means	0.2393 (0.1483)	0.2876 (0.2299)	0.2998 (0.1887)	0.2715 (0.2178)	0.2698 (0.2008)
	FAK-means	0.1380 (0.0026)	0.1372 (0.0027)	0.1377 (0.0039)	0.1374 (0.0035)	0.1382 (0.0035)
	BatK-means	0.1369 (0.0043)	0.1560 (0.0075)	0.1698 (0.0212)	0.1775 (0.0254)	0.1721 (0.0256)

*Note: the best value is highlighted in 'bold', standard deviation in (.).*

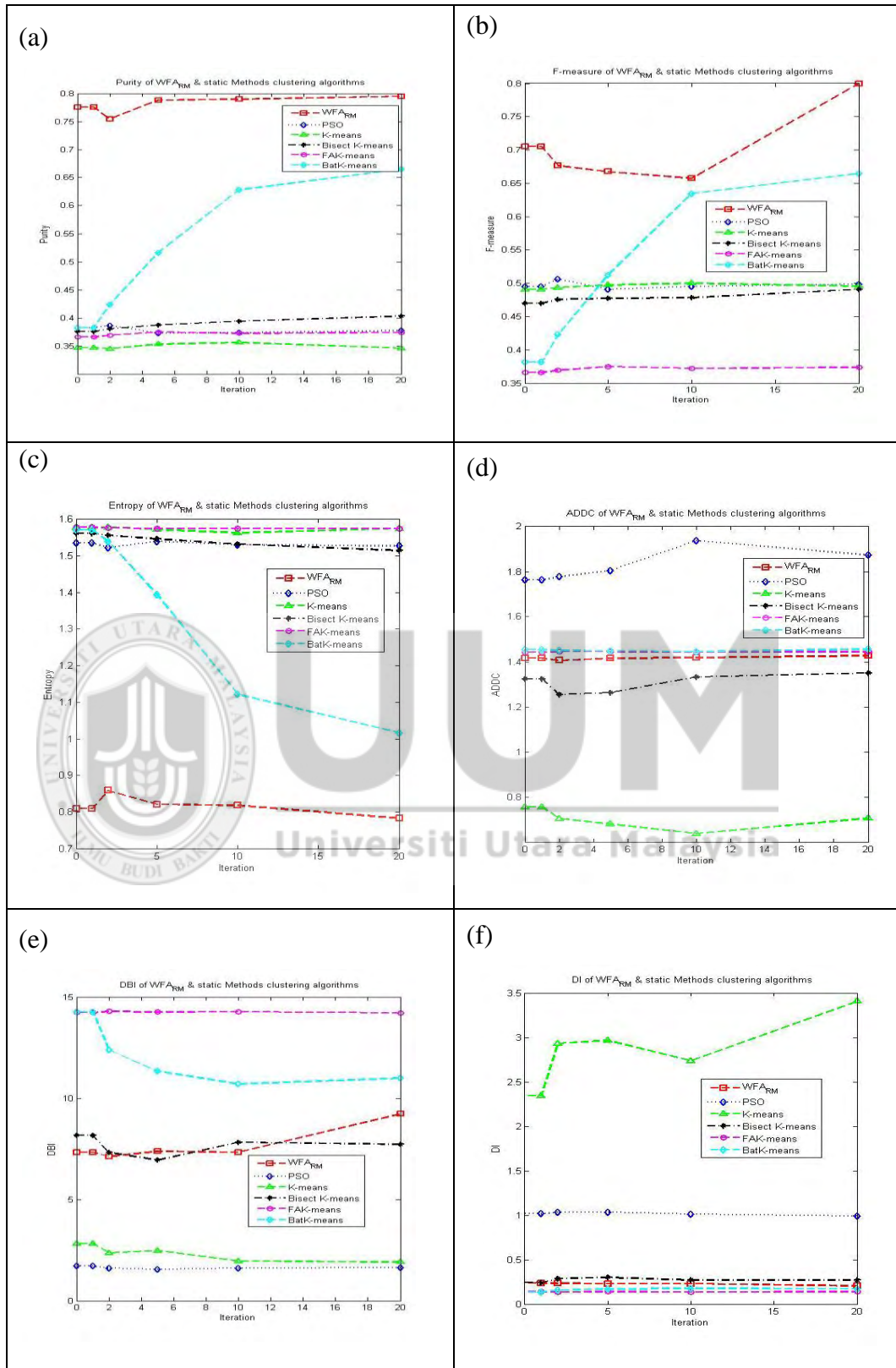


Figure 6.14. Graphical representation of quality metrics: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means (a) Purity, (b) F-measure, (c) Entropy, (d) ADDC, (e) DBI, and (f) DI.

This result indicates that  $WFA_{RM}$  is better than other methods. Figure 6.14.c shows the results of the average Entropy in a graphical representation of the mentioned methods.

As shown in Table 6.8, K-means is better than the others methods to generate a smaller average ADDC in most iterations. Furthermore, it can be noticed that  $WFA_{RM}$  is best as compared to hybrid methods, FAK-means and BatK-mean, in producing a smaller ADDC value, where the best average ADDC is 1.4263 produced by  $WFA_{RM}$  normalized in iteration 20 with the number of clusters 3, while FAK-means generates 1.4436 and BatK-mean generates 1.4563 in the same iteration. This result means that  $WFA_{RM}$  is best to generate the right number of clusters with the highest performance (lower ADDC) against hybrid methods. Figure 6.14.d shows a plotted graph of the average ADDC result of  $WFA_{RM}$ , PSO, K-means, Bisect K-means, FAK-means and BatK-means.

For DBI metrics, in Table 6.8 notices that PSO is best to generate smaller DBI values against other methods, where the best average DBI value is 1.6472 in iteration 20. It can be seen that  $WFA_{RM}$  is best to generate smaller average DBI values against Bisect K-means in most iterations (refer to iterations 1, 2, 5 and 10) and better than hybrid methods, FAK-means and BatK-means, in all iterations, where the best value generated by  $WFA_{RM}$  in iteration 20 is 9.2225, whilst FAK-means is 14.2158 and BatK-means is 10.9907. Figure 6.14.e illustrates the average DBI results of six methods represented in a graphical representation.

For DI metrics, Table 6.8 reports the DI value of six methods in different iterations. As can be seen in the table, K-means is better than other methods in generating the highest DI value in all iterations, followed by PSO, Bisect K-means, and WFA<sub>RM</sub> which is better than FAK-means and BatK-means, where the best average DI is 0.2058 generated by WFA<sub>RM</sub> in iteration 20, while FAK-means generates 0.1382, and BatK-means is 0.1721. Figure 6.14.c illustrates the average DI result in a graphical representation of six methods: WFA<sub>RM</sub>, PSO, K-means, Bisect K-means, FAK-means, and BatK-means.

The previous results indicate that WFA<sub>RM</sub> generates the best quality results in external metrics against all static methods as shown in Table 6.9, while, for the internal and relative metrics, the algorithm is generated best quality results against FAK-means and BatK-means as shown in Table 6.10.

Table 6.9

*Summary of external quality performance results: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means.*

Algorithms	External Metrics		
	Purity	F-measure	Entropy
WFA <sub>RM</sub>	✓	✓	✓
PSO			
K-means			
Bisect K-means			
FAK-means			
BatK-means			

Table 6.10

*Summary of internal and relative quality performance results: WFARM vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means.*

Algorithms	Internal and Relative Metrics		
	ADDC	DBI	DI
WFA <sub>RM</sub>	✓	✓	✓
FAK-means			
BatK-means			

### 6.3.2.3 Independent Samples T-test between WFA<sub>RM</sub> and Static Methods

In Table 6.11, the associated P-value (2-tailed test) is illustrated. Since the P-value between WFA<sub>RM</sub> and any other methods is smaller than (0.05), the null hypothesis is rejected as the mean in any metrics for WFA<sub>RM</sub> and any static methods is the same, and thus, the alternative hypothesis is accepted to conclude that there is a significant difference in the mean of any metric for WFA<sub>RM</sub> and any static methods.

Table 6.11

*The P-value between WFA<sub>RM</sub> & static methods.*

Algorithms	P-value using average Purity (sig 2 tailed)	
	Equal variances assumed	Equal variances not assumed
WFA <sub>RM</sub> and PSO	1.1679E-45	2.2618E-30
WFA <sub>RM</sub> and K-means	1.3371E-17	6.4988E-13
WFA <sub>RM</sub> and Bisect K-means	3.5568E-33	4.3610E-22
WFA <sub>RM</sub> and FAK-means	2.2263E-49	4.9800E-36
WFA <sub>RM</sub> and BatK-means	1.5587E-13	1.4414E-10
	P-value using average F-measure(sig 2 tailed)	
WFA <sub>RM</sub> and PSO	2.3898E-54	1.0052E-44

Table 6.11 continued

WFA <sub>RM</sub> and K-means	8.5237E-69	1.5193E-60
WFA <sub>RM</sub> and Bisect K-means	4.4436E-33	2.0948E-22
WFA <sub>RM</sub> and FAK-means	2.6931E-72	3.5099E-72
WFA <sub>RM</sub> and BatK-means	2.1750E-13	1.2784E-10
<b>P-value using average Entropy (sig 2 tailed)</b>		
WFA <sub>RM</sub> and PSO	8.9082E-52	1.8034E-37
WFA <sub>RM</sub> and K-means	2.2156E-72	3.4567E-70
WFA <sub>RM</sub> and Bisect K-means	2.0342E-37	5.8592E-25
WFA <sub>RM</sub> and FAK-means	2.8653E-78	1.8479E-47
WFA <sub>RM</sub> and BatK-means	3.4760E-09	9.5530E-08
<b>P-value using average ADDC (sig 2 tailed)</b>		
WFA <sub>RM</sub> and PSO	3.6977E-19	8.2027E-14
WFA <sub>RM</sub> and K-means	1.3371E-17	6.4988E-13
WFA <sub>RM</sub> and Bisect K-means	0.0313400	0.0354400
WFA <sub>RM</sub> and FAK-means	2.2263E-49	4.9800E-36
WFA <sub>RM</sub> and BatK-means	1.5587E-13	1.4414E-10
<b>P-value using average DBI (sig 2 tailed)</b>		
WFA <sub>RM</sub> and PSO	3.8237E-77	5.1042E-69
WFA <sub>RM</sub> and K-means	2.7754E-22	1.2587E-15
WFA <sub>RM</sub> and Bisect K-means	0.0050800	0.00681500
WFA <sub>RM</sub> and FAK-means	2.7274E-62	3.3236E-50
WFA <sub>RM</sub> and BatK-means	3.3339E-8	5.3186E-7
<b>P-value using average DI (sig 2 tailed)</b>		
WFA <sub>RM</sub> and PSO	2.9919E-48	3.1977E-29
WFA <sub>RM</sub> and K-means	2.5542E-9	9.7372E-8
WFA <sub>RM</sub> and Bisect K-means	<b>0.0918500</b>	<b>0.0971700</b>
WFA <sub>RM</sub> and FAK-means	4.6142E-56	5.6757E-54
WFA <sub>RM</sub> and BatK-means	2.0596E-9	6.0452E-8

*Hint: The value highlighted in bold indicates not significance.*

The P-value between WFA<sub>RM</sub> and Bisect K-means (bold value in Table 6.11), which is (0.09185 and 0.09717), is larger than (0.05), which means the mean DI metric for

WFA<sub>RM</sub> and Bisect K-means method are the same (accepting the null hypothesis in this situation).

### 6.3.3 Comparison between WFA<sub>RM</sub> and Dynamic Methods

This section includes the evaluation of WFA<sub>RM</sub> against two dynamic methods: Practical General Stochastic Clustering Method (PGSCM) (Tan, Ting, & Teng, 2011a), and Dynamic Hybrid Genetic algorithm with Particle Swarm Optimization (DCPG) (Kuo, Syu, Chen, & Tien, 2012). The evaluation process is conducted in three parts: the comparison of the number of generated clusters in the methods, the comparison between WFA<sub>RM</sub> with PGSCM and DCPG by performance metrics such as Purity, F-measure, Entropy, ADDC, DBI, and DI, and the last evaluation part is the comparison between WFA<sub>RM</sub> with PGSCM and DCPG by statistical approach (Independent samples T-test).

#### 6.3.3.1 Number of Clusters between WFA<sub>RM</sub> and Dynamic Methods

Table 6.12 displays the average number of clusters automatically generated by WFA<sub>RM</sub>, PGSCM and DCPG without any information support about the dataset.

Table 6.12

*Average number of clusters: WFA<sub>RM</sub> vs. PGSCM vs. DCPG.*

Average number of clusters	Algorithms		
	WFA <sub>RM</sub>	PGSCM	DCPG
	<b>3</b>	6.13 $\approx$ 6	8.27 $\approx$ 8

*Note: the best value is highlighted in 'bold'*

As can be seen in Table 6.12, the average number of clusters produced by WFA<sub>RM</sub> in iteration 20 is near the natural number of clusters in other algorithms, while PGSCM produces 6.13 and DCPG produces 8.27. This result means that WFA<sub>RM</sub> is better than PGSCM and DCPG.

### 6.3.3.2 Performance Metrics between WFA<sub>RM</sub> and Dynamic Methods

In this section, WFA<sub>RM</sub> is evaluated against two dynamic methods named PGSCM and DCPG. The evaluation is carried using external quality metrics such as Purity, F-measure, and Entropy, and using internal and relative quality metrics such as ADDC, DBI, and DI. Table 6.13 shows the external quality metrics of clustering.

Table 6.13

*External quality metrics of clustering and standard deviation: WFA<sub>RM</sub> vs. PGSCM vs. DCPG.*

External Metrics	Metrics	Algorithms		
		WFA <sub>RM</sub>	PGSCM	DCPG
	Purity	<b>0.7948</b> (0.0132)	0.3948 (0.0172)	0.4243 (0.0783)
	F-measure	<b>0.7997</b> (0.0129) [0.8047, 0.7948]	0.3406 (0.0421) [0.2995, 0.3948]	0.5003 (0.5003) [0.6095, 0.4243]
	Entropy	<b>0.7827</b> (0.0267)	1.5584 (0.0122)	1.4532 (0.1016)

*Note: the best value is highlighted in 'bold', standard deviation in (), average precision and average recall in [].*

As shown in Table 6.13, the WFA<sub>RM</sub> algorithm generates the highest average purity compared with PGSCM and DCPG. The best average purity value is 0.7948 generated by WFA<sub>RM</sub> in iteration 20, while DCPG generates 0.4243 in the same iteration and PGSCM generates the worst value 0.3948. The overall standard deviation of the WFA<sub>RM</sub> algorithm in iteration 20 is smaller than PGSCM and



DCPG, which means that the solution generated by  $WFA_{RM}$  is more reliable and high in robustness. Figure 6.15 shows a plotted graph of the external quality metrics: average purity, average F-measure, and average Entropy results of  $WFA_{RM}$ , PGSCM and DCPG.

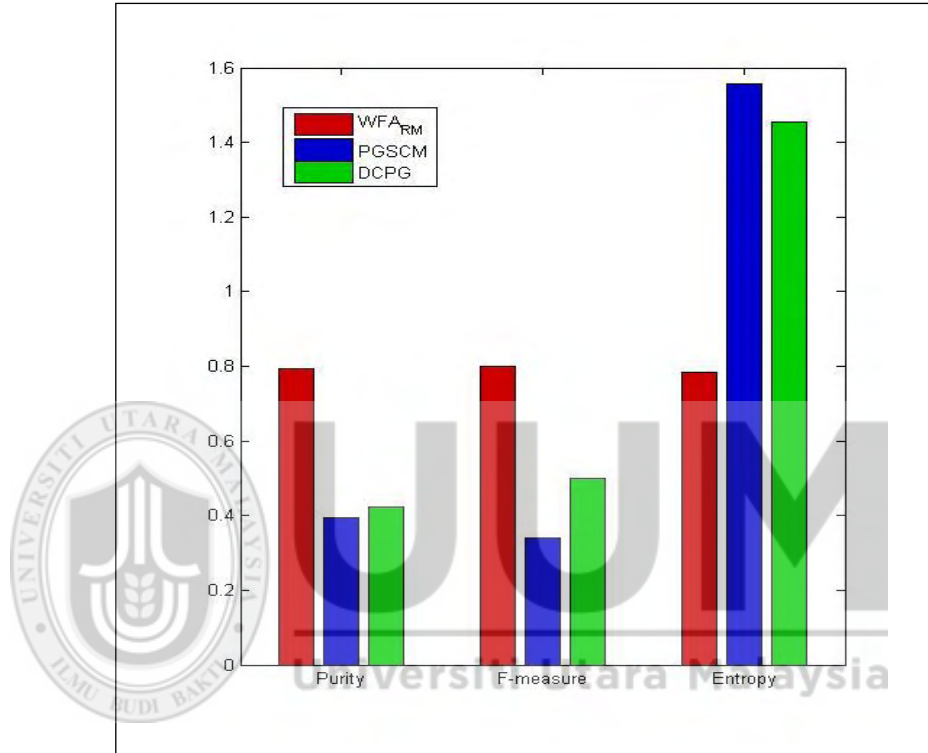


Figure 6.15. External quality metrics:  $WFA_{RM}$  vs. PGSCM vs. DCPG

For F-measure metric, it can be observed in Table 6.13 that  $WFA_{RM}$  generates the highest average F-measure of 0.7997, higher average precision of 0.8047, and higher average recall of 0.7948 against PGSCM and DCPG, where they generate an average F-measure of 0.3406 and 0.5003, respectively. The overall standard deviation of the  $WFA_{RM}$  algorithm is 0.0129, smaller than PGSCM which is 0.0421 and smaller than DCPG which is 0.5003. It can be concluded that the solution generated by  $WFA_{RM}$  is more reliable and high in robustness.

Furthermore, in Table 6.13, it can be seen that the Entropy of  $WFA_{RM}$  is better (smaller) than PGSCM and DCPG, where the best average Entropy is 0.7827 outputted by  $WFA_{RM}$ , whereas, PGSCM and DCPG produced 1.5584 and 1.4532, respectively. The highest value of Purity and F-measure and the lower value of Entropy means the best clustering algorithm (Forsati, Mahdavi, Shamsfard, & Meybodi, 2013; Murugesan & Zhang, 2011a, 2011b); and it can be concluded from the previous result that  $WFA_{RM}$  is the best as compared to PGSCM and DCPG in external quality metrics.

Table 6.14 includes the internal and relative quality metrics for the three methods:  $WFA_{RM}$ , PGSCM, and DCPG.

Table 6.14

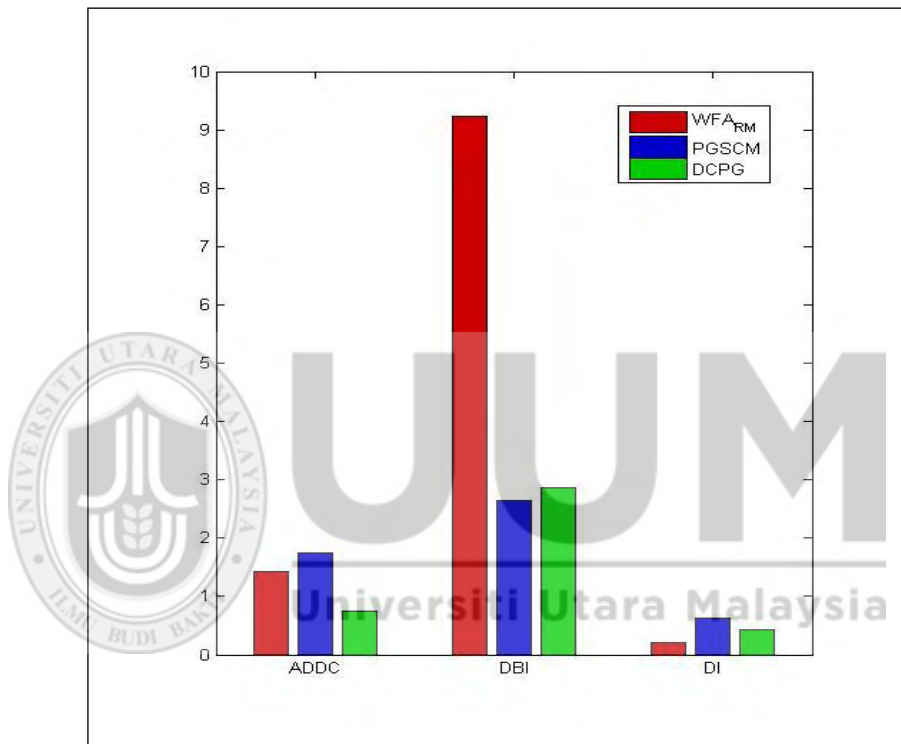
*Internal and relative quality metrics of clustering and standard deviation:  $WFA_{RM}$  vs. PGSCM vs. DCPG.*

Internal & Relative Metrics	Metrics	Algorithms		
		$WFA_{RM}$	PGSCM	DCPG
	ADDC	1.4263 (0.0018)	1.7440 (0.0159)	<b>0.7459</b> (0.2123)
	DBI	9.2225 (0.1518)	<b>2.6506</b> (0.1160)	2.8584 (1.4431)
	DI	0.2058 (0.0044)	<b>0.6302</b> (0.0455)	0.4328 (0.2085)

*Note: the best value is highlighted in 'bold', standard deviation in ().*

As shown in Table 6.14, the  $WFA_{RM}$  algorithm generates a lower average ADDC compared to PGSCM, while DCPG produces a lower average ADDC compared against  $WFA_{RM}$  and PGSCM. The best average ADDC value is 0.7459 generated by DCPG, followed by  $WFA_{RM}$  which produces 1.4263 in iteration 20, followed by PGSCM which generates the worst value of 1.7440. The overall standard deviation

of the  $WFA_{RM}$  algorithm is 0.0018, smaller than PGSCM and DCPG, which they have a standard deviation of 0.0159 and 0.2123, respectively; this result indicates that the solution generated by  $WFA_{RM}$  is more reliable and high in robustness. Figure 6.16 shows a plotted graph of the internal and relative quality metrics, average ADDC, average DBI and average DI results, of  $WFA_{RM}$ , PGSCM and DCPG.



