

IT UNIVERSITY OF COPENHAGEN

# Clustering Player Behaviors in Data Streams with K-Means in Map-Reduce

by

Sigurdur Karl Magnusson

A thesis submitted in partial fulfillment for the  
degree of Master of Science in IT

in the  
Computer Science  
Software Development and Technology

Supervisors  
Julian Togelius, Rasmus Pagh

April 2013

*“Any intelligent fool can make things bigger, more complex, and more violent. It takes a touch of genius – and a lot of courage – to move in the opposite direction.”*

Albert Einstein

# Abstract

The abstract is a short summary of the thesis. It announces in a brief and concise way the scientific goals, methods, and most important results. The chapter “conclusions” is not equivalent to the abstract! Nevertheless, the abstract may contain concluding remarks. The abstract should not be discursive. Hence, it cannot summarize all aspects of the thesis in very detail. Nothing should appear in an abstract that is not also covered in the body of the thesis itself. Hence, the abstract should be the last part of the thesis to be compiled by the author.

A good abstract has the following properties: *Comprehensive:* All major parts of the main text must also appear in the abstract. *Precise:* Results, interpretations, and opinions must not differ from the ones in the main text. Avoid even subtle shifts in emphasis. *Objective:* It may contain evaluative components, but it must not seem judgemental, even if the thesis topic raises controversial issues. *Concise:* It should only contain the most important results. It should not exceed 300–500 words or about one page. *Intelligible:* It should only contain widely-used terms. It should not contain equations and citations. Try to avoid symbols and acronyms (or at least explain them). *Informative:* The reader should be able to quickly evaluate, whether or not the thesis is relevant for his/her work.

An Example: The objective was to determine whether ... (*question/goal*). For this purpose, ... was ... (*methodology*). It was found that ... (*results*). The results demonstrate that ... (*answer*).

# *Acknowledgements*

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

# Contents

<b>Abstract</b>	<b>ii</b>
<b>Acknowledgements</b>	<b>iii</b>
<b>List of Figures</b>	<b>vi</b>
<b>List of Tables</b>	<b>vii</b>
<b>Abbreviations</b>	<b>viii</b>
<b>1 Chapter Title Here</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Problem Statement . . . . .	2
1.3 Contributions . . . . .	2
1.4 Project Outline . . . . .	2
<b>2 Background Theory</b>	<b>3</b>
2.1 Player Behaviour . . . . .	3
2.1.1 Game Metric . . . . .	3
2.1.2 Features . . . . .	3
2.2 Clustering . . . . .	3
2.2.1 K-Means . . . . .	3
2.2.2 Streaming and Data Streams . . . . .	3
2.3 Map-Reduce and Hadoop . . . . .	3
<b>3 Related and Recent Work</b>	<b>4</b>
3.1 Clustering Player Behaviors . . . . .	4
3.2 Clustering Large Data . . . . .	5
3.2.1 Related work vs the study . . . . .	8
<b>4 Methodology</b>	<b>9</b>
4.1 Game Events Data . . . . .	9
4.1.1 Building Features . . . . .	9
4.2 K-Means in Map-Reduce . . . . .	9
4.2.1 Euclidean distance . . . . .	9
4.2.2 Normalized Histograms . . . . .	9
4.3 Experimental Set-Up . . . . .	9

---

<b>5</b>	<b>Results and Discussion</b>	<b>10</b>
5.1	Results . . . . .	10
5.2	Discussion . . . . .	10
<b>6</b>	<b>Conclusions</b>	<b>11</b>
<b>A</b>	<b>Appendix Title Here</b>	<b>12</b>
	<b>Bibliography</b>	<b>13</b>

# List of Figures

# List of Tables



# Abbreviations

**LAH** List Abbreviations **Here**

# Chapter 1

## Chapter Title Here

### 1.1 Motivation

We live in a time of data, where data is growing at an amazing rate. Data is everywhere around us and companies are generating more and more data. Storing data is cheaper than ever and many smaller and mid-sized companies are buying storage subscriptions at other bigger companies that offer "Storage as a Service" (SaaS). A SaaS company offers e.g. data reliability and durability by storing critical data in multiple facilities and on multiple devices. Having all this data does not give a meaning by itself, it needs to be analysed and interpret to give a business value.

*"Information is the oil of the 21st century, and analytics is the combustion engine."*

Peter Sondergaard, senior VP at Gartner

The need to analyse all this data have caused many new companies to rise up that are specialised in analysing large quantities of data, extracting knowledge and converting it to a business value. Many of those companies offer "Analytics as a Service" (AaaS) that offer an analytical software to discover trends and unknown patterns in the data.

## 1.2 Problem Statement

GameAnalytics.com is a cloud hosted service for collecting, analyzing and reporting game metrics. Working with large quantities of game metric data that needs to be analysed and processed efficiently.