*Figure 6.16.* Internal and relative quality metrics:  $WFA_{RM}$  vs. PGSCM vs. DCPG

For DBI quality metric, in Table 6.14, it is noticed that PGSCM is the best to generate a smaller DBI value against  $WFA_{RM}$  and DCPG methods, where the best average DBI value is 2.6506 outputted by PGSCM, while  $WFA_{RM}$  produces the worst value of 9.2225 and DCPG produces 2.8584. The overall standard deviation of the  $WFA_{RM}$  algorithm is 0.1518 smaller than DCPG which has a standard deviation of 1.4431; however, PGSCM has the smallest standard deviation of 0.1160 than

WFA<sub>RM</sub> and DCPG. This result indicates that PGSCM is more accurate despite of the number of generated clusters is 6.13, which is higher than the actual number of clusters.

As can be seen in Table 6.14, the average DI value for PGSCM is 0.6302. It is larger than the DI value of WFA<sub>RM</sub> which is 0.2058 and the DI value of DCPG which is 0.4328. The overall standard deviation of the WFA<sub>RM</sub> algorithm is 0.0044, smaller than DCPG which has a standard deviation of 0.2085 and PGSCM has a standard deviation of 0.0455. Lower ADDC and DBI values and larger DI value indicates the best clustering algorithm.

The previous results indicate that WFA<sub>RM</sub> generates the best quality results in external performance metrics, Purity, F-measure and Entropy, against all dynamic methods, PGSCM and DCPG, as shown in Table 6.15. While, for the internal and relative metrics of ADDC, DBI and DI, the DCPG generates the best ADDC quality results and PGSCM generates the best DBI and DI.

Table 6.15

*Summary of quality performance results: WFA<sub>RM</sub> vs. PGSCM vs. DCPG.*

Algorithms	External Metrics			Internal and Relative Metrics		
	Purity	F-measure	Entropy	ADDC	DBI	DI
WFA <sub>RM</sub>	✓	✓	✓			
PGSCM					✓	✓
DCPG				✓		

### 6.3.3.3 Independent Samples T-test between $WFA_{RM}$ and Dynamic Methods

This section includes the analysis of Independent Samples T-test between  $WFA_{RM}$  and PGSCM, and between  $WFA_{RM}$  and DCPG methods as shown in Table 6.16.

Table 6.16

*The P-value between  $WFA_{RM}$  & dynamic methods.*

Algorithms	P-value using average Purity (sig 2 tailed)	
	Equal variances	Equal variances not
	assumed	assumed
$WFA_{RM}$ and PGSCM	6.0521E-67	1.5895E-63
$WFA_{RM}$ and DCPG	2.9644E-33	3.2203E-22
Algorithms	P-value using average F-measure(sig 2 tailed)	
	Equal variances	Equal variances not
	assumed	assumed
$WFA_{RM}$ and PGSCM	1.1457E-52	1.2077E-35
$WFA_{RM}$ and DCPG	1.9432E-46	1.4369E-33
Algorithms	P-value using average Entropy (sig 2 tailed)	
	Equal variances	Equal variances not
	assumed	assumed
$WFA_{RM}$ and PGSCM	6.9327E-76	9.2948E-57
$WFA_{RM}$ and DCPG	1.1415E-40	1.1919E-27
Algorithms	P-value using average ADDC (sig 2 tailed)	
	Equal variances	Equal variances not
	assumed	assumed
$WFA_{RM}$ and PGSCM	1.0321E-68	3.3946E-40
$WFA_{RM}$ and DCPG	7.1148E-25	5.4648E-17
Algorithms	P-value using average DBI (sig 2 tailed)	
	Equal variances	Equal variances not
	assumed	assumed
$WFA_{RM}$ and PGSCM	1.5079E-82	4.3476E-78
$WFA_{RM}$ and DCPG	7.8719E-32	5.5106E-21
Algorithms	P-value using average DI (sig 2 tailed)	
	Equal variances	Equal variances not
	assumed	assumed
$WFA_{RM}$ and PGSCM	7.8781E-50	2.4346E-30
$WFA_{RM}$ and DCPG	1.5863E-07	1.7713E-06

The hypotheses for Independent Samples T-test can be expressed in mathematical equivalents.

*Null hypothesis  $H_0$ :  $M(WFA_{RM}) = M(\text{any dynamic methods})$*

*Alternative hypothesis  $H_1$ :  $M(WFA_{RM}) \neq M(\text{any dynamic methods})$*

Where,  $M(WFA_{RM})$ ,  $M(\text{any dynamic methods})$  are the means of the population for  $WFA_{RM}$  method and any dynamic method.

In Table 6.16, the associated P-value (2-tailed test) is reported. Since the P-value between  $WFA_{RM}$  and PGSCM, and between  $WFA_{RM}$  and DCPG methods is smaller than (0.05), the null hypothesis is rejected so that the mean of any metrics for  $WFA_{RM}$  and any dynamic methods are the same and the alternative hypothesis is accepted to conclude that there is a significant difference in the mean of any metric for  $WFA_{RM}$  and any dynamic methods.

#### **6.4 Summary**

This chapter presented the proposed Adaptive Firefly Algorithm for text clustering, termed as  $WFA_{RM}$ . The aim of incorporating the clusters merging procedure is to minimize the number of clusters produced by  $WFA_R$ . The cluster merging algorithm includes two steps: merge clusters and refine merged clusters.

Three experiments are conducted in this chapter: 1) Comparison between  $WFA_{RM}$  and  $WFA_R$ , 2) Comparison between the proposed  $WFA_{RM}$  and statics clustering methods such as K-means, PSO, Bisect K-means, FAK-means and BatK-means, 3) The performance of the proposed  $WFA_{RM}$  against dynamic clustering methods such as DCPG and PGSCM.

In addition, the analyses of paired samples T-test between  $WFA_{RM}$  with  $WFA_R$  and Independent Samples T-test between  $WFA_{RM}$  and state-of-the-art methods are also performed.



## **CHAPTER SEVEN**

### **EVALUATION OF ADAPTIVE FA ON VARIOUS DATASETS**

#### **7.1 Introduction**

This chapter includes the evaluation and analysis of the proposed adaptive firefly algorithm, termed as WFA<sub>RM</sub>. The experiments were conducted on datasets with different sizes (balanced and un-balanced).

A comparison is made between WFA<sub>RM</sub> and state-of-the-art methods (static and dynamic methods). Each comparison consists of three evaluation parts: the evaluation of produced number of clusters, the evaluation of performance metrics, namely external, internal and relative metrics such as Purity, F-measure, Entropy, ADDC, DBI and DI, and the third part of evaluation is a statistical analysis of Independent Samples T-test (Ross, 2010).

#### **7.2 Comparison WFA<sub>RM</sub> with Static Methods**

This section involves a comparison between the WFA<sub>RM</sub> algorithm with some state-of-the-art methods (static methods), Particle Swarm Optimization (PSO), K-means, Bisect K-means, Hybrid Firefly algorithm with K-means (FAK-means), and Hybrid Bat algorithm with K-means (BatK-means). The comparison includes the evaluation of produced number of clusters, the evaluation using performance external, internal and relative metrics such as Purity, F-measure, Entropy, ADDC, DBI and DI, and the evaluation using a statistical analysis of Independent Samples T-test (Ross, 2010) that performs on the differences between the pairs, WFA<sub>RM</sub> and other methods (static methods).



### 7.2.1 Evaluation Number of Clusters between WFA<sub>RM</sub> and Static Methods

The purpose of this comparison is to examine the k number of cluster automatically generated by the proposed WFA<sub>RM</sub> algorithm, and compare it with other static methods that provide the number of k cluster. Table 7.1 displays the average number of clusters automatically generated by WFA<sub>RM</sub> (set the number of iterations equal to 20) using different datasets.

Table 7.1

*Average numbers of clusters: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means using different datasets.*

Datasets	Number of clusters of algorithms					
	WFA <sub>RM</sub>	PSO	K-means	Bisect K-means	FAK-means	BatK-means
Reuters (300 documents and 6 classes)	6	6	6	6	6	6
TR11 (414 documents and 9 classes)	9.27 $\approx$ 9	9	9	9	9	9
TR12 (313 documents and 8 classes)	8	8	8	8	8	8
TR23 (204 documents and 6 classes)	6.07 $\approx$ 6	6	6	6	6	6
TR45 (690 documents and 10 classes)	9.83 $\approx$ 10	10	10	10	10	10

As can be seen in Table 7.1, the number of clusters produced by the proposed WFA<sub>RM</sub> is 6 using the Reuters dataset, is 9.27 using TR11, is 8 using TR12, is 6.07 using TR23, and is 9.83 using TR45. The percentage of success to obtain a near

natural number of clusters in the proposed  $WFA_{RM}$  is 100 %. Figure 7.1 shows the results of the number of obtained clusters by  $WFA_{RM}$  and the real number of clusters of all static methods.

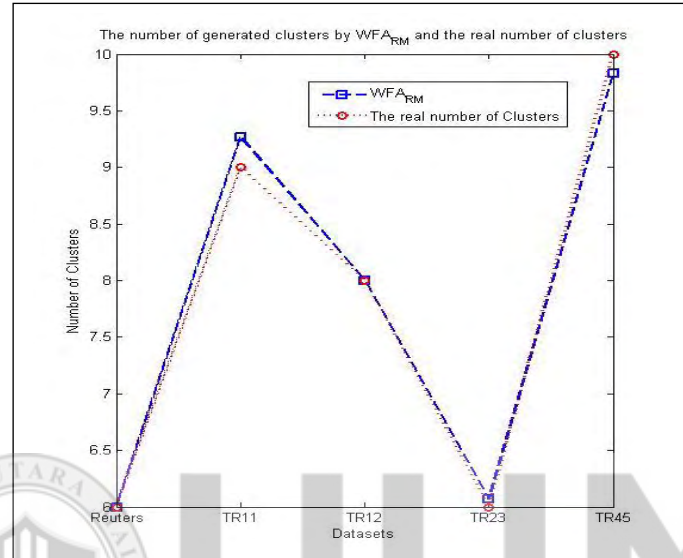


Figure 7.1. Results of the number of generated clusters by  $WFA_{RM}$  and the real number of clusters of all static methods

### 7.2.2 Evaluation of Performance Metrics between $WFA_{RM}$ and Static Methods

In this section, the performance results of  $WFA_{RM}$  are compared against two types of static methods; single methods such as Particle Swarm Optimization (PSO) (Cui, Potok, & Palathingal, 2005), K-means (Jain, 2010), and Bisect K-means (Murugesan & Zhang, 2011a, 2011b), and hybrid methods are such as Hybrid Firefly algorithm with K-means (FAK-means) (Tang, Fong, Yang, & Deb, 2012), and Hybrid Bat algorithm with K-means (BatK-means) (Tang, Fong, Yang, & Deb, 2012). The purpose of this comparison is to investigate the effectiveness of the non-determined k number of cluster in the  $WFA_{RM}$  algorithm in producing quality clusters, even though it has not been provided the same support as the other algorithms that require

Table 7.2

*External quality Purity (average, best, worst, standard deviation): WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets (balanced and un-balanced datasets).*

Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	<b>0.7351</b>	0.7433	0.7300	0.0051
	PSO	0.3389	0.5033	0.2167	0.0791
	K-means	0.3270	0.5267	0.2000	0.0692
	Bisect K-means	0.4073	0.6033	0.2100	0.1065
	FAK-means	0.2361	0.2567	0.2200	0.0098
	BatK-means	0.6514	0.7733	0.5200	0.0667
Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	<b>0.6810</b>	0.7246	0.6594	0.0224
	PSO	0.4159	0.5290	0.3237	0.0622
	K-means	0.3860	0.5290	0.3188	0.0722
	Bisect K-means	0.5333	0.7778	0.3720	0.1004
	FAK-means	0.3239	0.3430	0.3188	0.0060
	BatK-means	0.6657	0.7899	0.3961	0.0793
Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	0.4752	0.4760	0.4696	0.0022
	PSO	0.3891	0.5911	0.3163	0.0672
	K-means	0.4370	0.6038	0.3163	0.0827
	Bisect K-means	0.4850	0.5847	0.3195	0.0629
	FAK-means	0.3013	0.3259	0.2971	0.0065
	BatK-means	<b>0.5808</b>	0.6997	0.4760	0.0639
Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	<b>0.6266</b>	0.6324	0.6176	0.0032
	PSO	0.5444	0.6569	0.4608	0.0503
	K-means	0.5451	0.6471	0.4657	0.0559
	Bisect K-means	0.5159	0.5441	0.4657	0.0239
	FAK-means	0.4495	0.4706	0.4461	0.0072
	BatK-means	0.5956	0.7059	0.5245	0.0524
Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	<b>0.6355</b>	0.6609	0.5812	0.0161
	PSO	0.3277	0.4986	0.2406	0.0592
	K-means	0.4416	0.6348	0.2580	0.0988
	Bisect K-means	0.4280	0.6609	0.2449	0.1059
	FAK-means	0.2422	0.2580	0.2348	0.0056
	BatK-means	0.6083	0.7594	0.4580	0.0789

*Note: the best value is highlighted in 'bold'.*

a predefined number of k clusters.

As shown in Table 7.2, the WFA<sub>RM</sub> algorithm generates the highest average purity in most datasets, balanced and un-balanced datasets (refer to Reuters, TR11, TR23 and TR45) compared to other algorithms, where the average purity value is 0.7351 generated by WFA<sub>RM</sub> in the Reuters dataset, is 0.6810 in TR11 dataset, is 0.6266 in TR23 dataset, and is 0.6355 in TR45 dataset, while, BatK-means generates the highest average purity of 0.5808 in the TR12 dataset. In addition, the overall standard deviation of the WFA<sub>RM</sub> algorithm is smaller than other methods in most datasets. This result indicates that the solution generated by WFA<sub>RM</sub> is more reliable and high in robustness. The percentage of success to obtain the highest purity is 80% by WFA<sub>RM</sub>. Figure 7.2 illustrates the purity result in a graphical representation of the methods: WFA<sub>RM</sub>, PSO, K-means, Bisect K-means, FAK-means, and BatK-means.

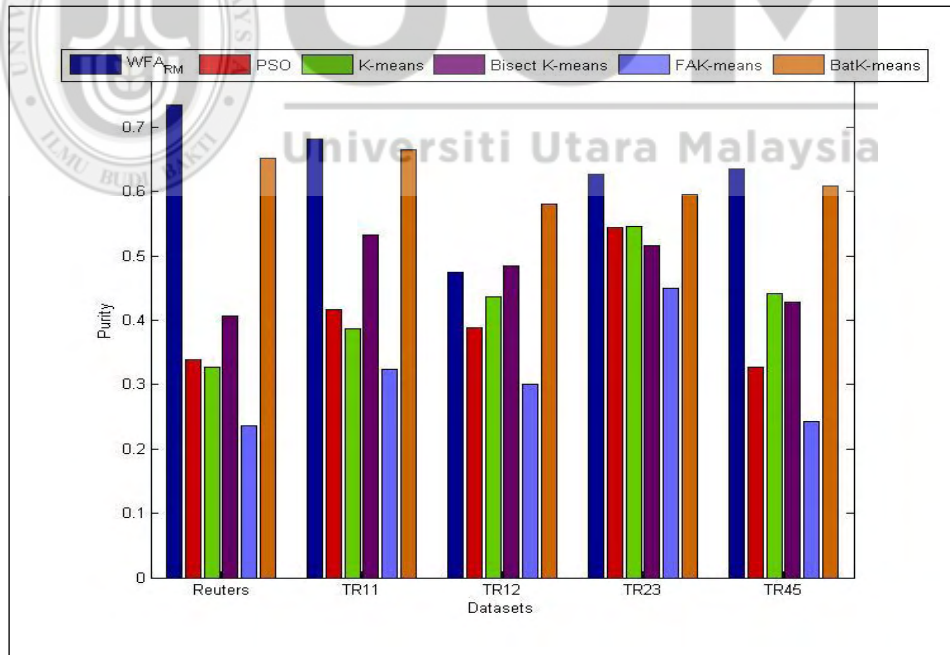


Figure 7.2. Average Purity results: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets

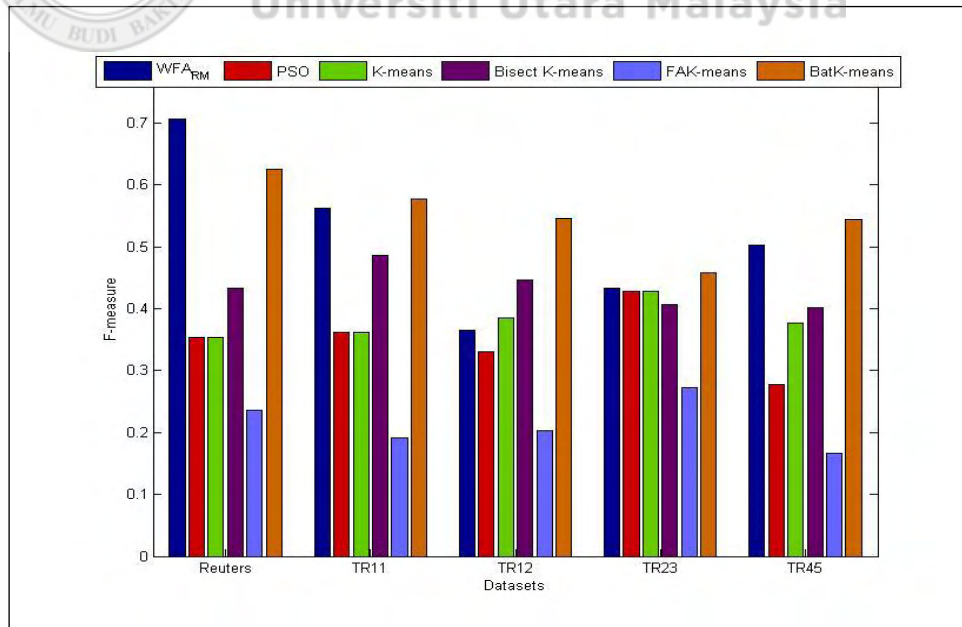
Table 7.3

*External quality F-measure (average, best, worst, standard deviation): WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets.*

Datasets	Algorithms	Average F-measure [precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	<b>0.7069</b> [0.6808, 0.7351]	0.7138	0.7022	0.0045
	PSO	0.3536 [0.3696, 0.3389]	0.5152	0.2766	0.0720
	K-means	0.3531 [0.3837, 0.3270]	0.4870	0.2770	0.0705
	Bisect K-means	0.4339 [0.4642, 0.4073]	0.6703	0.2749	0.1048
	FAK-means	0.2357 [0.2353, 0.2361]	0.2569	0.2095	0.0108
	BatK-means	0.6260 [0.6025, 0.6514]	0.7780	0.5049	0.0722
Datasets	Algorithms	Average F-measure [precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	0.5631 [0.4800, 0.6810]	0.6547	0.4615	0.0520
	PSO	0.3620 [0.3205, 0.4159]	0.4906	0.2723	0.0651
	K-means	0.3627 [0.3421, 0.3860]	0.5273	0.2990	0.0769
	Bisect K-means	0.4869 [0.4479, 0.5333]	0.7366	0.2677	0.1069
	FAK-means	0.1916 [0.1360, 0.3239]	0.2119	0.1731	0.0098
	BatK-means	<b>0.5780</b> [0.5107, 0.6657]	0.7509	0.3225	0.0884
Datasets	Algorithms	Average F-measure [precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	0.3649 [0.2962, 0.4752]	0.3676	0.3467	0.0071
	PSO	0.3305 [0.2872, 0.3891]	0.4953	0.2608	0.0630
	K-means	0.3856 [0.3450, 0.4370]	0.5861	0.2458	0.0954
	Bisect K-means	0.4468 [0.4142, 0.4850]	0.5630	0.3161	0.0651
	FAK-means	0.2037 [0.1539, 0.3013]	0.2233	0.1833	0.0104
	BatK-means	<b>0.5463</b> [0.5157, 0.5808]	0.6750	0.4270	0.0714
Datasets	Algorithms	Average F-measure [precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	0.4337 [0.3316, 0.6266]	0.4460	0.4024	0.0089
	PSO	0.4277 [0.3522, 0.5444]	0.6158	0.3439	0.0678
	K-means	0.4286 [0.3531, 0.5451]	0.5682	0.3436	0.0575
	Bisect K-means	0.4068 [0.3358, 0.5159]	0.4544	0.3749	0.0210
	FAK-means	0.2727 [0.1957, 0.4495]	0.3377	0.2398	0.0199
	BatK-means	<b>0.4573</b> [0.3711, 0.5956]	0.5890	0.3275	0.0699
Datasets	Algorithms	Average F-measure [precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	0.5031 [0.4164, 0.6355]	0.5489	0.4314	0.0227
	PSO	0.2772 [0.2402, 0.3277]	0.4063	0.2246	0.0539
	K-means	0.3772 [0.3292, 0.4416]	0.5845	0.2186	0.1046
	Bisect K-means	0.4017 [0.3785, 0.4280]	0.6224	0.2394	0.1064
	FAK-means	0.1658 [0.1260, 0.2422]	0.1825	0.1560	0.0080
	BatK-means	<b>0.5448</b> [0.4933, 0.6083]	0.7059	0.3445	0.0893

*Note: the best value is highlighted in 'bold', average precision and average recall in [ ].*

For F-measure metric, as can be seen in Table 7.3, the WFA<sub>RM</sub> algorithm generates the highest average F-measure (higher precision and recall) in the balanced dataset (Reuters dataset) compared against other methods. In the TR11, TR23 and TR45 datasets, WFA<sub>RM</sub> generates the highest F-measure (higher precision and recall) compared to all methods excluding the BatK-means method that generates the highest F-measure in most datasets (refer to TR11, TR12, TR23 and TR45). The overall standard deviation of the WFA<sub>RM</sub> algorithm is smaller than other methods in most datasets, which indicates that the solution generated by WFA<sub>RM</sub> is more reliable and high in robustness. The percentage of success to obtain the highest F-measure is 20% by WFA<sub>RM</sub> against the BatK-means method and 80% against other methods. Figure 7.3 demonstrates the result of the average F-measure in a pictorial representation of WFA<sub>RM</sub>, PSO, K-means, Bisect K-means, FAK-means, and BatK-means.



*Figure 7.3.* Average F-measure result: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets

Table 7.4

*External quality Entropy (average, best, worst, standard deviation): WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means using different datasets.*

Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	<b>1.0420</b>	1.0044	1.0653	0.0258
	PSO	2.1709	1.8116	2.4480	0.1797
	K-means	2.0872	1.4561	2.5022	0.2450
	Bisect K-means	1.8843	1.2484	2.4687	0.3393
	FAK-means	2.5205	2.4831	2.5500	0.0161
	BatK-means	1.2260	0.7787	1.6723	0.2145
Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	<b>1.2584</b>	1.0489	1.3384	0.0798
	PSO	2.2701	1.9254	2.5870	0.1948
	K-means	2.3809	1.7211	2.7022	0.3310
	Bisect K-means	1.7906	0.9504	2.4933	0.3271
	FAK-means	2.5788	2.5465	2.6085	0.0166
	BatK-means	1.3774	0.9481	2.0766	0.2525
Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	1.9200	1.9154	1.9531	0.0119
	PSO	2.3761	1.7145	2.6743	0.2435
	K-means	2.1587	1.4495	2.6577	0.3273
	Bisect K-means	1.9298	1.4824	2.4222	0.2317
	FAK-means	2.6492	2.6103	2.6984	0.0238
	BatK-means	<b>1.5966</b>	1.3047	1.9511	0.2124
Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	<b>1.4980</b>	1.4295	1.5306	0.0198
	PSO	1.7316	1.3594	1.9449	0.1332
	K-means	1.7211	1.3142	1.9014	0.1445
	Bisect K-means	1.8156	1.7402	1.8938	0.0706
	FAK-means	1.9890	1.9224	2.0324	0.0254
	BatK-means	1.5146	1.2221	1.7800	0.1656
Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	1.6742	1.5923	1.7450	0.0434
	PSO	2.6552	2.1542	2.9531	0.2053
	K-means	2.2262	1.4110	2.9282	0.4016
	Bisect K-means	2.2180	1.5735	2.9614	0.3370
	FAK-means	2.9417	2.9162	2.9650	0.0124
	BatK-means	<b>1.6049</b>	1.1448	2.1046	0.2625

*Note: the best value is highlighted in 'bold'.*

In Table 7.4, it is reported the average Entropy value, best Entropy value, worst Entropy value and standard deviation of thirty executions for WFA<sub>RM</sub>, PSO, K-means, Bisect K-means, FAK-means, and BatK-means. As shown in the table, the average Entropy of WFA<sub>RM</sub> is smaller than other methods in most datasets (refer to Reuters, TR11 and TR23), while BatK-means generates the best Entropy (minimum value) in the TR12 and TR45 datasets. The overall standard deviation of the WFA<sub>RM</sub> algorithm is smaller than other methods in most datasets (refer to TR12 and TR23) which means the solution generated by WFA<sub>RM</sub> is more reliable and high in robustness. The percentage of success to obtain a lower Entropy is 60% by WFA<sub>RM</sub> against BatK-means method and 100% against other methods. Figure 7.4 shows the result of the average Entropy in a graphical representation of WFA<sub>RM</sub>, PSO, K-means, Bisect K-means, FAK-means, and BatK-means.

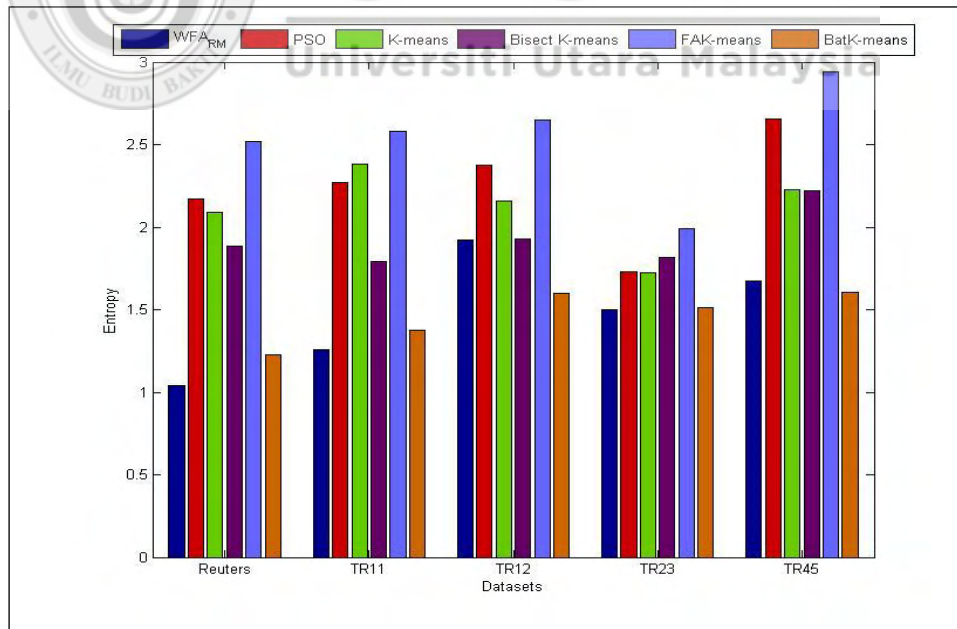


Figure 7.4. Average Entropy result: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets.



Table 7.5

*Internal quality ADDC (average, best, worst, standard deviation): WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means using different datasets.*

Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	1.1786	1.1737	1.1807	0.0027
	PSO	1.3637	0.9185	1.5835	0.1775
	K-means	<b>0.7317</b>	0.3505	1.1354	0.1769
	Bisect K-means	0.9847	0.6169	1.3227	0.1423
	FAK-means	1.2206	1.2137	1.2240	0.0023
	BatK-means	1.3018	1.1162	1.3884	0.0517
Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	0.8293	0.8198	0.8529	0.0088
	PSO	0.7293	0.4713	0.9598	0.1301
	K-means	<b>0.5355</b>	0.0977	0.9981	0.3026
	Bisect K-means	0.7109	0.4604	0.8667	0.0935
	FAK-means	0.8709	0.8677	0.8728	0.0012
	BatK-means	1.0203	0.7509	1.1085	0.0680
Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	0.7928	0.7821	0.7942	0.0033
	PSO	0.7366	0.356	1.0787	0.1902
	K-means	<b>0.5478</b>	0.3172	0.7796	0.1244
	Bisect K-means	0.7264	0.4565	0.9008	0.1231
	FAK-means	0.8820	0.8785	0.8850	0.0012
	BatK-means	0.9822	0.7064	1.1904	0.1190
Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	0.6527	0.6407	0.6559	0.0030
	PSO	0.7542	0.5855	0.9681	0.0966
	K-means	<b>0.4335</b>	0.2660	0.6187	0.0795
	Bisect K-means	0.5079	0.4166	0.7092	0.0644
	FAK-means	0.7406	0.7346	0.7447	0.0020
	BatK-means	0.8897	0.6039	1.0372	0.1305
Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	0.9448	0.9284	0.9503	0.0068
	PSO	0.8453	0.4458	1.1363	0.1599
	K-means	<b>0.6297</b>	0.3120	0.8525	0.1209
	Bisect K-means	0.8030	0.6266	1.0189	0.1169
	FAK-means	0.9993	0.9971	1.0013	0.0009
	BatK-means	1.1668	0.8785	1.3934	0.1270

*Note: the best value is highlighted in 'bold'.*

Table 7.5 includes the quality performance results of internal metric ADDC for six algorithms, namely  $WFA_{RM}$ , PSO, K-means, Bisect K-means, FAK-means, and BatK-means. All algorithms are implemented in the same environment and are run thirty times. As shown in the table, K-means is better than the others methods in generating the smaller average ADDC in all datasets, while the proposed  $WFA_{RM}$  generates a smaller ADDC against FAK-means and BatK-means in all types of dataset and a smaller ADDC against PSO only in the Reuters and TR23 datasets. This result means that  $WFA_{RM}$  is best to generate the right number of clusters with the highest performance (lower ADDC) against hybrid methods (refer to FAK-means and BatK-means). The percentage of success to obtain a lower ADDC is 100% by  $WFA_{RM}$  against hybrid methods, and 40% against the PSO method. Figure 7.5 shows a plotted graph of the average ADDC result for  $WFA_{RM}$ , PSO, K-means, Bisect K-means, FAK-means, and BatK-means.

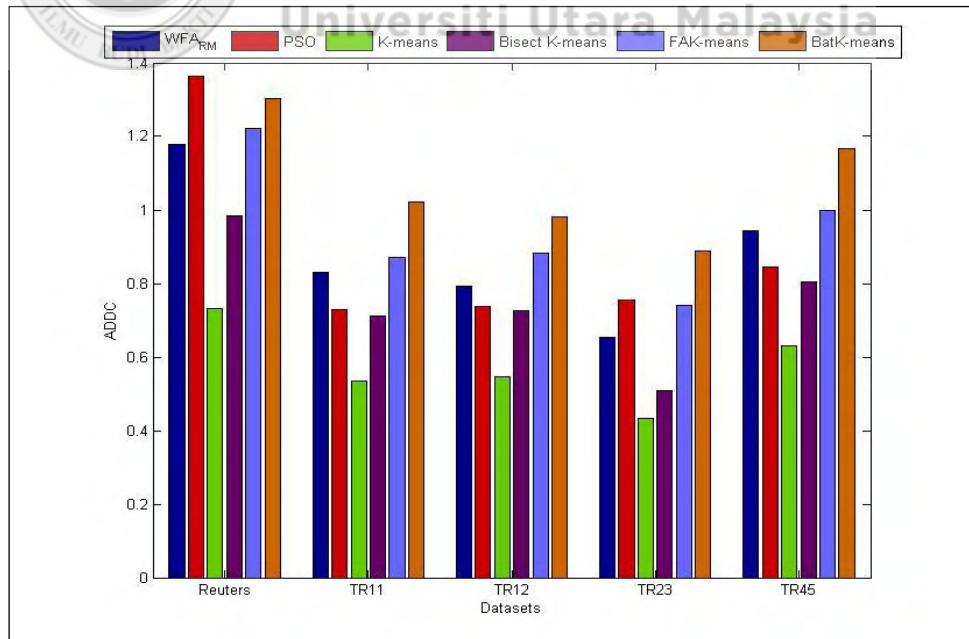


Figure 7.5. Average ADDC result:  $WFA_{RM}$  vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets

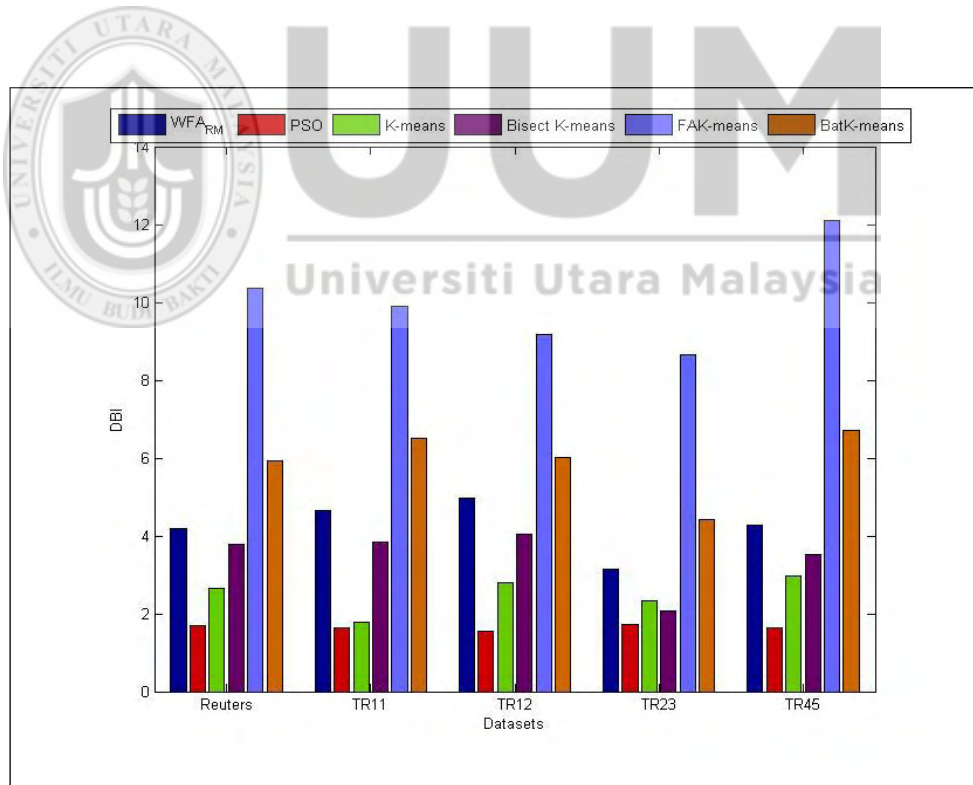
Table 7.6

*Relative quality DBI (Average, Best, Worst, standard deviation): WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets.*

Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	4.1820	4.1148	4.2540	0.0395
	PSO	<b>1.6825</b>	1.4268	1.9790	0.1461
	K-means	2.6385	1.4703	4.3945	0.7094
	Bisect K-means	3.7740	1.8275	5.1108	0.6847
	FAK-means	10.3502	9.8647	10.8133	0.2656
	BatK-means	5.9289	4.2084	8.8921	1.0682
Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	4.6518	4.4176	4.8660	0.1022
	PSO	<b>1.6241</b>	1.3495	2.1186	0.1725
	K-means	1.7858	0.1227	3.9547	0.7849
	Bisect K-means	3.8466	2.3599	5.1885	0.5293
	FAK-means	9.8927	9.6783	10.1788	0.1123
	BatK-means	6.5040	3.2826	8.9772	1.6546
Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	4.9590	4.8340	4.9750	0.0361
	PSO	<b>1.5605</b>	1.2371	1.9013	0.1733
	K-means	2.8110	1.6858	4.0169	0.7292
	Bisect K-means	4.0448	2.4865	5.3419	0.8268
	FAK-means	9.1618	8.8490	9.5123	0.1481
	BatK-means	6.0015	4.0032	9.9218	1.4734
Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	3.1373	3.0462	3.2194	0.0373
	PSO	<b>1.7271</b>	1.3798	3.5081	0.3786
	K-means	2.3197	1.5730	3.1529	0.4317
	Bisect K-means	2.0830	1.5591	3.3308	0.4410
	FAK-means	8.6446	7.7680	9.1920	0.2983
	BatK-means	4.4182	2.1310	7.2550	1.2898
Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	4.2858	4.1705	4.3443	0.0399
	PSO	<b>1.6252</b>	1.3379	1.9207	0.1258
	K-means	2.9692	1.3755	4.0272	0.6142
	Bisect K-means	3.5129	2.2107	4.5727	0.5629
	FAK-means	12.1014	11.8120	12.4824	0.1786
	BatK-means	6.7070	3.4809	10.9666	2.2131

*Note: the best value is highlighted in 'bold'.*

For DBI metrics, in Table 7.6, it is noticed that PSO is the best to generate a smaller DBI value against other methods in all datasets. Furthermore, it can be seen that  $WFA_{RM}$  is the best to generate a smaller average DBI value against hybrid methods FAK-means and BatK-means in balanced and un-balanced datasets, where the best value generated by  $WFA_{RM}$  is (4.1820, 4.6518, 4.9590, 3.1373 and 4.2858) in (Reuters, TR11, TR12, TR23 and TR45) respectively. The percentage of success to obtain a lower DBI is 100% by  $WFA_{RM}$  against hybrid methods. Figure 7.6 illustrates the average DBI result of six methods; namely  $WFA_{RM}$ , PSO, K-means, Bisect K-means, FAK-means, and BatK-means represented in a graphical representation.