A valuable application for GA is to design and implement a scalable streaming version of a clustering algorithm. That can read a large data in mini-batches into memory and cluster the data incrementally.

The goal would be to incrementally find clusters efficiently in a streaming data, showing predominant characteristics of human behavior, e.g. the hardcore players, casual players, people who didn't understand the game, etc.

## 1.3 Contributions

- Clustering Player Behaviours using Map-Reduce framework
- Software and knowledge to GameAnalytics for further development

## 1.4 Project Outline

## Chapter 2

# Background Theory

### 2.1 Player Behaviour

#### 2.1.1 Game Metric

#### 2.1.2 Features

### 2.2 Clustering

#### 2.2.1 K-Means

#### 2.2.2 Streaming and Data Streams

### 2.3 Map-Reduce and Hadoop

## Chapter 3

# Related and Recent Work

Clustering player behaviors and large data sets have been researched actively in the recent years. In our work we focus on processing a large amount of data of finite length in parallel using k-means in Map-Reduce and incrementally cluster new data sets arriving each day. The behavior of the data can change from day to day like when dealing with an endless stream of data. In subsequent sections some of the recent and related work are given a short introduction.

### 3.1 Clustering Player Behaviors

Many researches have been done on clustering and predicting player behaviors over the last years to get a better understanding which kind of user behaviors are to be found when playing a game that can be actionable for game developers [1–5]. User behavior analysis has becoming increasingly popular in the recent years because of rise of the free-to-play (F2P) genre games in *Facebook* and *Google Play* where populations can be in millions creating complex game interactions [4, 6]. Playing these games are free and many are of persistent nature where the world in the game continues when a player exits. To be profitable these games drive their revenue via micro transactions, e.g. players buying upgrades or virtual items in game for real money [4, 6–8]. Major game publishers have also been collecting and analyzing large scale of behavior telemetry data but details of their methods are kept confidential [9, 10]. Most available research work

is case based where a specific algorithm is applied to a specific game and commercial game data sets has only become accessible recently for academic researchers [10].

**Thurau world of warcraft guilds** A work of analyzing social groups in the popular Massively Multiplayer Online Role-Playing Game (MMORPG) World of Warcraft was done by Thurau and Bauckhage [CITE]. They analyzed how groups (guilds) evolve over time from both American and European based guilds. Their paper is the first study analyzing such amount of data in a MMORPG, analyzing large-scale data from 18 million players belonging in 1.4 million groups over a period of 4 years. Convex-Hull Non Negative Matrix Factorization (CH-NMF) technique was applied to the data and the results show that there are no significant cultural difference in formation processes of guilds from either the US or the EU. Interpretability of CH-NMF was more distinguishable and representing archetypal guilds than the more conventional clustering method k-means that represent the cluster centroids with similar characteristic.

Tomb raider Julian + mahlmann

Drachen et al. [5] did a clustering analysis for two major commercial games applied to large-scale of high-dimensionality player behavior telemetry. K-means and Simplex Volume Maximization (SIVM) clustering were applied to the MMORPG *Tera* and the multi-player first-person shooter strategy game *Battlefield: Bad Company 2*. SIVM clustering is an adaption of Archetype Analysis (AA) for large-scale data sets to find extreme player behaviors profiles [3, 11]. The authors show the contribution differences from the two algorithms where k-means gives insights into the general distribution of behaviors vs. SIVM showing players with extreme behaviors. The selection of the most important features from the data set were followed by a method suggested by Drachen et al. [12], behavioral profiles were extracted and interpreted in terms of design language [9, 12].

In a recent study by Drachen et al. [13] the authors compare four different popular methods with purpose of clustering player behaviors and develop profiles from large-scale game metric data set from the highly popular commercial MMORPG World of Warcraft. The data set was collected from mining the Warcraft Realms site, recordings of on-line time and what level each player reached for each day in the years 2005-2010 for approx. 70 thousands of players. The authors selected playtime and leveling speed as their behavioral variables to show a measure of the overall player engagement in the

game, where playtime is one of the most important measure for calculating the churn rate (attrition) [7, 8]. Interpretable behaviors profiles were only generated by the k-means and the SIVM algorithm. The SIVM Archetype Analysis algorithm produces however significantly different behaviors that result in easier interpretation of behavior profiles compared to the k-means algorithm where the centroids are overall similar.

### 3.2 Clustering Large Data

K-means is one of the most studied clustering algorithm out there and is still actively researched. It's a simple algorithm that partition the data into  $k$  partitions by minimizing its objective function sum of squared error. From its appearance in a standard algorithm version by Stuart Lloyd in 1982 [14], it has been one of the most popular clustering algorithm to research because of its simplicity. There are many different research areas regarding k-means e.g. manually set the number of  $k$  partitions to cluster, initializing the centers of the partitions, dealing with outliers in data and scalability when dataset increases. In recent years k-means has also been very popular algorithm to study in the Map-Reduce framework where the algorithm can easily be applied to cluster large amount of data sets in parallel [15].