*Figure 7.6. Average DBI result:  $WFA_{RM}$  vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets.*

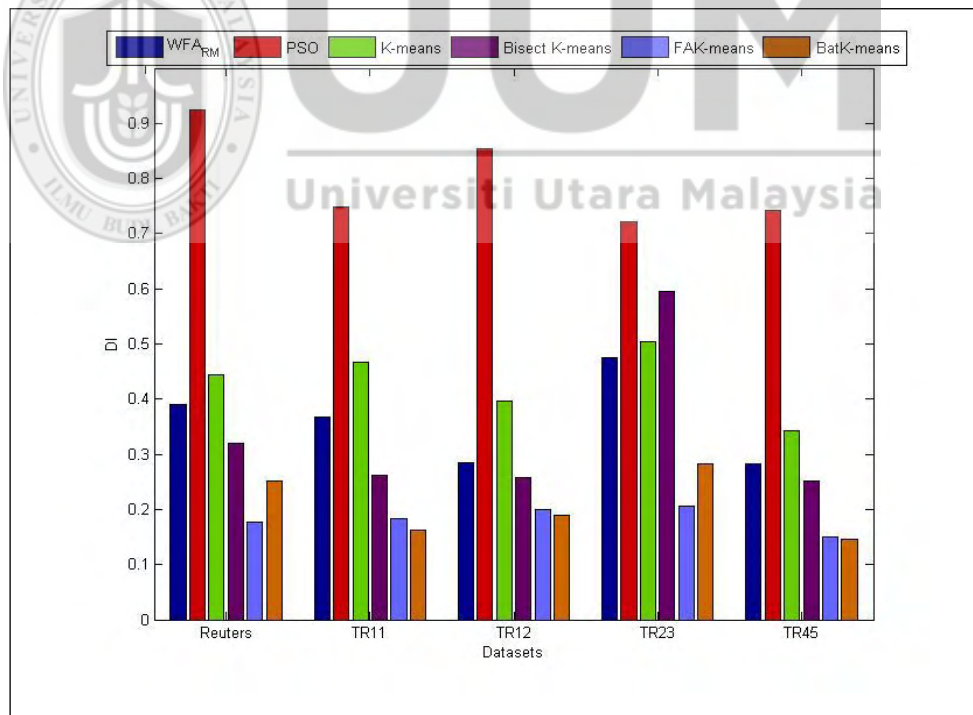
Table 7.7

*Relative quality DI (average, best DI, worst DI, standard deviation): WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets.*

Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	0.3909	0.3955	0.3642	0.0054
	PSO	<b>0.9244</b>	1.0712	0.7079	0.0842
	K-means	0.4432	0.6457	0.2494	0.1163
	Bisect K-means	0.3202	0.7293	0.2188	0.0999
	FAK-means	0.1776	0.1954	0.1623	0.0082
	BatK-means	0.2515	0.3459	0.1825	0.0361
Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	0.3673	0.3770	0.3412	0.0096
	PSO	<b>0.7479</b>	0.9954	0.0959	0.1862
	K-means	0.4668	5.6201	0.0458	0.9887
	Bisect K-means	0.2613	0.5005	0.2074	0.0517
	FAK-means	0.1842	0.1947	0.1752	0.0052
	BatK-means	0.1624	0.3728	0.0911	0.0611
Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	0.2857	0.2870	0.2816	0.0015
	PSO	<b>0.8539</b>	1.0377	0.6548	0.0737
	K-means	0.3971	0.8110	0.2245	0.1455
	Bisect K-means	0.2576	0.4154	0.1969	0.0474
	FAK-means	0.2004	0.2112	0.1910	0.0061
	BatK-means	0.1886	0.3186	0.1017	0.0523
Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	0.4747	0.5046	0.4626	0.0109
	PSO	<b>0.7205</b>	1.0782	0.0822	0.2466
	K-means	0.5051	0.7027	0.3103	0.1009
	Bisect K-means	0.5954	0.7642	0.3497	0.1057
	FAK-means	0.2066	0.2317	0.1854	0.0105
	BatK-means	0.2824	0.5977	0.1281	0.1141
Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	0.2824	0.2903	0.2719	0.0059
	PSO	<b>0.7414</b>	0.9313	0.5170	0.1065
	K-means	0.3419	0.6684	0.2448	0.0920
	Bisect K-means	0.2512	0.4020	0.1933	0.0544
	FAK-means	0.1501	0.1566	0.1396	0.0033
	BatK-means	0.1469	0.2685	0.0774	0.0517

*Note: the best value is highlighted in 'bold'.*

For DI metrics, Table 7.7 reports the DI value of six methods in balanced and unbalanced datasets. As can be seen in Table 7.7, PSO is better than the other methods in generating the highest DI value in all datasets, while WFA<sub>RM</sub> generates better than FAK-means and BatK-means, where the best average DI is (0.3909, 0.3673, 0.2857, 0.4747 and 0.2824) in (Reuters, TR11, TR12, TR23 and TR45) respectively. The overall standard deviation of the WFA<sub>RM</sub> algorithm is smaller than the other methods in most datasets (refer to Reuters and TR12). The percentage of success to obtain a high DI is 100% by WFA<sub>RM</sub> against hybrid methods. This result means the solution is more reliable and high in robustness. Figure 7.7 illustrates the average DI result of the six methods represented in a graphical representation.



*Figure 7.7. Average DI result: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FA K-means vs. BatK-means using different datasets*

Table 7.8

*Summary of quality performance results: WFA<sub>RM</sub> vs. PSO vs. K-means vs. Bisect K-means vs. FAK-means vs. BatK-means.*

Datasets	Algorithms	External Metrics			Internal & Relative Metrics		
		Purity	F-measure	Entropy	ADDC	DBI	DI
Reuters (300 document s and 6 classes)	WFA <sub>RM</sub>	✓	✓	✓			
	PSO					✓	✓
	K-means				✓		
	Bisect K-means						
	FAK-means						
	BatK-means						
Datasets	Algorithms	Purity	F-measure	Entropy	ADDC	DBI	DI
TR11 (414 document s and 9 classes)	WFA <sub>RM</sub>	✓		✓			
	PSO					✓	✓
	K-means				✓		
	Bisect K-means						
	FAK-means						
	BatK-means		✓				
Datasets	Algorithms	Purity	F-measure	Entropy	ADDC	DBI	DI
TR12 (313 document s and 8 classes)	WFA <sub>RM</sub>	✓		✓			
	PSO					✓	✓
	K-means				✓		
	Bisect K-means						
	FAK-means						
	BatK-means		✓				
Datasets	Algorithms	Purity	F-measure	Entropy	ADDC	DBI	DI
TR23 (204 document s and 6 classes)	WFA <sub>RM</sub>	✓		✓			
	PSO					✓	✓
	K-means				✓		
	Bisect K-means						
	FAK-means						
	BatK-means		✓				
Datasets	Algorithms	Purity	F-measure	Entropy	ADDC	DBI	DI
TR45 (690 document s and 10 classes)	WFA <sub>RM</sub>	✓		✓			
	PSO					✓	✓
	K-means				✓		
	Bisect K-means						
	FAK-means						
	BatK-means		✓				

The previous results indicate that WFA<sub>RM</sub> generates the best quality results in external performance metrics; namely Purity, and Entropy, against all static methods (PSO, K-means, Bisect K-means, FAK-means, and BatK-means), but in F-measure, WFA<sub>RM</sub> produces the highest F-measure compared to all methods excluding the BatK-means method as shown in Table 7.8. For the internal and relative metrics, ADDC, DBI, and DI are generated as the best quality results only against hybrid methods, FAK-means and BatK-means.

### 7.2.3 Evaluation Independent Samples T-test between WFA<sub>RM</sub> and Static Methods

Independent Samples T-test compares the means of two unrelated groups on similar dependent variable. Alternately, Independent Samples T-test is used to understand whether there is a significant (statistically) difference in the dependent variable based on the independent variable (Ross, 2010). In the evaluation of WFA<sub>RM</sub> against other static methods, it is questioned whether the average of [any metrics] for WFA<sub>RM</sub> is significantly (statistically) different from the average of [any metrics] for any other static methods. This involves testing whether the sample means for the average of any metrics among any two methods subjects in the sample are statistically different. The hypotheses for Independent Samples T-test can be expressed in mathematical equivalents.

Null hypothesis  $H_0$ :  $Mean(WFA_{RM}) = Mean(\text{any static methods})$

Alternative hypothesis  $H_1$ :  $Mean(WFA_{RM}) \neq Mean(\text{any static methods})$

Where,  $Mean(WFA_{RM})$ ,  $Mean(\text{any static methods})$  are the population means for the WFA<sub>RM</sub> algorithm and any other static method.



Table 7.9

*The P-value between  $WFA_{RM}$  & static methods using average purity results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average Purity (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PSO	7.5357E-35	2.1850E-22
	$WFA_{RM}$ and K-means	1.0720E-38	1.9788E-24
	$WFA_{RM}$ and Bisect K-means	5.4242E-24	1.5012E-16
	$WFA_{RM}$ and FAK-means	2.3550E-89	4.5275E-70
	$WFA_{RM}$ and BatK-means	5.1725E-09	1.4877E-07
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PSO	8.2885E-30	1.2987E-22
	$WFA_{RM}$ and K-means	3.5848E-29	1.7263E-21
	$WFA_{RM}$ and Bisect K-means	1.0471E-10	5.8665E-09
	$WFA_{RM}$ and FAK-means	2.4134E-62	2.8869E-40
	$WFA_{RM}$ and BatK-means	<b>0.3136</b>	<b>0.3166</b>
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PSO	2.7896E-09	1.0220E-07
	$WFA_{RM}$ and K-means	0.0141	0.0171
	$WFA_{RM}$ and Bisect K-means	<b>0.3972</b>	<b>0.4006</b>
	$WFA_{RM}$ and FAK-means	5.0969E-75	1.7287E-50
	$WFA_{RM}$ and BatK-means	1.0808E-12	5.8565E-10
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PSO	1.7600E-12	7.5911E-10
	$WFA_{RM}$ and K-means	6.5395E-11	7.984E-09
	$WFA_{RM}$ and Bisect K-means	7.1881E-33	1.0171E-21
	$WFA_{RM}$ and FAK-means	6.2078E-72	3.5164E-53
	$WFA_{RM}$ and BatK-means	0.00199	0.00299
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PSO	5.9420E-35	1.8525E-24
	$WFA_{RM}$ and K-means	3.3588E-15	9.2252E-12
	$WFA_{RM}$ and Bisect K-means	3.2908E-15	9.8506E-12
	$WFA_{RM}$ and FAK-means	1.7519E-72	4.6537E-49
	$WFA_{RM}$ and BatK-means	<b>0.0697</b>	<b>0.0740</b>

*Hint: The value highlighted in bold indicates not significance value.*

In Table 7.9, the associated P-value (sig 2-tailed test) using average Purity is illustrated. Since the P-value between WFA<sub>RM</sub> and any other methods is smaller than (0.05), the null hypothesis is rejected, the mean of any metrics for WFA<sub>RM</sub> and any static methods is the same and the alternative hypothesis is accepted to conclude that there is a significant difference in the mean of purity metric for WFA<sub>RM</sub> and any static methods. This excludes the P-value between WFA<sub>RM</sub> and BatK-means (bold value in Table 7.9) in two datasets, TR11 and TR45, which are (0.3136 and 0.3166, 0.0697 and 0.0740), larger than (0.05), and the P-value between WFA<sub>RM</sub> and K-means (bold value in Table 7.9) in the TR12 dataset. This means that the sample means for average purity between WFA<sub>RM</sub> and BatK-means and between WFA<sub>RM</sub> and K-means subjects in the sample are statistically not different (accepting the null hypothesis in this situation).

In Table 7.10, the associated P-value (sig 2-tailed test) using average F-measure is reported. As seen in the table, the P-value between WFA<sub>RM</sub> and any other methods is smaller than (0.05), the alternative hypothesis is accepted to conclude that there is a significant difference in the mean of purity metric for WFA<sub>RM</sub> and any static methods and the null hypothesis is rejected. This excludes the P-value between WFA<sub>RM</sub> and BatK-means (bold value in Table 7.10) in the TR11 dataset, which is larger than (0.05), and the P-value between WFA<sub>RM</sub> and K-means (bold value in Table 7.10) in the TR12 and TR23 datasets, and also between WFA<sub>RM</sub> and PSO in the TR23 dataset, where, the null hypothesis is accepted because the sample means for average F-measure are statistically not different.

Table 7.10

*The P-value between  $WFA_{RM}$  & static methods using average F-measure results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average F-measure (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PSO	2.1661E-34	3.8607E-22
	$WFA_{RM}$ and K-means	6.6481E-35	2.0587E-22
	$WFA_{RM}$ and Bisect K-means	1.3424E-20	1.1775E-14
	$WFA_{RM}$ and FAK-means	1.4353E-86	1.1085E-61
	$WFA_{RM}$ and BatK-means	8.5899E-08	1.1067E-06
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PSO	3.6937E-19	9.1096E-19
	$WFA_{RM}$ and K-means	4.3328E-17	3.2146E-16
	$WFA_{RM}$ and Bisect K-means	8.6466E-04	0.0011
	$WFA_{RM}$ and FAK-means	5.5624E-43	8.8676E-28
	$WFA_{RM}$ and BatK-means	<b>0.4309</b>	<b>0.4316</b>
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PSO	0.0043	0.0058
	$WFA_{RM}$ and K-means	<b>0.2412</b>	<b>0.2459</b>
	$WFA_{RM}$ and Bisect K-means	5.2691E-09	1.4151E-07
	$WFA_{RM}$ and FAK-means	1.0036E-57	1.9769E-52
	$WFA_{RM}$ and BatK-means	4.8149E-20	1.8406E-14
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PSO	<b>0.6291</b>	<b>0.6308</b>
	$WFA_{RM}$ and K-means	<b>0.6298</b>	<b>0.6315</b>
	$WFA_{RM}$ and Bisect K-means	2.4755E-08	1.2238E-07
	$WFA_{RM}$ and FAK-means	3.1148E-44	2.9028E-34
	$WFA_{RM}$ and BatK-means	<b>0.07152</b>	<b>0.0763</b>
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PSO	5.7629E-29	5.7274E-23
	$WFA_{RM}$ and K-means	2.5187E-08	3.1646E-07
	$WFA_{RM}$ and Bisect K-means	3.8213E-06	1.5047E-05
	$WFA_{RM}$ and FAK-means	4.4715E-60	1.2379E-41
	$WFA_{RM}$ and BatK-means	0.0161	0.0185

*Hint: The value highlighted in bold indicates not significance value.*

Table 7.11

*The P-value between  $WFA_{RM}$  & static methods using average Entropy results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average Entropy (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PSO	4.8503E-40	1.1814E-25
	$WFA_{RM}$ and K-means	4.5425E-31	1.4058E-20
	$WFA_{RM}$ and Bisect K-means	1.2400E-19	3.6375E-14
	$WFA_{RM}$ and FAK-means	3.0772E-91	1.6751E-78
	$WFA_{RM}$ and BatK-means	1.8770E-05	6.0939E-05
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PSO	6.0493E-34	3.3017E-26
	$WFA_{RM}$ and K-means	1.7498E-25	1.8388E-18
	$WFA_{RM}$ and Bisect K-means	4.9078E-12	6.0876E-10
	$WFA_{RM}$ and FAK-means	1.1813E-63	2.2057E-39
	$WFA_{RM}$ and BatK-means	0.0169	0.0190
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PSO	1.2506E-14	3.5941E-11
	$WFA_{RM}$ and K-means	1.8669E-04	4.0755E-04
	$WFA_{RM}$ and Bisect K-means	<b>0.8171</b>	<b>0.8179</b>
	$WFA_{RM}$ and FAK-means	7.7271E-77	1.2582E-59
	$WFA_{RM}$ and BatK-means	1.7514E-11	3.3768E-09
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PSO	2.0144E-13	1.3595E-10
	$WFA_{RM}$ and K-means	1.4415E-11	2.2316E-09
	$WFA_{RM}$ and Bisect K-means	1.5171E-31	1.5072E-22
	$WFA_{RM}$ and FAK-means	3.9209E-62	1.7294E-59
	$WFA_{RM}$ and BatK-means	<b>0.5874</b>	<b>0.5894</b>
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PSO	2.6751E-33	1.0892E-22
	$WFA_{RM}$ and K-means	4.5030E-10	2.5965E-08
	$WFA_{RM}$ and Bisect K-means	3.2437E-12	9.0738E-10
	$WFA_{RM}$ and FAK-means	1.8378E-77	1.3670E-49
	$WFA_{RM}$ and BatK-means	<b>0.1590</b>	<b>0.1638</b>

*Hint: The value highlighted in bold indicates not significance value.*

In Table 7.11, the associated P-value (sig 2-tailed test) using average Entropy is recorded. The P-value between WFA<sub>RM</sub> and other methods is smaller than (0.05). In this situation, the null hypothesis is rejected and the alternative hypothesis is accepted. The alternative hypothesis indicates that statistically, there is a significant difference in the mean of Entropy metric for WFA<sub>RM</sub> and any static methods. However, this excludes the P-value between WFA<sub>RM</sub> and Bisect K-means (illustrates in bold value in Table 7.11) in the TR12 dataset, and also the P-value between WFA<sub>RM</sub> and BatK-means (highlight in bold value in Table 7.11) in the two datasets, TR23 and TR45, which are larger than (0.05). In this case, the null hypothesis is accepted, which indicates that there is no difference in the mean of Entropy.

In Table 7.12, the associated P-value (sig 2-tailed test) using average ADDC is reported. As seen in Table 7.12, the P-value between WFA<sub>RM</sub> and any other methods is smaller than (0.05), the alternative hypothesis is accepted to conclude that there is a significant difference in the mean of ADDC metric for WFA<sub>RM</sub> and any static methods and the null hypothesis is rejected. This excludes the P-value between WFA<sub>RM</sub> and PSO (highlighted value in bold in Table 7.12) in the TR12 dataset which is (0.1116 and 0.1170), larger than (0.05); in this status, the null hypothesis  $H_0$  is accepted because the sample means for average Entropy are statistically not different.

Table 7.12

*The P-value between  $WFA_{RM}$  & static methods using average ADDC results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average ADDC (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PSO	4.1111E-07	3.5550E-06
	$WFA_{RM}$ and K-means	5.0850E-20	2.6432E-14
	$WFA_{RM}$ and Bisect K-means	4.9067E-10	3.1703E-08
	$WFA_{RM}$ and FAK-means	4.0892E-56	5.3011E-55
	$WFA_{RM}$ and BatK-means	6.7768E-19	1.0681E-13
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PSO	9.3507E-05	2.2941E-04
	$WFA_{RM}$ and K-means	1.7647E-06	1.0468E-05
	$WFA_{RM}$ and Bisect K-means	4.2178E-09	1.2511E-07
	$WFA_{RM}$ and FAK-means	1.9093E-33	5.0490E-22
	$WFA_{RM}$ and BatK-means	5.8447E-22	1.1521E-15
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PSO	<b>0.1116</b>	<b>0.1170</b>
	$WFA_{RM}$ and K-means	1.7738E-15	1.1452E-11
	$WFA_{RM}$ and Bisect K-means	0.0046	0.0062
	$WFA_{RM}$ and FAK-means	9.3256E-75	1.8792E-50
	$WFA_{RM}$ and BatK-means	3.8691E-12	1.3257E-09
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PSO	3.5086E-07	3.1477E-06
	$WFA_{RM}$ and K-means	9.5827E-22	2.7166E-15
	$WFA_{RM}$ and Bisect K-means	8.1957E-18	4.5868E-13
	$WFA_{RM}$ and FAK-means	1.0029E-73	6.4749E-66
	$WFA_{RM}$ and BatK-means	3.7862E-14	7.3691E-11
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PSO	0.0012	0.0019
	$WFA_{RM}$ and K-means	1.3067E-20	1.1072E-14
	$WFA_{RM}$ and Bisect K-means	1.2275E-08	2.7890E-07
	$WFA_{RM}$ and FAK-means	3.5010E-46	8.6433E-29
	$WFA_{RM}$ and BatK-means	1.5965E-13	1.7110E-10

*Hint: The value highlighted in bold indicates not significance value.*

Table 7.13

*The P-value between  $WFA_{RM}$  & static methods using average DBI results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average DBI (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PSO	3.9736E-64	2.4535E-41
	$WFA_{RM}$ and K-means	3.3554E-17	1.0197E-12
	$WFA_{RM}$ and Bisect K-means	0.0019	0.0028
	$WFA_{RM}$ and FAK-means	2.1062E-72	9.6453E-43
	$WFA_{RM}$ and BatK-means	1.5944E-12	7.4668E-10
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PSO	6.9172E-62	1.1491E-52
	$WFA_{RM}$ and K-means	1.6095E-27	8.6671E-19
	$WFA_{RM}$ and Bisect K-means	3.0579E-11	2.9550E-09
	$WFA_{RM}$ and FAK-means	1.2605E-82	5.0308E-82
	$WFA_{RM}$ and BatK-means	8.6388E-08	1.1116E-06
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PSO	6.6255E-68	1.0582E-41
	$WFA_{RM}$ and K-means	4.3956E-23	4.7159E-16
	$WFA_{RM}$ and Bisect K-means	1.1245E-07	1.3654E-06
	$WFA_{RM}$ and FAK-means	5.5711E-77	9.2456E-48
	$WFA_{RM}$ and BatK-means	2.7468E-04	5.6162E-04
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PSO	4.9061E-28	6.5693E-19
	$WFA_{RM}$ and K-means	9.0250E-15	2.6256E-11
	$WFA_{RM}$ and Bisect K-means	6.6772E-19	9.1849E-14
	$WFA_{RM}$ and FAK-means	5.9442E-67	1.7099E-39
	$WFA_{RM}$ and BatK-means	1.1302E-06	7.4928E-06
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PSO	4.0577E-69	7.1991E-46
	$WFA_{RM}$ and K-means	6.3778E-17	1.4437E-12
	$WFA_{RM}$ and Bisect K-means	4.1942E-10	2.6889E-08
	$WFA_{RM}$ and FAK-means	5.4503E-88	3.6790E-53
	$WFA_{RM}$ and BatK-means	1.4078E-07	1.6264E-06

*Hint: The value highlighted in bold indicates not significance value.*

Table 7.14

*The P-value between  $WFA_{RM}$  & static methods using average DI results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average DI (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PSO	1.9596E-40	2.7765E-25
	$WFA_{RM}$ and K-means	0.0170	0.0201
	$WFA_{RM}$ and Bisect K-means	2.745E-04	5.5816E-04
	$WFA_{RM}$ and FAK-means	4.6583E-71	5.6965E-63
	$WFA_{RM}$ and BatK-means	1.0172E-28	1.4515E-19
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PSO	4.2066E-16	4.6106E-12
	$WFA_{RM}$ and K-means	<b>0.5833</b>	<b>0.5854</b>
	$WFA_{RM}$ and Bisect K-means	7.0952E-16	2.8851E-12
	$WFA_{RM}$ and FAK-means	1.7237E-64	2.4456E-52
	$WFA_{RM}$ and BatK-means	1.3589E-25	7.0368E-18
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PSO	3.0417E-45	1.3087E-27
	$WFA_{RM}$ and K-means	9.4771E-05	2.3518E-04
	$WFA_{RM}$ and Bisect K-means	0.0020	0.0030
	$WFA_{RM}$ and FAK-means	3.2519E-59	4.8446E-38
	$WFA_{RM}$ and BatK-means	1.6991E-14	4.4805E-11
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PSO	1.0636E-06	7.1099E-06
	$WFA_{RM}$ and K-means	<b>0.1057</b>	<b>0.1108</b>
	$WFA_{RM}$ and Bisect K-means	5.8999E-08	7.9744E-07
	$WFA_{RM}$ and FAK-means	8.1165E-66	9.812E-66
	$WFA_{RM}$ and BatK-means	6.5333E-13	3.6776E-10
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PSO	2.1441E-31	1.5248E-20
	$WFA_{RM}$ and K-means	8.0575E-04	0.0014
	$WFA_{RM}$ and Bisect K-means	0.0028	0.0040
	$WFA_{RM}$ and FAK-means	1.9527E-68	1.8833E-56
	$WFA_{RM}$ and BatK-means	1.2840E-20	7.7344E-15

*Hint: The value highlighted in bold indicates not significance value.*



In Table 7.13, the associated P-value (sig 2-tailed test) using average DBI is reported. From the table, the P-value between WFA<sub>RM</sub> and any other methods is smaller than (0.05), the alternative hypothesis is accepted to conclude that there is a significant difference in the mean of ADDC metric for WFA<sub>RM</sub> and any static methods and the null hypothesis is rejected.

In Table 7.14, the associated P-value (sig 2-tailed test) using average DI is shown. As seen in the table, the P-value between WFA<sub>RM</sub> and any other methods is smaller than (0.05), the alternative hypothesis is accepted to conclude that there is a significant difference in the mean of DI metric for WFA<sub>RM</sub> and any static methods and the null hypothesis is rejected. This excludes the P-value between WFA<sub>RM</sub> and K-means (highlighted value in bold in Table 7.14) in the TR11 and TR23 datasets, which are (0.1116 and 0.1170, 0.1057 and 0.1108), and are larger than (0.05); the null hypothesis  $H_0$  is accepted in this case because the sample means for average DI are statistically not different.

### **7.3 Comparison WFA<sub>RM</sub> with Dynamic Methods**

This section includes the evaluation of WFA<sub>RM</sub> against two dynamic methods, Practical General Stochastic Clustering Method (PGSCM) (Tan, Ting, & Teng, 2011a), and Dynamic Hybrid Genetic algorithm with Particle Swarm Optimization (DCPG) (Kuo, Syu, Chen, & Tien, 2012). The comparison is conducted in three parts; evaluation of the produced number of clusters among WFA<sub>RM</sub>, PGSCM and DCPG, evaluation of the performance metrics among WFA<sub>RM</sub>, PGSCM and DCPG,

and evaluation of the statistical Independent Samples T-test among  $WFA_{RM}$ , PGSCM and DCPG.

### 7.3.1 Evaluation Number of Clusters between $WFA_{RM}$ and Dynamic Methods

Table 7.15 displays the average number of clusters automatically generated by  $WFA_{RM}$ , PGSCM and DCPG without any information support about the dataset.

Table 7.15

*Average number of clusters:  $WFA_{RM}$  vs. PGSCM vs. DCPG using different datasets.*

Datasets	Average number of clusters		
	$WFA_{RM}$	PGSCM	DCPG
Reuters (300 documents and 6 classes)	6	6.07 $\approx$ 6	9.03 $\approx$ 9
TR11 (414 documents and 9 classes)	9.27 $\approx$ 9	6.93 $\approx$ 7	9.80 $\approx$ 10
TR12 (313 documents and 8 classes)	8	6.27 $\approx$ 6	8.90 $\approx$ 9
TR23 (204 documents and 6 classes)	6.07 $\approx$ 6	3.63 $\approx$ 4	6.53 $\approx$ 7
TR45 (690 documents and 10 classes)	9.83 $\approx$ 10	3.90 $\approx$ 4	13.77 $\approx$ 14

As can be seen in Table 7.15, the average number of clusters produced by the  $WFA_{RM}$  in iteration 20 is approximately (6, 9, 8, 6, and 10) in the (Reuters, TR11, TR12, TR23 and TR45) datasets respectively, these values are near to the real number of clusters; while PGSCM only produces 6 in the Reuters dataset. This result means that  $WFA_{RM}$  is better than PGSCM and DCPG. Figure 7.8 presents the results of the number of generated clusters by  $WFA_{RM}$ , PGSCM, DCPG and the real number of clusters of the datasets.

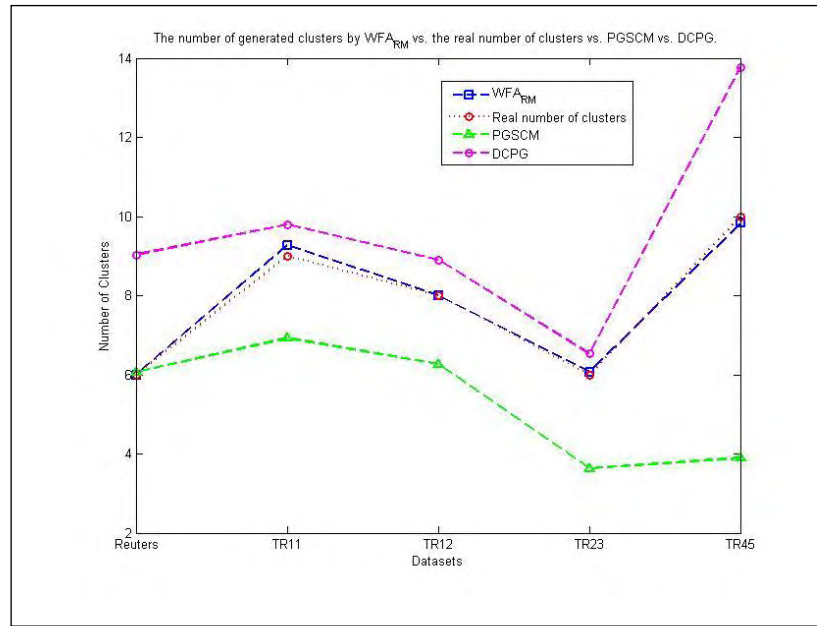


Figure 7.8. Number of generated clusters: WFA<sub>RM</sub> vs. the real number of clusters vs. PGSCM vs. DCPG

### 7.3.2 Evaluation Performance Metrics between WFA<sub>RM</sub> and Dynamic Methods

In this section, WFA<sub>RM</sub> is evaluated against two dynamic methods, namely PGSCM and DCPG. The evaluations are carried out using external quality metrics such as Purity, F-measure and Entropy, and using internal and relative quality metrics such as ADDC, DBI and DI. Table 7.16 reports the average purity, best and worst purity, and standard deviation of WFA<sub>RM</sub>, PGSCM and DCPG.

As observed in Table 7.16, WFA<sub>RM</sub> generates a higher purity against PGSCM and DCPG, where the best average purity value is 0.7351, 0.6810, 0.4752, 0.6266, and 0.6355 generated by WFA<sub>RM</sub> in Reuters, TR11, TR12, TR23 and TR45 respectively. The overall standard deviation of the WFA<sub>RM</sub> algorithm is smaller than PGSCM and DCPG in most datasets (refer to Reuters, TR12, TR23 and TR45), which means the

solution generated by  $WFA_{RM}$  is more reliable and high in robustness. The percentage of success to obtain the highest purity is 100% by  $WFA_{RM}$ .

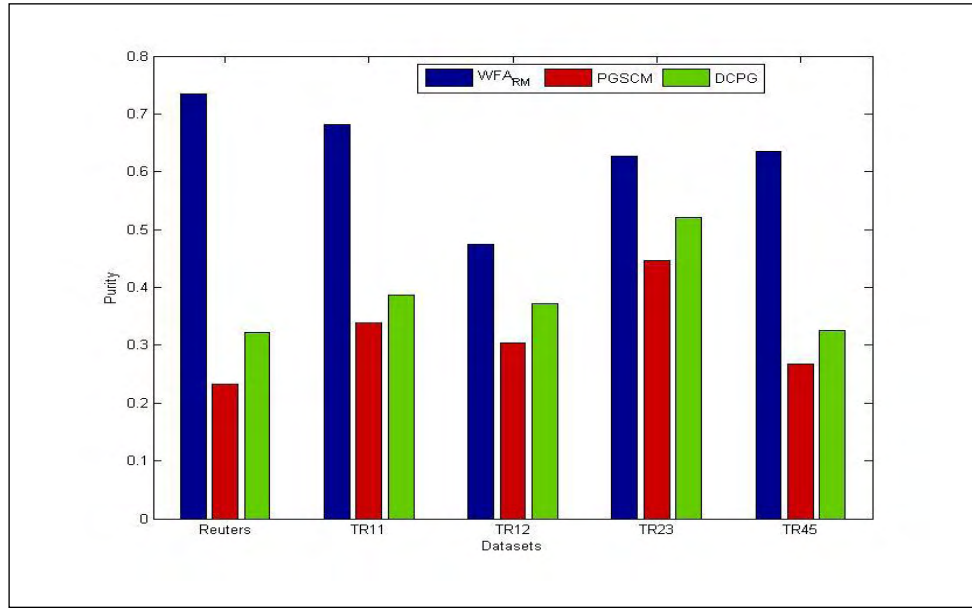
Table 7.16

*External quality Purity (average, best, worst, standard deviation):  $WFA_{RM}$  vs. PGSCM vs. DCPG using different datasets.*

Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
Reuters (300 documents and 6 classes)	$WFA_{RM}$	<b>0.7351</b>	0.7433	0.7300	0.0051
	PGSCM	0.2333	0.2633	0.1967	0.0160
	DCPG	0.3226	0.5967	0.1933	0.1091
Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
TR11 (414 documents and 9 classes)	$WFA_{RM}$	<b>0.6810</b>	0.7246	0.6594	0.0224
	PGSCM	0.3385	0.3671	0.3188	0.0160
	DCPG	0.3874	0.5290	0.3261	0.0582
Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
TR12 (313 documents and 8 classes)	$WFA_{RM}$	<b>0.4752</b>	0.4760	0.4696	0.0022
	PGSCM	0.3036	0.3195	0.2971	0.0065
	DCPG	0.3714	0.4441	0.3131	0.0435
Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
TR23 (204 documents and 6 classes)	$WFA_{RM}$	<b>0.6266</b>	0.6324	0.6176	0.0032
	PGSCM	0.4472	0.4706	0.4461	0.0046
	DCPG	0.5209	0.6569	0.4608	0.0489
Datasets	Algorithms	Average Purity	Best Purity	Worst Purity	Standard Deviation
TR45 (690 documents and 10 classes)	$WFA_{RM}$	<b>0.6355</b>	0.6609	0.5812	0.0161
	PGSCM	0.2681	0.3464	0.2319	0.0321
	DCPG	0.3258	0.5145	0.2478	0.0616

*Note: the best value is highlighted in 'bold'.*

Figure 7.9 shows a plotted graph of the external quality metrics; namely the average purity result for  $WFA_{RM}$ , PGSCM and DCPG.



*Figure 7.9. Average Purity result: WFA<sub>RM</sub> vs. PGSCM vs. DCPG using different datasets*

For F-measure metric, Table 7.17 reports the average F-measure (precision and recall), best and worst F-measure and standard deviation of WFA<sub>RM</sub>, PGSCM and DCPG. As observed in Table 7.17, WFA<sub>RM</sub> generates the highest average F-measure (0.7069, 0.5631, 0.3649, and 0.5031) against PGSCM and DCPG in most datasets, Reuters, TR11, TR12 and TR45 respectively. However, DCPG generates a higher F-measure of 0.4457 only in the TR23 dataset. Further, the precision and recall for WFA<sub>RM</sub> are higher in most datasets which highly affect the F-measure value of WFA<sub>RM</sub>. The overall standard deviation of the WFA<sub>RM</sub> algorithm is smaller than PGSCM and DCPG in most datasets (refer to Reuters, TR12, TR23 and TR45). It can be concluded that the solution generated by WFA<sub>RM</sub> is more reliable and high in robustness. The percentage of success to obtain the highest F-measure is 80% by WFA<sub>RM</sub>. Figure 7.10 shows a plotted graph of the external quality metrics, the average F-measure result for WFA<sub>RM</sub>, PGSCM and DCPG.

Table 7.17

*External quality F-measure (average, best, worst, standard deviation): WFA<sub>RM</sub> vs. PGSCM vs. DCPG using different datasets.*

Datasets	Algorithms	Average F-measure [Precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	<b>0.7069</b> [0.6808, 0.7351]	0.7138	0.7022	0.0045
	PGSCM	0.2369 [0.2406, 0.2333]	0.2740	0.2067	0.0178
	DCPG	0.3565 [0.3984, 0.3226]	0.5585	0.2755	0.0839
Datasets	Algorithms	Average F-measure [Precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	<b>0.5631</b> [0.4800, 0.6810]	0.6547	0.4615	0.0520
	PGSCM	0.2566 [0.2066, 0.3385]	0.3092	0.1950	0.0321
	DCPG	0.3346 [0.2945, 0.3874]	0.5345	0.2661	0.0690
Datasets	Algorithms	Average F-measure [Precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	<b>0.3649</b> [0.2962, 0.4752]	0.3676	0.3467	0.0071
	PGSCM	0.2273 [0.1817, 0.3036]	0.2563	0.1974	0.0167
	DCPG	0.3121 [0.2691, 0.3714]	0.4097	0.2705	0.0490
Datasets	Algorithms	Average F-measure [Precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	0.4337 [0.3316, 0.6266]	0.4460	0.4024	0.0089
	PGSCM	0.3465 [0.2828, 0.4472]	0.4061	0.2779	0.0327
	DCPG	<b>0.4457</b> [0.3895, 0.5209]	0.5839	0.3719	0.0479
Datasets	Algorithms	Average F-measure [Precision, Recall]	Best F-measure	Worst F-measure	Standard Deviation
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	<b>0.5031</b> [0.4164, 0.6355]	0.5489	0.4314	0.0227
	PGSCM	0.2399 [0.2171, 0.2681]	0.3131	0.2061	0.0312
	DCPG	0.2581 [0.2137, 0.3258]	0.4495	0.2301	0.0470

*Note: the best value is highlighted in 'bold', average precision and average recall in [].*

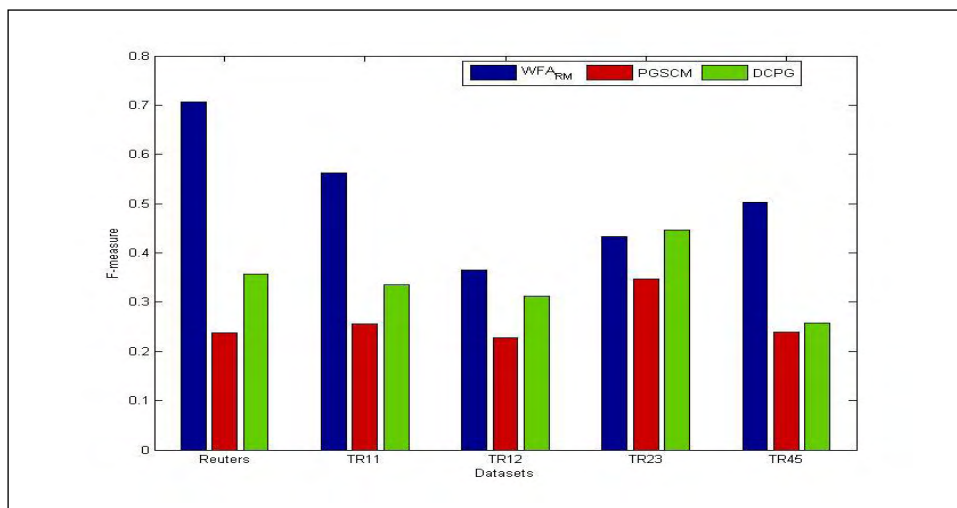


Figure 7.10. Average F-measure result: WFA<sub>RM</sub> vs. PGSCM vs. DCPG using different datasets

For Entropy metric, Table 7.18 reports the average Entropy, best and worst Entropy, and standard deviation of WFA<sub>RM</sub>, PGSCM and DCPG. In Table 7.18, it can be seen that the Entropy of WFA<sub>RM</sub> is better (smaller) than PGSCM and DCPG in all datasets (balanced and un-balanced datasets), where the best average Entropy is 1.0420, 1.2584, 1.9200, 1.4980, and 1.6742 outputted by WFA<sub>RM</sub>. The overall standard deviation of the WFA<sub>RM</sub> algorithm is smaller than PGSCM and DCPG in most un-balanced datasets (refer to TR12, TR23 and TR45). The percentage of success to obtain a lower Entropy is 100% by WFA<sub>RM</sub>.

Table 7.18

*External quality Entropy (average, best, worst, standard deviation): WFA<sub>RM</sub> vs. PGSCM vs. DCPG using different datasets.*

Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	<b>1.0420</b>	1.0044	1.0653	0.0258
	PGSCM	2.5176	2.4590	2.5687	0.0255
	DCPG	2.1418	1.4689	2.5154	1.4689
Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	<b>1.2584</b>	1.0489	1.3384	0.0798
	PGSCM	2.5603	2.4322	2.6168	0.0521
	DCPG	2.3780	1.7252	2.6227	0.2371
Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	<b>1.9200</b>	1.9154	1.9531	0.0119
	PGSCM	2.6613	2.6006	2.7138	0.0301
	DCPG	2.4581	2.1306	2.7122	0.1844
Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	<b>1.4980</b>	1.4295	1.5306	0.0198
	PGSCM	2.0439	1.9702	2.0828	0.0245
	DCPG	1.7994	1.4088	2.0245	0.1720
Datasets	Algorithms	Average Entropy	Best Entropy	Worst Entropy	Standard Deviation
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	<b>1.6742</b>	1.5923	1.7450	0.0434
	PGSCM	2.9460	2.7703	3.0147	0.0584
	DCPG	2.6581	1.8688	2.9685	0.2258

*Note: the best value is highlighted in 'bold'.*

The highest values of Purity and F-measure and a lower value of Entropy indicate the best clustering algorithm (Forsati, Mahdavi, Shamsfard, & Meybodi, 2013; Murugesan & Zhang, 2011a, 2011b), It can be concluded from the previous results that  $WFA_{RM}$  is better than PGSCM and DCPG in external quality metrics. Figure 7.11 shows a plotted graph of the external quality metrics, namely the average Entropy result for  $WFA_{RM}$ , PGSCM and DCPG.

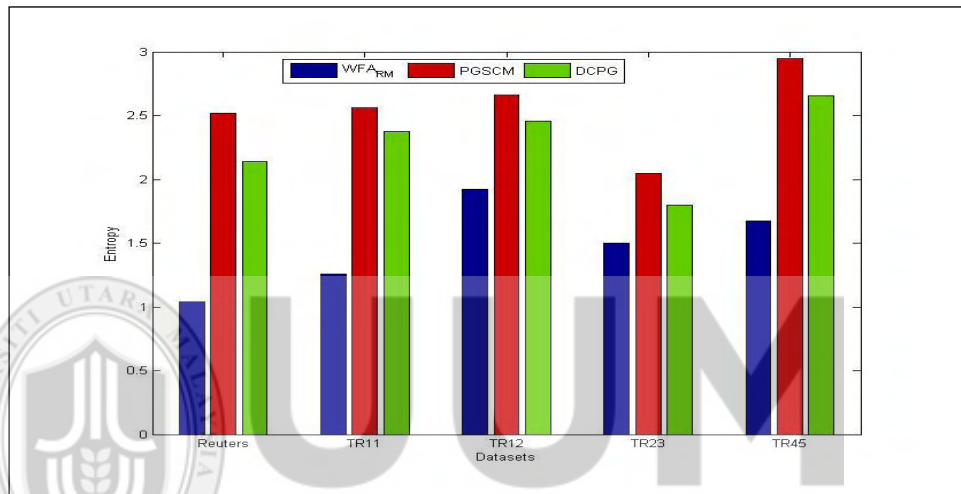


Figure 7.11. Average Entropy result:  $WFA_{RM}$  vs. PGSCM vs. DCPG using different datasets

For ADDC metric, Table 7.19 reports the average ADDC, best and worst ADDC and standard deviation of  $WFA_{RM}$ , PGSCM and DCPG. As shown in Table 7.19, the  $WFA_{RM}$  algorithm generates a lower average ADDC compared to PGSCM in all balanced and un-balanced datasets, while DCPG produces a lower average ADDC compared against  $WFA_{RM}$  and PGSCM in all balanced and un-balanced datasets. A smaller ADDC value means the best quality clustering algorithm (Cui, Potok, & Palathingal, 2005; Forsati, Mahdavi, Shamsfard, & Meybodi, 2013). The overall standard deviation of the  $WFA_{RM}$  algorithm is smaller than PGSCM and DCPG in all datasets.



Table 7.19

Internal quality ADDC (average, best, worst, standard deviation):  $WFA_{RM}$  vs. PGSCM vs. DCPG using different datasets.

Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
Reuters (300 documents and 6 classes)	$WFA_{RM}$	1.1786	1.1737	1.1807	0.0027
	PGSCM	1.4079	1.3891	1.4453	0.0134
	DCPG	<b>0.6047</b>	0.1802	0.9780	0.2046
Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
TR11 (414 documents and 9 classes)	$WFA_{RM}$	0.8293	0.8198	0.8529	0.0088
	PGSCM	1.0379	1.0226	1.0670	0.0093
	DCPG	<b>0.4833</b>	0.2257	0.7256	0.1211
Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
TR12 (313 documents and 8 classes)	$WFA_{RM}$	0.7928	0.7821	0.7942	0.0033
	PGSCM	1.0605	1.0467	1.0832	0.0077
	DCPG	<b>0.4924</b>	0.1566	0.7368	0.1662
Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
TR23 (204 documents and 6 classes)	$WFA_{RM}$	0.6527	0.6407	0.6559	0.0030
	PGSCM	0.8644	0.8091	0.8907	0.0156
	DCPG	<b>0.4474</b>	0.1800	0.7969	0.1736
Datasets	Algorithms	Average ADDC	Best ADDC	Worst ADDC	Standard Deviation
TR45 (690 documents and 10 classes)	$WFA_{RM}$	0.9448	0.9284	0.9503	0.0068
	PGSCM	1.1178	1.0716	1.1733	0.0261
	DCPG	<b>0.5339</b>	0.2264	1.1048	0.2193

Note: the best value is highlighted in 'bold'.

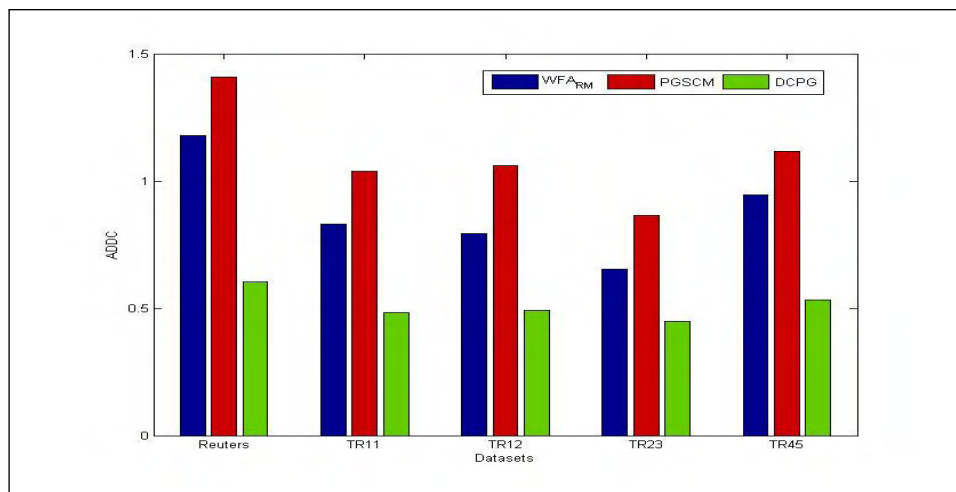


Figure 7.12. Average Entropy result:  $WFA_{RM}$  vs. PGSCM vs. DCPG using different datasets

This result indicates that the solution generated by  $WFA_{RM}$  is more reliable and high in robustness. The percentage of success to obtain a lower ADDC is 100% by  $WFA_{RM}$  against PGSCM and 0% against DCPG. Figure 7.12 shows a plotted graph of the result of the internal quality metrics, average ADDC, for  $WFA_{RM}$ , PGSCM and DCPG.

For DBI metric, Table 7.20 illustrates the average DBI, best and worst DBI and standard deviation of  $WFA_{RM}$ , PGSCM and DCPG.

Table 7.20

*Relative quality DBI (average, best, worst, standard deviation):  $WFA_{RM}$  vs. PGSCM vs. DCPG using different datasets.*

Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
Reuters (300 documents and 6 classes)	$WFA_{RM}$	4.1820	4.1148	4.2540	0.0395
	PGSCM	9.1919	2.8720	19.6237	4.9623
	DCPG	<b>2.3574</b>	1.0284	3.8852	0.9094
Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
TR11 (414 documents and 9 classes)	$WFA_{RM}$	4.6518	4.4176	4.8660	0.1022
	PGSCM	2.8577	2.6030	3.6195	0.1954
	DCPG	<b>2.5883</b>	1.3649	4.9672	0.8395
Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
TR12 (313 documents and 8 classes)	$WFA_{RM}$	4.9590	4.8340	4.9750	0.0361
	PGSCM	2.6426	2.5128	2.9368	0.1039
	DCPG	<b>2.3300</b>	1.1849	3.7710	0.6423
Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
TR23 (204 documents and 6 classes)	$WFA_{RM}$	3.1373	3.0462	3.2194	0.0373
	PGSCM	3.6783	5.0163	2.8169	0.6700
	DCPG	<b>2.6989</b>	0.9209	6.6405	1.6553
Datasets	Algorithms	Average DBI	Best DBI	Worst DBI	Standard Deviation
TR45 (690 documents and 10 classes)	$WFA_{RM}$	4.2858	4.1705	4.3443	0.0399
	PGSCM	3.3990	4.3600	2.6893	0.4044
	DCPG	<b>2.4342</b>	1.3880	5.6184	0.9863

*Note: the best value is highlighted in 'bold'.*

In Table 7.20, it is noticed that DCPG is the best to generate a smaller DBI value in balanced and un-balanced datasets against the  $WFA_{RM}$  and PGSCM methods, where the best average DBI value is 2.3574, 2.5883, 2.3300, 2.6989 and 2.4342 in Reuters, TR11, TR12, TR23, TR45 respectively outputted by DCPG; while  $WFA_{RM}$  produces the worst DBI values in all datasets. The standard deviation of the  $WFA_{RM}$  algorithm is smaller than DCPG and PGSCM. This result indicates that  $WFA_{RM}$  is more accurate. Figure 7.13 shows a plotted graph of the result of the relative quality metrics, average DBI, for  $WFA_{RM}$ , PGSCM and DCPG.

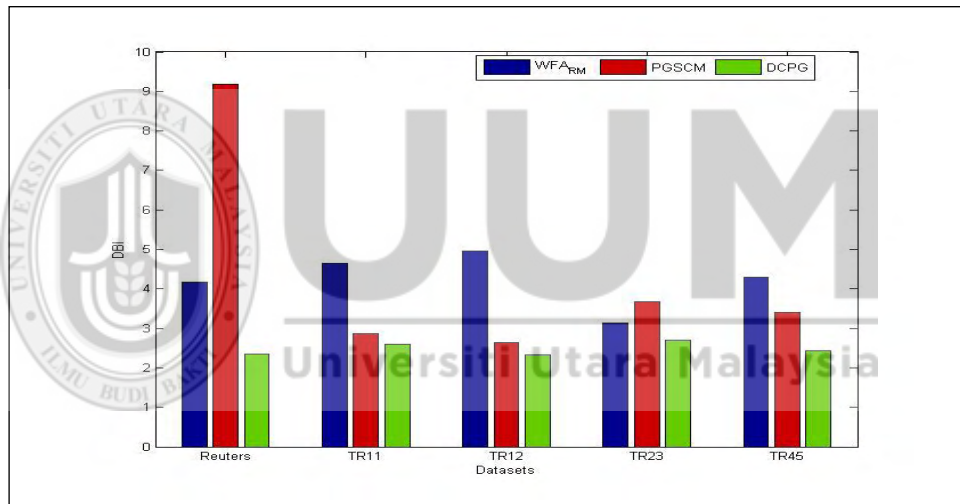


Figure 7.13. Average DBI result:  $WFA_{RM}$  vs. PGSCM vs. DCPG using different datasets

Table 7.21 illustrates the average DI, best and worst DI and standard deviation of  $WFA_{RM}$ , PGSCM and DCPG. As noticed in the table, the average DI value for PGSCM is larger than the DI value of  $WFA_{RM}$  and DCPG in three un-balanced datasets; TR11, TR12, and TR45; while in the remaining datasets, the balanced dataset (Reuters) and un-balanced (TR23) dataset show the  $WFA_{RM}$  outperforming PGSCM.

Table 7.21

*Relative quality DI (average, best DI, worst DI, standard deviation): WFA<sub>RM</sub> vs. PGSCM vs. DCPG using different datasets.*

Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	0.3909	0.3955	0.3642	0.0054
	PGSCM	0.1818	0.6402	0.0532	0.1763
	DCPG	<b>0.4494</b>	0.9944	0.0997	0.2863
Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	0.3673	0.3770	0.3412	0.0096
	PGSCM	<b>0.5964</b>	0.7188	0.3914	0.0723
	DCPG	0.3657	0.6937	0.1460	0.1579
Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	0.2857	0.2870	0.2816	0.0015
	PGSCM	<b>0.6743</b>	0.7453	0.5101	0.0591
	DCPG	0.4125	0.7973	0.1316	0.1642
Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	0.4747	0.5046	0.4626	0.0109
	PGSCM	0.4721	0.6442	0.2968	0.1268
	DCPG	<b>0.5094</b>	1.0303	0.1080	0.3291
Datasets	Algorithms	Average DI	Best DI	Worst DI	Standard Deviation
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	0.2824	0.2903	0.2719	0.0059
	PGSCM	<b>0.5364</b>	0.7113	0.4036	0.0870
	DCPG	0.3277	0.7490	0.0758	0.1667

*Note: the best value is highlighted in 'bold'.*

Furthermore, in Table 7.21, it can be observed that DCPG is the best in generating a high DI value against PGSCM and WFA<sub>RM</sub> in the Reuters and TR23 datasets. According to Youssef (2011), a higher DI value indicates the best quality clustering. The standard deviation of solution found by the WFA<sub>RM</sub> algorithm is smaller than DCPG and PGSCM in balanced and un-balanced datasets, which means that WFA<sub>RM</sub> can find a near optimal solution in most runs. Figure 7.14 shows a plotted graph of the result of the relative quality metrics, average DI, for WFA<sub>RM</sub>, PGSCM and DCPG.

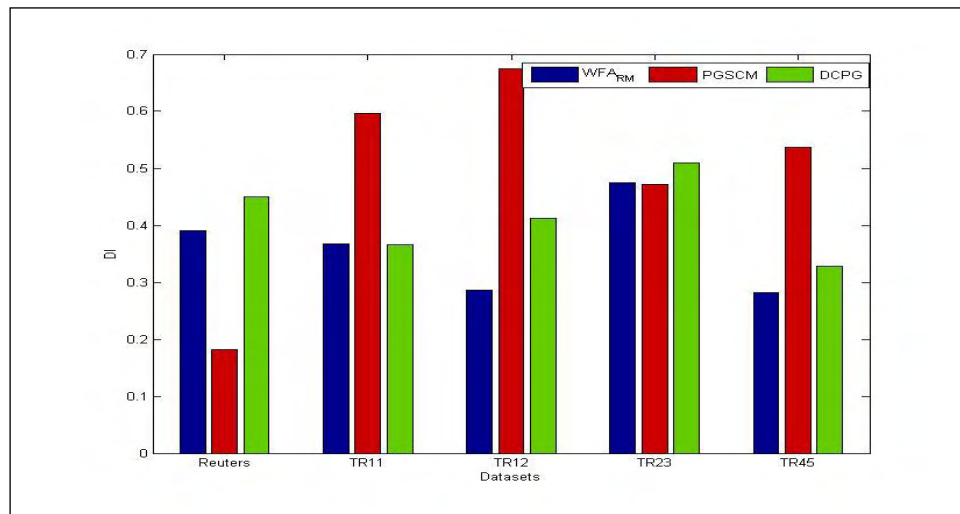


Figure 7.14. Average DI result: WFA<sub>RM</sub> vs. PGSCM vs. DCPG using different datasets.

The previous results indicate that WFA<sub>RM</sub> generates the best quality results in external performance metrics, namely Purity, F-measure and Entropy, against two dynamic methods, PGSCM and DCPG, as shown in Table 7.22. Whereas, for the internal and relative metrics, ADDC, DBI and DI, DCPG generates the best quality results in ADDC and DBI metrics, and PGSCM produces the best DI in most datasets.

Table 7.22

Summary of quality performance results: WFA<sub>RM</sub> vs. PGSCM vs. DCPG.

Datasets	Algorithms	External Metrics			Internal and Relative Metrics		
		Purity	F-measure	Entropy	ADDC	DBI	DI
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub>	✓	✓	✓			
	PGSCM						
	DCPG				✓	✓	✓

Table 7.22 continued

Datasets	Algorithms	External Metrics			Internal and Relative Metrics		
		Purity	F-measure	Entropy	ADDC	DBI	DI
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub>	✓	✓	✓			
	PGSCM						✓
	DCPG				✓	✓	
Datasets	Algorithms	External Metrics			Internal and Relative Metrics		
		Purity	F-measure	Entropy	ADDC	DBI	DI
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub>	✓	✓	✓			
	PGSCM						✓
	DCPG				✓	✓	
Datasets	Algorithms	External Metrics			Internal and Relative Metrics		
		Purity	F-measure	Entropy	ADDC	DBI	DI
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub>	✓		✓			
	PGSCM						
	DCPG		✓		✓	✓	✓
Datasets	Algorithms	External Metrics			Internal and Relative Metrics		
		Purity	F-measure	Entropy	ADDC	DBI	DI
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub>	✓	✓	✓			
	PGSCM						✓
	DCPG				✓	✓	

### 7.3.3 Evaluation Independent Samples T-test between WFA<sub>RM</sub> and Dynamic Methods

This section includes the analysis of Independent Samples T-test between WFA<sub>RM</sub> and PGSCM, and between WFA<sub>RM</sub> and DCPG. The hypotheses for Independent Samples T-test can be expressed in mathematical equivalents.

Null hypothesis  $H_0$ :  $Mean(WFA_{RM}) = Mean(\text{any dynamic methods})$

Alternative hypothesis  $H_1$ :  $Mean(WFA_{RM}) \neq Mean(\text{any dynamic methods})$

Where,  $Mean(WFA_{RM})$ ,  $Mean(\text{any dynamic methods})$  are the population means for WFA<sub>RM</sub> method and any dynamic method.

In Table 7.23, the associated P-value (2-tailed test) using thirty samples of average purity between WFA<sub>RM</sub> and PGSCM, and between WFA<sub>RM</sub> and DCPG are reported. Since the P-value is smaller than (0.05), the null hypothesis is rejected, so that the mean of purity metric for WFA<sub>RM</sub> and dynamic methods are the same and the alternative hypothesis is accepted to conclude that there is a significant difference in the mean of purity for WFA<sub>RM</sub> and any dynamic methods.

Table 7.23

*The P-value between  $WFA_{RM}$  & dynamic methods using average purity results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using Average Purity (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PGSCM	4.4059E-79	6.4431E-52
	$WFA_{RM}$ and DCPG	1.8942E-28	5.8698E-19
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PGSCM	4.9219E-57	7.1252E-53
	$WFA_{RM}$ and DCPG	1.8024E-33	1.9460E-25
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PGSCM	1.9896E-74	5.8563E-50
	$WFA_{RM}$ and DCPG	6.3600E-19	1.0330E-13
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PGSCM	7.3247E-81	1.9843E-73
	$WFA_{RM}$ and DCPG	4.5355E-17	1.1807E-12
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PGSCM	3.2665E-52	1.1143E-41
	$WFA_{RM}$ and DCPG	3.1525E-34	7.0949E-24

*Hint: The value highlighted in bold indicates not significance value.*

In Table 7.24, the associated P-value (2-tailed test) using thirty samples of average F-measure between  $WFA_{RM}$  and PGSCM, and between  $WFA_{RM}$  and DCPG are reported. As see in the table, all P-values are smaller than (0.05), so it can be concluded that the mean of purity metric for  $WFA_{RM}$  and dynamic methods are not similar. In this case, the null hypothesis is rejected and the alternative hypothesis is accepted. This excludes the P-value highlighted in bold in Table 7.24 between



WFA<sub>RM</sub> and DCPG in the TR23 dataset that indicates a not significant value, so in this situation, the null hypothesis is accepted.

Table 7.24

*The P-value between WFA<sub>RM</sub> & dynamic methods using average F-measure results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average F-measure (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	WFA <sub>RM</sub> and PGSCM	3.9511E-75	5.3917E-47
	WFA <sub>RM</sub> and DCPG	1.1051E-30	3.6710E-20
TR11 (414 documents and 9 classes)	WFA <sub>RM</sub> and PGSCM	6.0046E-35	3.6873E-31
	WFA <sub>RM</sub> and DCPG	6.2278E-21	3.1001E-20
TR12 (313 documents and 8 classes)	WFA <sub>RM</sub> and PGSCM	8.5602E-45	6.3043E-34
	WFA <sub>RM</sub> and DCPG	2.4536E-07	2.0968E-06
TR23 (204 documents and 6 classes)	WFA <sub>RM</sub> and PGSCM	2.2552E-20	1.3820E-15
	WFA <sub>RM</sub> and DCPG	<b>0.1892</b>	<b>0.1937</b>
TR45 (690 documents and 10 classes)	WFA <sub>RM</sub> and PGSCM	2.8176E-42	1.0575E-39
	WFA <sub>RM</sub> and DCPG	2.0262E-33	2.9935E-27

*Hint: The value highlighted in bold indicates not significance value.*

In Table 7.25, the associated P-values (2-tailed test) using thirty samples of average Entropy between WFA<sub>RM</sub> and PGSCM, and between WFA<sub>RM</sub> and DCPG are reported. From the table, all P-values are smaller than (0.05), so it can be concluded

that the mean of Entropy metric for  $WFA_{RM}$  and dynamic methods are not similar. In this case, the null hypothesis is rejected and the alternative hypothesis is accepted.

Table 7.25

*The P-value between  $WFA_{RM}$  & dynamic methods using average Entropy results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average Entropy (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PGSCM	9.8114E-87	1.0031E-86
	$WFA_{RM}$ and DCPG	6.0191E-27	2.9564E-18
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PGSCM	2.1753E-59	5.8230E-53
	$WFA_{RM}$ and DCPG	2.6886E-32	7.7915E-24
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PGSCM	2.2843E-72	3.4215E-51
	$WFA_{RM}$ and DCPG	7.0966E-23	5.7373E-16
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PGSCM	2.5191E-65	3.2335E-63
	$WFA_{RM}$ and DCPG	1.7835E-13	1.4925E-10
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PGSCM	1.4465E-65	1.4555E-61
	$WFA_{RM}$ and DCPG	2.8654E-31	2.4103E-21

*Hint: The value highlighted in bold indicates not significance value.*

In Table 7.26, the associated P-values (2-tailed test) using thirty samples of average ADDC between  $WFA_{RM}$  and PGSCM, and between  $WFA_{RM}$  and DCPG are reported. As can be seen in Table 7.25, all P-values are smaller than (0.05), so it can

be concluded that the mean of ADDC metric for  $WFA_{RM}$  and dynamic methods are not similar. In this case, the null hypothesis is rejected and the alternative hypothesis is accepted.

Table 7.26

*The P-value between  $WFA_{RM}$  & dynamic methods using average ADDC results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average ADDC (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PGSCM	1.6385E-64	1.2136E-39
	$WFA_{RM}$ and DCPG	4.1734E-22	1.7973E-15
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PGSCM	7.4487E-64	1.0987E-63
	$WFA_{RM}$ and DCPG	1.9985E-22	9.7693E-16
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PGSCM	1.4968E-80	1.3753E-58
	$WFA_{RM}$ and DCPG	4.5782E-14	8.3039E-11
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PGSCM	8.4728E-59	1.9617E-36
	$WFA_{RM}$ and DCPG	2.2122E-08	4.3436E-07
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PGSCM	8.9946E-41	1.2422E-27
	$WFA_{RM}$ and DCPG	1.2082E-14	3.6333E-11

*Hint: The value highlighted in bold indicates not significance value.*

In Table 7.27, the associated P-values (2-tailed test) using thirty samples of average DBI between  $WFA_{RM}$  and PGSCM, and between  $WFA_{RM}$  and DCPG are reported. From the table, all P-values are smaller than (0.05), so it can be concluded that the

mean of DBI metric for  $WFA_{RM}$  and dynamic methods are not similar. In this case, the null hypothesis is rejected and the alternative hypothesis is accepted. This excludes, the P-value highlighted in bold in Table 7.27 between  $WFA_{RM}$  and DCPG in the TR23 dataset that indicates a not significant value, so in this situation, the null hypothesis is accepted.

Table 7.27

*The P-value between  $WFA_{RM}$  & dynamic methods using average DBI results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average DBI (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PGSCM	8.0093E-07	5.8255E-06
	$WFA_{RM}$ and DCPG	8.0761E-16	7.2634E-12
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PGSCM	1.4588E-46	4.2161E-38
	$WFA_{RM}$ and DCPG	2.3359E-19	3.9018E-14
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PGSCM	3.2801E-70	9.9954E-48
	$WFA_{RM}$ and DCPG	3.2043E-30	6.3391E-20
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PGSCM	4.4507E-05	1.2669E-04
	$WFA_{RM}$ and DCPG	<b>0.1524</b>	<b>0.1577</b>
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PGSCM	2.7752E-17	7.5703E-13
	$WFA_{RM}$ and DCPG	1.1293E-14	3.4333E-11

*Hint: The value highlighted in bold indicates not significance value.*

In Table 7.28, the associated P-values (2-tailed test) using thirty samples of average DI between  $WFA_{RM}$  and PGSCM, and between  $WFA_{RM}$  and DCPG are reported. As seen in Table 7.28, the P-values between  $WFA_{RM}$  and PGSCM in most datasets (Reuters, TR11, TR12 and TR45) and between  $WFA_{RM}$  and DCPG in TR12 are smaller than (0.05), so it can be concluded that the mean of DI metric for  $WFA_{RM}$  and dynamic methods are not similar.

Table 7.28

*The P-value between  $WFA_{RM}$  & dynamic methods using average DI results (sig 2 tailed) with different datasets.*

Datasets	Algorithms	P-value using average DI (sig 2 tailed)	
		Equal variances assumed	Equal variances not assumed
Reuters (300 documents and 6 classes)	$WFA_{RM}$ and PGSCM	2.0658E-08	4.1150E-07
	$WFA_{RM}$ and DCPG	<b>0.2679</b>	<b>0.2725</b>
TR11 (414 documents and 9 classes)	$WFA_{RM}$ and PGSCM	1.8436E-24	4.2055E-17
	$WFA_{RM}$ and DCPG	<b>0.9583</b>	<b>0.9585</b>
TR12 (313 documents and 8 classes)	$WFA_{RM}$ and PGSCM	2.1910E-41	1.1876E-25
	$WFA_{RM}$ and DCPG	8.3758E-05	2.1282E-04
TR23 (204 documents and 6 classes)	$WFA_{RM}$ and PGSCM	<b>0.9099</b>	<b>0.9103</b>
	$WFA_{RM}$ and DCPG	<b>0.5665</b>	<b>0.5687</b>
TR45 (690 documents and 10 classes)	$WFA_{RM}$ and PGSCM	7.3197E-23	5.7341E-16
	$WFA_{RM}$ and DCPG	<b>0.1427</b>	<b>0.1481</b>

*Hint: The value highlighted in bold indicates not significance value.*

In this case, the null hypothesis is rejected and the alternative hypothesis is accepted. However, this excludes the P-value between  $WFA_{RM}$  and PGSCM in the TR23 and between  $WFA_{RM}$  and DCPG in most datasets (Reuters, TR11, TR23 and TR45) highlighted in bold in Table 7.28 are not significant values, so in this situation, the null hypothesis is accepted.

#### 7.4 Summary

This chapter includes the experimental results, evaluation and analysis of the proposed hybrid of  $WFA_R$  with the merging algorithm for text clustering ( $WFA_{RM}$ ) using different sizes of datasets. It is distributed in three sections.

Section one is the evaluation of the proposed  $WFA_{RM}$  against  $WFA_R$ , where each evaluation is conducted in three parts: 1) Evaluation of produced number of clusters, 2) Evaluation using performance metrics, namely external, internal and relative metrics such as Purity, F-measure, Entropy, ADDC, DBI and DI, and 3) Evaluation using a statistical analysis of paired samples T-test (Ross, 2010) that performs on the differences between the pair of  $WFA_{RM}$  and  $WFA_R$ .

Section Two is the evaluation of  $WFA_{RM}$  against state-of-the-art methods (static method) such as Bisect K-means (Murugesan & Zhang, 2011a, 2011b), K-means (Jain, 2010), PSO (Cui, Potok, & Palathingal, 2005), FAK-means (Tang, Fong, Yang, & Deb, 2012), BatK-means (Tang, Fong, Yang, & Deb, 2012). The evaluation is also conducted in three parts: 1) Evaluation of produced number of clusters, 2) Evaluation using performance metrics, and 3) Evaluation using a statistical analysis

of Independent Samples T-test (Ross, 2010) that performs on the differences between the pair of  $WFA_{RM}$  and one of any static methods.

Finally, in Section Three the proposed  $WFA_{RM}$  has been evaluated against state-of-the-art methods (dynamic method) such as Practical General Stochastic Clustering Method (PGSCM) (Tan, Ting, & Teng, 2011a), and Dynamic hybrid Genetic algorithm with Particle Swarm Optimization (DCPG) (Kuo, Syu, Chen, & Tien, 2012). The evaluation has the similar evaluation parts in Section Two.



## **CHAPTER EIGHT**

### **CONCLUSION AND FUTURE WORK**

The main goal of this thesis is to propose an Adaptive Firefly Algorithm for hierarchical text clustering. To achieve this goal, first, this research proposes a Weight-based Firefly algorithm (WFA) to identify the centers and its cluster. Later, this research introduces a document re-locating procedure into the Weight-based Firefly Algorithm to enhance the quality of the obtained clusters, and this is presented as WFA<sub>R</sub>. Finally, WFA<sub>R</sub> is improved by incorporating it with a cluster merging algorithm to discover the optimal number of clusters.

#### **8.1 Research Contribution**

In this thesis, there are four main contributions. The first contribution classifies the existing clustering methods and represent them in a taxonomy (refer to Figure 2.1). Each of the categories are explained in detail with some related works in Chapter Two.

The second contribution is the proposed Weight-based Firefly Algorithm that identifies the centers and the clusters for text documents, termed as WFA (illustrated in Figure 4.5). It has been included in Chapter Four. The WFA algorithm works in a dynamic manner without any specific information about the number of clusters. The fireflies in this algorithm operate with a normalized positioning in the search space. Experiments in Chapter Four reveal that the Weight-based Firefly Algorithm performed better than some of the state-of-the-art methods.



The third contribution includes the document re-locating procedure which was later integrated into the Weight-based Firefly Algorithm, and termed as  $WFA_R$  (illustrated in Figure 5.2). The proposed algorithm enhanced the quality of the obtained clusters. Experiments in Chapter Five indicated that the document re-locating procedure  $WFA_R$  performed better as compared to several existing methods.

In addition, this thesis also contributes  $WFA_{RM}$  that integrates  $WFA_R$  and a cluster merging algorithm (illustrated in Figure 6.1). The merging algorithm is based on the un-weighted pair group method with arithmetic mean (UPGMA), and termed as eUPGMA (refer to Figure 6.2). Experiments in Chapters Six and Seven demonstrated that the  $WFA_{RM}$  algorithm generates better results compared than existing static and dynamic methods and achieved the optimal number of clusters in most datasets used in the research

As a conclusion, the proposed Adaptive FA for hierarchical text clustering produces satisfied results which make it a competitive method in the text clustering field.

## 8.2 Future Work

In future, it is suggested to test the performance of the adaptive hierarchical text clustering,  $WFA_{RM}$ , on large text documents such as the ones with more than 1000 documents. Additionally, the research can also be performed on designing a distributed hierarchical text clustering algorithm.

Future works can also propose a dynamic function for the similarity threshold, where the existing similarity threshold is static predefined.

Furthermore, for future work, it is suggested to enhance the local search in the Weight-based Firefly Algorithm by integrating with single meta-heuristic algorithm such as Tabu search, and simulated annealing.



## REFERENCES

- 20NewsgroupsDataSet. (2006). <http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-4/text-learning/www/datasets.html>.
- Abshouri, A. A., & Bakhtiary, A. (2012). A new clustering method based on Firefly and KHM. *Journal of Communication and Computer*, 9, 387–391. Retrieved from <http://www.davidpublishing.com/davidpublishing/Upfile/6/4/2012/2012060483417489.pdf>
- Adaniya, M. H. A. C., Abr  o, T., & Proenc,a Jr., M. L. (2013). Anomaly Detection Using Metaheuristic Firefly Harmonic Clustering. *Journal of Networks*, 8(1), 82–91. Retrieved from doi:10.4304/jnw.8.1.82-91
- Aggarwal, C. C., & Zhai, C. X. (2012). A survey of text clustering algorithms. In *Mining Text Data*, Springer US (pp. 77–128). Retrieved from doi:10.1007/978-1-4614-3223-4\_4
- Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998). automatic subspace clustering of high dimensional data. *SIGMOD '98 Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, 94–105. Retrieved from doi: 10.1145/276304.276314
- Aliguliyev, R. M. (2009a). Clustering of document collection-A weighted approach. *Elsevier, Expert Systems with Applications*, 36(4), 7904–7916. Retrieved from doi: 10.1016/j.eswa.2008.11.017
- Aliguliyev, R. M. (2009b). Performance evaluation of density-based clustering methods. *Elsevier, Information Sciences*, 179(20), 3583–3602. Retrieved from doi: 10.1016/j.ins.2009.06.012
- Aljanabi, A. I. (2010). *Interacted multiple ant colonies for search stagnation problem*. College of Arts and Sciences. Universiti Utara Malaysia.
- Alsmadi, M. K. (2014). A hybrid firefly algorithm with fuzzy-c mean algorithm for MRI brain segmentation. *American Journal of Applied Sciences*, 11(9), 1676–1691.
- Amigo, E., Gonzalo, J., Artiles, J., & Verdejo, F. (2009). *A comparison of extrinsic clustering evaluation metrics based on formal constraints*. Springer, *Information Retrieval* (Vol. 12, pp. 461–486). Retrieved from doi: 10.1007/s10791-008-9066-8
- Anitha Elavarasi, S., Akilandeswari, J., & Sathiyabhama, B. (2011). A survey on partition clustering algorithms. *International Journal of Enterprise Computing*

and Business Systems, 1(1). Retrieved from Retrieved from at <http://www.ijecbs.com>

- Apostolopoulos, T., & Vlachos, A. (2011). Application of the Firefly Algorithm for Solving the Economic Emissions Load Dispatch Problem. *International Journal of Combinatorics*, Volume 201, 23 pages. Retrieved from doi:10.1155/2011/523806
- Bache, K., & Lichman, M. (2013). UCI Machine Learning Repository [<http://archive.ics.uci.edu/ml>]. Irvine: CA: University of California, School of Information and Computer Science.
- Banati, H., & Bajaj, M. (2013). Performance analysis of Firefly algorithm for data clustering. *Int. J. Swarm Intelligence*, 1(1), 19–35.
- Beasley, D., Bull, D. R., & Martin, R. R. (1993). An Overview of Genetic Algorithms : Part 1, Fundamentals. *University Computing*, 15(2), 58–69.
- Bojic, I., Podobnik, V., Ljubi, I., Jezic, G., & Kusek, M. (2012). A self-optimizing mobile network: Auto-tuning the network with firefly-synchronized agents. *Elsevier, Information Sciences*, 182(1), 77–92.
- Boley, D. (1998). Principal Direction Divisive Partitioning. *ACM, Data Mining and Knowledge Discovery*, 2(4), 325–344. Retrieved from doi: 10.1023/A:1009740529316
- Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. New York, NY: Oxford University Press, Santa Fe Institute Studies in the Sciences of Complexity.
- Bordogna, G., & Pasi, G. (2012). A quality driven Hierarchical Data Divisive Soft Clustering for information retrieval. *Elsevier, Knowledge-Based Systems*, 26, 9–19. Retrieved from doi:10.1016/j.knosys.2011.06.012
- Boussaïd, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Elsevier, Information Sciences*, 237, 82–117.
- Cao, D., & Yang, B. (2010). An improved k-medoids clustering algorithm. In *The 2nd International Conference on Computer and Automation Engineering (ICCAE)* (Vol. 3, pp. 132–135). Singapore: IEEE. Retrieved from doi: 10.1109/ICCAE.2010.5452085
- Chehreghani, M. H., Abolhassani, H., & Chehreghani, M. H. (2008). Improving density-based methods for hierarchical clustering of web pages. *Elsevier, Data & Knowledge Engineering*, 67(1), 30–50. Retrieved from doi: 10.1016/j.datak.2008.06.006

- Chen, T. S., Tsai, T. H., Chen, Y. T., Lin, C. C., Chen, R. C., Li, S. Y., & Chen, H. Y. (2005). A combined K-means and hierarchical clustering method for improving the clustering efficiency of microarray. In *Proceedings of intelligent signal processing and communication systems, IEEE* (pp. 405–408). IEEE. Retrieved from doi: 10.1109/ISPACS.2005.1595432
- Cui, X., Gao, J., & Potok, T. E. (2006). A flocking based algorithm for document clustering analysis. *Journal of Systems Architecture*, 52(8-9), 505–515.
- Cui, X., Potok, T. E., & Palathingal, P. (2005). Document Clustering using Particle Swarm Optimization. In *Proceedings 2005 IEEE Swarm Intelligence Symposium, SIS 2005*. (pp. 185–191). IEEEExplore. Retrieved from doi:10.1109/SIS.2005.1501621
- Das, S., Abraham, A., & Konar, A. (2008). Automatic Clustering Using an Improved Differential Evolution Algorithm. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 38(1), 218–237. doi:10.1109/TSMCA.2007.909595
- Das, S., Abraham, A., & Konar, A. (2009). *Metaheuristic Clustering*. Verlag Berlin Heidelberg: Springer. Retrieved from doi: 10.1007/978-3-540-93964-1
- Davies, D. L., & Bouldin, D. W. (1979). A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (Vol. PAMI-1, pp. 224–227). Retrieved from doi:10.1109/TPAMI.1979.4766909
- Demir, M., & Karci, A. (2015). Data Clustering on Breast Cancer Data Using Firefly Algorithm with Golden Ratio Method. *Advances in Electrical and Computer Engineering*, 15(2), 75–84. doi:10.4316/AECE.2015.02010
- Deneubourg, J. L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., & Chrétien, L. (1991). The dynamics of collective sorting: robot-like ants and ant-like robots. In *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats* (pp. 356–363). MIT Press Cambridge, MA, USA.
- Ding, Y., & Fu, X. (2012). The Research of Text Mining Based on Self-Organizing Maps. *Procedia Engineering*, 29(0), 537–541. doi:http://dx.doi.org/10.1016/j.proeng.2011.12.757
- Doding, G. (2002). *Computer Science in a Theory of Science Discourse*. Department of Computer Science. Malardalen University, Sweden.
- Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms*. Politecnico di Milano, Italie.

- Dorigo, M., & Gambardella, L. M. (1997). Ant colonies for the traveling salesman problem. *Elsevier, Biosystems*, 43(2), 73–81. Retrieved from doi:10.1016/S0303-2647(97)01708-5
- Dos Santos Coelho, L., de Andrade Bernert, D. L., & Mariani, V. C. (2011). A chaotic firefly algorithm applied to reliability-redundancy optimization. In *2011 IEEE Congress on Evolutionary Computation (CEC)* (pp. 517–521). New Orleans, LA. Retrieved from doi:10.1109/CEC.2011.5949662
- Dunn, J. (1974). Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4, 95–104. Retrieved from doi:10.1080/01969727408546059
- El-Abd, M., & Kamel, M. (2005). A taxonomy of cooperative search algorithms. *Hybrid Metaheuristics*, 3636, 32–41. Retrieved from doi:10.1007/11546245\_4
- El-Feghi, I., Errateeb, M., Ahmadi, M., & Sid-Ahmed, M. a. (2009). An adaptive ant-based clustering algorithm with improved environment perception. In *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics* (pp. 1431–1438). doi:10.1109/ICSMC.2009.5346291
- Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*. AAAI Press. (pp. 226–231).
- Falcon, R., Almeida, M., & Nayak, A. (2011). Fault Identification with Binary Adaptive Fireflies in Parallel and Distributed Systems. In *2011 IEEE Congress on Evolutionary Computation (CEC)*, (pp. 1359–1366). New Orleans, LA: IEEE Explore. Retrieved from doi:10.1109/CEC.2011.5949774
- Feng, L., Qiu, M. H., Wang, Y. X., Xiang, Q. L., Yang, Y. F., & Liu, K. (2010). A fast divisive clustering algorithm using an improved discrete particle swarm optimizer. *Elsevier, Pattern Recognition Letters*, 31(11), 1216–1225. Retrieved from doi: 10.1016/j.patrec.2010.04.001
- Fister, I., Jr, I. F., Yang, X. S., & Brest, J. (2013). A comprehensive review of Firefly Algorithms. *Elsevier, Swarm and Evolutionary Computation*, 13, 34–46.
- Folino, G., Forestiero, A., & Spezzano, G. (2009). An adaptive flocking algorithm for performing approximate clustering. *Information Sciences*, 179(18), 3059–3078.
- Fong, S., Deb, S., Yang, X. S., & Zhuang, Y. (2014). Towards Enhancement of Performance of K-Means Clustering Using Nature-Inspired Optimization Algorithms. *The Scientific World Journal*, 2014(564829), 16 pages.

- Forsati, R., Mahdavi, M., Shamsfard, M., & Meybodi, M. R. (2013). Efficient stochastic algorithms for document clustering. *Elsevier, Information Sciences*, 220, 269–291. Retrieved from doi: 10.1016/j.ins.2012.07.025
- Gil-Garcia, R., & Pons-Porrata, A. (2010). Dynamic hierarchical algorithms for document clustering. *Elsevier, Pattern Recognition Letters*, 31(6), 469–477. Retrieved from doi: 10.1016/j.patrec.2009.11.011
- Glass, A. (2011). *Explanation of Adaptive Systems*. Stanford University.
- Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13(No.5), 533–549.
- Gu, J., Zhou, J., & Chen, X. (2009). An Enhancement of K-means Clustering Algorithm. In *IEEE, International Conference on Business Intelligence and Financial Engineering* (pp. 237–240). Beijing: IEEE. Retrieved from doi: 10.1109/BIFE.2009.204
- Guan, R., Shi, X., Marchese, M., Yang, C., & Liang, Y. (2011). Text Clustering with Seeds Affinity Propagation. *IEEE Transactions on Knowledge and Data Engineering*, 23(4), 627–637. Retrieved from doi: 10.1109/TKDE.2010.144
- Gupta, P., & Sharma, A. K. (2010). A framework for hierarchical clustering based indexing in search engines. In *Proceedings of 1st International Conference on Parallel, Distributed and Grid Computing (PDGC - 2010)* (pp. 372–377). Solan: IEEE. Retrieved from doi: 10.1109/PDGC.2010.5679966
- Han, J., & Kamber, M. (2006). *Data mining: Concepts and techniques (2nd ed.)*. San Francisco: Morgan Kaufman.
- Han, J., Kamber, M., & Pei, J. (2011). *Data Mining: Concepts and Techniques, 3rd edition. The Morgan Kaufmann Series in Data Management Systems* (p. 744 pages). Morgan Kaufmann.
- Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A K-Means Clustering Algorithm. *JStor, Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(No.1). Retrieved from <http://www.jstor.org/stable/2346830>
- Hassanzadeh, T., Faez, K., & Seyfi, G. (2012). A Speech Recognition System Based on Structure Equivalent Fuzzy Neural Network Trained by Firefly Algorithm. In *International Conference on Biomedical Engineering (ICoBE)* (pp. 63–67). Penang: IEEE Explore. Retrieved from doi:10.1109/ICoBE.2012.6178956
- Hassanzadeh, T., & Meybodi, M. R. (2012). A new hybrid approach for data clustering using Firefly algorithm and k-means. In *The 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP 2012), IEEE* (pp. 7–11). Retrieved from doi: 10.1109/AISP.2012.6313708

- Hassanzadeh, T., Vojodi, H., & Moghadam, A. M. E. (2011). An Image Segmentation Approach Based on Maximum Variance Intra-Cluster Method and Firefly Algorithm. In *Seventh International Conference on Natural Computation (ICNC)* (Vol. 3, pp. 1817–1821). Shanghai: IEEE Explore. Retrieved from doi:10.1109/ICNC.2011.6022379
- Hatamlou, A., Abdullah, S., & Nezamabadi-pour, H. (2012). A combined approach for clustering based on K-means and gravitational search algorithms. *Elsevier, Swarm and Evolutionary Computation*, 6, 47–52. Retrieved from doi: 10.1016/j.swevo.2012.02.003
- He, Y., Hui, S. C., & Sim, Y. (2006). Anovel ant-based clustering approach document clustering. *Information Retrieval Technology*, 4182, 537–544.
- Hinneburg, A., & Keim, D. (1999). Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering. In *Proceedings of the 25th International Conference on Very Large Data Bases* (pp. 506–517). Morgan Kaufmann Publishers Inc.
- Holland, J. (1992). *Adaptation in Natural and Artificial Systems : An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence* (p. 211). Cambridge, MA, USA.
- Horng, M. H., & Jiang, T. W. (2010). Multilevel Image Thresholding Selection based on the Firefly Algorithm. In *7th International Conference on Ubiquitous Intelligence & Computing and 7th International Conference on Autonomic & Trusted Computing (UIC/ATC)*, (pp. 58–63). Xian, Shaanxi: IEEE Explore. Retrieved from doi:10.1109/UIC-ATC.2010.47
- Hu, G., Zhou, S., Guan, J., & Hu, X. (2008). Towards effective document clustering: A constrained K-means based approach. *Elsevier , Information Processing & Management*, 44(4), 1397–1409. Retrieved from doi: 10.1016/j.ipm.2008.03.001
- Ilango, M., & Mohan, V. (2010). A Survey of Grid Based Clustering Algorithms. *International Journal of Engineering Science and Technology*, 2(8), 3441–3446.
- Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Elsevier, Pattern Recognition Letters*, 31(8), 651–666. Retrieved from doi: 10.1016/j.patrec.2009.09.011
- Jensi, R., & Jiji, D. G. W. (2013). A Survey on optimization approaches to text document clustering. *International Journal on Computational Sciences & Applications (IJCSA)*, 3(6), 31–44.



- Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transaction of the ASME-Journal of Basic Engineering*, 82 (series, 35–45).
- Kao, Y., & Lee, S.-Y. (2009). Combining K-means and particle swarm optimization for dynamic data clustering problems. In *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on* (Vol. 1, pp. 757–761).
- Karypis, G. (2002). *CLUTO a clustering toolkit*, Technical Report 02-017. Dept. of Computer Science, University of Minnesota. Retrieved from Available at <http://glaros.dtc.umn.edu/gkhome/views/cluto>
- Karypis, G., Han, E. H., & Kumar, V. (1999). Chameleon: Hierarchical Clustering Using Dynamic Modeling. *IEEE Computer Society*, 32(8), 68–75. Retrieved from doi: 10.1109/2.781637
- Kashef, R., & Kamel, M. (2010). Cooperative clustering. *Elsevier, Pattern Recognition*, 43(6), 2315–2329. Retrieved from doi: 10.1016/j.patcog.2009.12.018
- Kashef, R., & Kamel, M. S. (2009). Enhanced bisecting k-means clustering using intermediate cooperation. *Elsevier, Pattern Recognition*, 42(11), 2557–2569.
- Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. *Proceedings of IEEE International Conference on Neural Networks IV*. Perth, WA: IEEE. Retrieved from doi:10.1109/ICNN.1995.488968
- Kennedy, J. F., & Eberhart, R. C. (2001). *Swarm intelligence* (p. 512). Morgan Kaufmann.
- Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. *New Series*, 220(No. 4598), 671–680.
- Kohonen, T. (1998). The self-organizing map. *Elsevier, Neurocomputing*, 21(1-3), 1–6. Retrieved from doi: 10.1016/S0925-2312(98)00030-7
- Kohonen, T. (2001). *Self organizing map 3rd ed*. Springer-Verlag Berlin Heidelberg New York.
- Kuo, R. J., Syu, Y. J., Chen, Z., & Tien, F. C. (2012). Integration of particle swarm optimization and genetic algorithm for dynamic clustering. *Elsevier, Information Sciences*, 195, 124–140.
- Kuo, R. J., & Zulvia, F. E. (2013). automatic clustering using an improved particle swarm optimization. *Journal of Industrial and Intelligent Information*, 1(1), 46–51.

- Lahane, S. V., Kharat, M. U., & Halgaonkar, P. S. (2012). Divisive approach of Clustering for Educational Data. In *Fifth International Conference on Emerging Trends in Engineering and Technology* (pp. 191–195). Himeji: IEEE. Retrieved from doi:10.1109/ICETET.2012.55
- Lee, C. Y., & Antonsson, E. K. (2000). Dynamic partitional clustering using evolution strategies. In *IEEE* (Vol. 4, pp. 2716–2721).
- Lewis, D. (1999). The reuters-21578 text categorization test collection. Retrieved from Available online at :<http://kdd.ics.uci.edu/database/reuters21578/reuters21578.html>
- Liu, Y. C., Wu, C., & Liu, M. (2011). Research of fast SOM clustering for text information. *Elsevier, Expert Systems with Applications*, 38(8), 9325–9333. Retrieved from doi: 10.1016/j.eswa.2011.01.126
- Liu, Y. C., Wu, X., & Shen, Y. (2011). Automatic clustering using genetic algorithms. *Elsevier, Applied Mathematics and Computation*, 218(4), 1267–1279.
- Lu, Y., Wang, S., Li, S., & Zhou, C. (2009). Text Clustering via Particle Swarm Optimization. In *Swarm Intelligence Symposium, 2009. SIS '09. IEEE* (pp. 45–51). Nashville, TN: IEEEExplore. Retrieved from doi:10.1109/SIS.2009.4937843
- Luo, C., Li, Y., & Chung, S. M. (2009). Text document clustering based on neighbors. *Elsevier, Data & Knowledge Engineering*, 68(11), 1271–1288. Retrieved from doi: 10.1016/j.datak.2009.06.007
- MacQueen, J. B. (1967). Kmeans Some Methods for classification and Analysis of Multivariate Observations. *5th Berkeley Symposium on Mathematical Statistics and Probability 1967*, 1(233), 281–297. doi:citeulike-article-id:6083430
- Mahmuddin, M. (2008). *Optimisation using Bees algorithm on unlabelled data problems*. Manufacturing engineering centre. Cardif university, Cardiff, UK.
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*, 1 ed. New York, USA: Cambridge University Press.
- Martens, D., Backer, M. D., Haesen, R., Vanthienen, J., Snoeck, M., & Baesens, B. (2007). Classification With Ant Colony Optimization. *IEEE Transactions on Evolutionary Computation*, 11(5), 651–665. Retrieved from doi: 10.1109/TEVC.2006.890229
- Meghabghab, G., & Kandel, A. (2008). *Search engines, link analysis, and user's web behaviour* (Vol. 99). Springer Berlin Heidelberg. Retrieved from doi:10.1007/978-3-540-77469-3

- Miner, G., Elder, J., Fast, A., Hill, T., Nisbet, R., & Delen, D. (2012). *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications, 1st ed.* Elsevier.
- Mishra, B. K., Nayak, N. R., Rath, A., & Swain, S. (2012). Far Efficient K-Means Clustering Algorithm. In *Proceedings of the International Conference on Advances in Computing, Communications and Informatics* (pp. 106–110). ACM. Retrieved from doi:10.1145/2345396.2345414
- Muñoz, D. M., Llanos, C. H., Coelho, L. D. S., & Ayala-Rincon, M. (2011). Opposition-based shuffled PSO with passive congregation applied to FM matching synthesis. In *2011 IEEE Congress on Evolutionary Computation (CEC)*, (pp. 2775–2781). New Orleans, LA: IEEE Xplore. Retrieved from doi:10.1109/CEC.2011.5949966
- Murugesan, K., & Zhang, J. (2011a). Hybrid Bisect K-means clustering algorithm. In *International Conference on Business Computing and Global Informatization* (pp. 216–219). Retrieved from doi:10.1109/BCGIN.2011.62
- Murugesan, K., & Zhang, J. (2011b). *Hybrid hierarchical clustering: An experimental analysis* (p. 26). university of Kentucky.
- Nandy, S., Sarkar, P. P., & Das, A. (2012). Analysis of a Nature Inspired Firefly Algorithm based Back-propagation Neural Network Training. *International Journal of Computer Applications*, 43(22), 8–16. Retrieved from doi:10.5120/6401-8339
- Pelleg, M., & Moore, A. (2000). X-means: Extending K-means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 727–734). Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.
- Picarougne, F., Azzag, H., Venturini, G., & Guinot, C. (2007). A New Approach of Data Clustering Using a Flock of Agents. *Evolutionary Computation, Cambridge: MIT Press* (2007), 15(3), 345–367.
- Poomagal, S., & Hamsapriya, T. (2011). Optimized k-means clustering with intelligent initial centroid selection for web search using URL and tag contents. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics* (pp. 1–8). Sogndal, Norway: ACM. Retrieved from doi:10.1145/1988688.1988764
- Pop, C. B., Chifu, V. R., Salomie, I., Baico, R. B., Dinsoreanu, M., & Copil, G. (2011). A Hybrid Firefly-inspired Approach for Optimal Semantic Web Service Composition. *Scientific International Journal for Parallel and Distributed Computing*, 12(3), 363–369. Retrieved from retrived from: <http://www.scpe.org/index.php/scpe/article/view/730/0>

- Rafsanjani, M. K., Varzaneh, Z. A., & Chukanlo, N. E. (2012). A survey of hierarchical clustering algorithms. *The Journal of Mathematics and Computer Science, TJMCS*, 5, No. 3, 229–240. Retrieved from Available online at : <http://www.TJMCS.com>
- Rana, S., Jasola, S., & Kumar, R. (2010). A hybrid sequential approach for data clustering using K-Means and particle swarm optimization algorithm. *International Journal of Engineering, Science and Technology*, 2, No.6, 167–176. Retrieved from Available online at : <http://www.ajol.info/index.php/ijest/article/view/63708>
- Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009). GSA: A Gravitational Search Algorithm. *Elsevier, Information Sciences*, 179(13), 2232–2248.
- Rokach, L., & Maimon, O. (2005). *Clustering Methods, Data Mining and Knowledge Discovery Handbook*. Springer (pp. 321–352.).
- Ross, S. M. (2010). *Introductory Statistics*. Elsevier Science. Retrieved from <http://books.google.com.my/books?id=ZKswvkqhygYC>
- Rothlauf, F. (2011). *Design of Modern Heuristics Principles and Application*. Springer-Verlag Berlin Heidelberg. Retrieved from doi:10.1007/978-3-450-72962-4
- Rui, T., Fong, S., Yang, X. S., & Deb, S. (2012). Nature-Inspired Clustering Algorithms for Web Intelligence Data. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT)* (Vol. 3, pp. 147–153). Macau. Retrieved from doi:10.1109/WI-IAT.2012.83
- Sander, J. (2010). Density-Based Clustering. *Encyclopedia of Machine Learning SE - 211*. Springer US DA - 2010/01/01. Retrieved from doi:10.1007/978-0-387-30164-8\_211
- Sarkar, M., Yegnanarayana, B., & Khemani, D. (1997). A clustering algorithm using an evolutionary programming-based approach. *Elsevier, Pattern Recognition*, 18(10), 975–986.
- Sayadi, M. K., Hafezalkotob, A., & Naini, S. G. J. (2013). Firefly-inspired algorithm for discrete optimization problems: An application to manufacturing cell formation. *Elsevier, Journal of Manufacturing Systems*, 32(1), 78–84.
- Sayed, A., Hacid, H., & Zighed, D. (2009). Exploring validity indices for clustering textual data. In *Mining Complex Data*, 165, 281–300.
- Senthilnath, J., Omkar, S. N., & Mani, V. (2011). Clustering using firefly algorithm: Performance study. *Elsevier, Swarm and Evolutionary Computation*, 1(3), 164–171. Retrieved from doi: 10.1016/j.swevo.2011.06.003

- Shannon, C. E. (1948). A Mathematical theory of communication. *Bell System Technical Journal*, 27, 379–423, 623–656,. Retrieved from Retrieved from: <http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf>
- Singh, R. V., & Bhatia, M. P. S. (2011). Data clustering with modified K-means algorithm. In *International Conference on Recent Trends in Information Technology (ICRTIT)* (pp. 717–721). Chennai, Tamil Nadu: IEEE. Retrieved from doi:10.1109/ICRTIT.2011.5972376
- Stahlbock, R., Crone, S. F., & Lessmann, S. (2010). *Data Mining Special Issue in Annals of Information Systems* (Vol. 8). Springer US. Retrieved from doi: 10.1007/978-1-4419-1280-0
- Tan, P. N., Steinbach, M., & Kumar, V. (2006). *Introduction to data mining*. Pearson Education, Addition Wesley.
- Tan, S. C. (2012). Simplifying and improving swarm based clustering. In *IEEE Congress on Evolutionary Computation (CEC)* (pp. 1–8). Brisbane, QLD: IEEE.
- Tan, S. C., Ting, K. M., & Teng, S. W. (2011a). A general stochastic clustering method for automatic cluster discovery. *Elsevier, Pattern Recognition*, 44(10-11), 2786–2799.
- Tan, S. C., Ting, K. M., & Teng, S. W. (2011b). Simplifying and improving ant-based clustering. In *Procedia computer science* (pp. 46–55).
- Tang, R., Fong, S., Yang, X. S., & Deb, S. (2012). Integrating nature-inspired optimization algorithms to K-means clustering. In *Seventh International Conference on Digital Information Management (ICDIM), 2012* (pp. 116–123). Macau: IEEE. Retrieved from doi:10.1109/ICDIM.2012.6360145
- Toreini, E., & Mehrnejad, M. (2011). Clustering Data with Particle Swarm Optimization Using a New Fitness. In *2011 3rd Conference on Data Mining and Optimization (DMO)* (pp. 266–270). Putrajaya: IEEEExplore. Retrieved from doi:10.1109/DMO.2011.5976539
- TREC. (1999). Text REtrieval Conference (TREC). Retrieved from Available online at :<http://trec.nist.gov/>
- Van der Merwe, D. W., & Engelbrecht, A. P. (2003). Data clustering using particle swarm optimization. In *The 2003 Congress on Evolutionary Computation, 2003. CEC '03.* (Vol. 1, pp. 215–220). Retrieved from doi:10.1109/CEC.2003.1299577
- Vijayalakshmi, M., MCA, M., & Devi, M. R. (2012). A survey of different issue of different clustering algorithms used in large data sets. *International Journal of*

*Advance Research in Computer Science and Software Engineering*, 2(3), 305–307. Retrieved from Available online at : <http://www.ijarcse.com>

- Wang, H., Yang, X., Zhang, J., Zhang, M., Bai, X., Yin, W., & Dong, J. (2011). BP neural network model based on cluster analysis for wind power prediction. In *2011 IEEE International Conference on Service Operations, Logistics, and Informatics (SOLI)* (pp. 278–280). Beijing: IEEE Xplore. Retrieved from doi:10.1109/SOLI.2011.5986570
- Wang, W., Yang, J., & Muntz, R. (1997). STING : A Statistical Information Grid Approach to Spatial Data Mining. In *VLDB '97 Proceedings of the 23rd International Conference on Very Large Data Bases* (pp. 186–195). Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.
- Wang, X., Shen, J., & Tang, H. (2009). Novel hybrid document clustering algorithm based on Ant Colony and agglomerate. In *Second International Symposium on Knowledge Acquisition and Modeling* (Vol. 3, pp. 65–68). Wuhan: IEEE computer society. Retrieved from doi:10.1109/KAM.2009.182
- Wang, Z., Liu, Z., Chen, D., & Tang, K. (2011). A New Partitioning Based Algorithm For Document Clustering. In *Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)* (Vol. 3, pp. 1741–1745). Shanghai: IEEE. Retrieved from doi: 10.1109/FSKD.2011.6019857
- Wilson, H. G., Boots, B., & Millward, A. A. (2002). A comparison of hierarchical and partitional clustering techniques for multispectral image classification. In *IEEE* (Vol. 3, pp. 1624–1626). IEEE International Geoscience and Remote Sensing Symposium, 2002. IGARSS. Retrieved from doi: 10.1109/IGARSS.2002.1026201
- Xinwu, L. (2010). Research on Text Clustering Algorithm Based on Improved K-means. In *International Conference On Computer Design And Applications (ICCDAA 2010)* (Vol. 4, pp. V4–573 – V4–576). Qinhuangdao: IEEE. Retrieved from doi: 10.1109/ICCDAA.2010.5540727
- Xu, Y. (2005). Hybrid clustering with application to web mining. In *Proceedings of the International Conference on Active Media Technology (AMT 2005)*. (pp. 574–578). Retrieved from doi: 10.1109/AMT.2005.1505425
- Yang, H. (2010). A Document Clustering Algorithm for Web Search Engine Retrieval System. In *International Conference on e-Education, e-Business, e-Management, and e-Learning, 2010. IC4E '10* (pp. 383–386). Sanya: IEEE. Retrieved from doi:10.1109/IC4E.2010.72
- Yang, X. S. (2009). Firefly Algorithms for Multimodal Optimization. In O. Watanabe & T. Zeugmann (Eds.), *Stochastic Algorithms: Foundations and*

- Applications* (pp. 169–178). Springer Berlin Heidelberg. doi:10.1007/978-3-642-04944-6\_14
- Yang, X. S. (2010a). Firefly Algorithm, Stochastic Test Functions and Design Optimisation. *Int. J. Bio-Inspired Computation*, 2(2), 78–84.
- Yang, X. S. (2010b). *Nature-inspired metaheuristic algorithms 2nd edition*. United Kingdom: Luniver press.
- Yang, X. S., & He, X. (2013). Firefly algorithm: recent advances and applications. *Int. J. Swarm Intelligence*, 1(1), 36–50. Retrieved from doi:10.1504/IJSI.2013.055801
- Yang, X. S., Hosseini, S. S. S., & Gandomi, A. H. (2012). Firefly Algorithm for solving non-convex economic dispatch problems with valve loading effect. *Elsevier, Applied Soft Computing*, 12(3), 1180–1186. Retrieved from doi:10.1016/j.asoc.2011.09.017
- Yao, M., Pi, D., & Cong, X. (2012). Chinese text clustering algorithm based k-means. In *2012 International Conference on Medical Physics and Biomedical Engineering (ICMPBE2012)* (Vol. 33, pp. 301–307). Elsevier. Retrieved from doi: 10.1016/j.phpro.2012.05.066, Available online at www.sciencedirect.com
- Ye, N., Gauch, S., Wang, Q., & Luong, H. (2010). An adaptive ontology based hierarchical browsing system for CiteSeerX. In *Second International Conference on Knowledge and Systems Engineering (KSE), IEEE* (pp. 203–208). Retrieved from doi: 10.1109/KSE.2010.32
- Yin, Y., Kaku, I., Tang, J., & Zhu, J. (2011). *Data Mining Concepts, Methods and Application in Management and Engineering Design*. Springer-Verlag London.
- Youssef, S. M. (2011). A New Hybrid Evolutionary-based Data Clustering Using Fuzzy Particle Swarm Optimization. In *23rd IEEE International Conference on Tools with Artificial Intelligence* (pp. 717–724). IEEE. Retrieved from doi: 10.1109/ICTAI.2011.113
- Yue, S., Wei, M., Wang, J. S., & Wang, H. (2008). A general grid-clustering approach. In *Elsevier, Pattern Recognition Letters* (Vol. 29, pp. 1372–1384). Retrieved from doi: 10.1016/j.patrec.2008.02.019
- Yujian, L., & Liye, X. (2010). Unweighted Multiple Group Method with Arithmetic Mean. In *IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)* (pp. 830–834). Changsha: IEEE. Retrieved from doi:10.1109/BICTA.2010.5645232

- Yunrong, X., & Liangzhong, J. (2009). Water quality prediction using LS-SVM and particle swarm optimization. In *Knowledge Discovery and Data Mining, 2009. WKDD 2009. Second International Workshop on* (pp. 900–904).
- Zhang, L., & Cao, Q. (2011). A novel ant-based clustering algorithm using the kernel method. *Elsevier, Information Sciences*, 181(20), 4658–4672. Retrieved from doi:10.1016/j.ins.2010.11.005
- Zhang, L., Cao, Q., & Lee, J. (2013). A novel ant-based clustering algorithm using Renyi entropy. *Elsevier, Applied Soft Computing*, 13(5), 2643–2657. Retrieved from doi:10.1016/j.asoc.2012.11.022
- Zhang, W., Yoshida, T., Tang, X., & Wang, Q. (2010). Text clustering using frequent itemsets. *Elsevier, Knowledge-Based Systems*, 23(5), 379–388. Retrieved from doi:10.1016/j.knosys.2010.01.011
- Zhao, Y., Cao, J., Zhang, C., & Zhang, S. (2011). Enhancing grid-density based clustering for high dimensional data. *Elsevier, Journal of Systems and Software*, 84(9), 1524–1539. Retrieved from doi:10.1016/j.jss.2011.02.047
- Zhao, Y., & Karypis, G. (2001). *Criterion functions for document clustering: Experiments and analysis*.
- Zhong, J., Liu, L., & Li, Z. (2010). A novel clustering algorithm based on gravity and cluster merging. *Advanced Data Mining and Applications*, 6440, 302–309. Retrieved from doi:10.1007/978-3-642-17316-5\_30
- Zhu, Y., Fung, B. C. M., Mu, D., & Li, Y. (2008). An efficient hybrid hierarchical document clustering method. In *IEEE, Fifth international conference on Fuzzy systems and knowledge discovery* (Vol. 2, pp. 395–399). Retrieved from doi:10.1109/FSKD.2008.159