Guha et al. [16] designed an algorithm called STREAM that is based on the divide-and-conquer strategy and achieves a constant-factor approximation solving the k-median problem (a k-means variant). The algorithm divides the dataset into  $m$  pieces of similar sizes. Each of the pieces are independently clustered sequentially and all the centers from all the pieces are then clustered further. They show a new k-median algorithm called LSEARCH that is used by the stream algorithm and is based on local search algorithm solving the facility location problem [17] to solve the k-median problem. Results show that LSEARCH produced better quality clusters than k-means and the hierarchical algorithm BIRCH [18] but took longer to run. In 2009 Ailon et al. [19] extended the the work of Guha et al. introducing an one pass streaming algorithm for k-means with approximation guarantees. Achieving that they introduced a new algorithm called k-means# that builds on the non-streaming algorithm k-means++ by Arthur and Vassilvitskii [20] that is a combined algorithm of seeding the initial centers and running k-means. k-means# provides a bi-criterion  $(\alpha, \beta)$  approximation algorithm by choosing  $\alpha * k$  centers with approximation factor  $\beta$ . In the divide-and-conquer strategy they run

the k-means# independently on each piece of the data to achieve  $O(k * \log(k))$  random centers non-uniformly and use the k-means++ algorithm to find k centers from the intermediate centers from all the pieces of the data set.

Another approach is using a coresets by selecting a weighted subset from the original data set such that by running any k-means algorithm on the subset will give near similar results to running k-means on the original data set. Ackermann et al. [21] introduce a new algorithm called StreamKM++ that uses k-means++ algorithm from Arthur et al. [20] to solve k-means on the subset and they also design a new data structure called coresets tree to speed up the time for the sampling in the center initialization. StreamKM++ is a streaming version of k-means++ to cluster large data sets. Their approach was shown to be on par with LSEARCH algorithm in cluster quality but outperformed BIRCH by factor of 2. A recent work by Shindler et al. [22] proposed an algorithm called *Fast streaming k-means* based on the online facility location algorithm [23] and extends the work of Braverman et al. [24] proving a faster running time and a better approximation factor. Their algorithm outperformed both work of Ailon et al. [19] and Ackermann et al. [21] mentioned above.

Clustering data streams of an unknown length, evolving over time [25, 26] are challenging where it is not possible to access historic data points because of the amount of data arriving continuously. Aggarwal et al. [27] proposed a well-known stream clustering framework called CluStream for clustering large evolving data streams and is guided by application-centered requirements. CluStream has an online component that maintains snapshots of statistical information about micro-clusters (a.k.a. *cluster feature vector* [18]) in a pyramidal time window and an offline component that uses the compact intermediate summary statistics from the micro-clusters to find higher level  $k$  clusters using k-means, in a time horizon defined by an analyst. The authors proposed also a new high-dimensional, data stream clustering algorithm called HPStream that is highly scalable [28]. HPStream uses projected clustering [29], which can determine clusters for a subset of dimensions, to data streams and a new data structure called *fading cluster structure* that allows historical and current data to integrate nicely with a user-specified fading factor. Zhou et al. [30] presented a clustering algorithm called SWClustering that clusters evolving data streams over sliding windows to be able to analyze also the evolution of the individual clusters by eliminating influence by historic data points while



the new data points arrive. The Authors show that the CluStream algorithm is more sensitive to influences of outdated data and is less efficient.

Processing large of amount of data efficiently using parallel processing is an active research and gain much of popularity when the Google's MapReduce programming model was introduced by Dean and Ghemawat [15] that allows researchers easily to create highly scalable and fault tolerance algorithms by just implementing a Map and a Reduce function. Zhao et al. [31] implemented a parallel version of k-means (PKMeans) in MapReduce framework and show that their algorithm can effectively run on large data sets. They designed the Map function to calculate the closest cluster centroid for each point at a time and output a  $\langle key, value \rangle$  pair where key is the closest cluster centroid id and value is the data point. After each map task they apply a combiner function called Combine that is executed on the same machine as the Map function. The Combine function partly sum the values of the data points assigned to the same cluster and outputs a  $\langle key, value \rangle$  pair where key is the id of the cluster centroid and the value is comprised of the sum values and the number of data points in that cluster. Using a combiner reduces the amount of intermediate information that is processed and sent over the network to the Reduce task [15, 31]. Then the Reduce function sums up all the intermediate sub sum values from all the combiners and calculates the new centers for the next MapReduce iteration.

To overcome the sensibility of k-means to outliers in data, Li et al. [32] proposed an algorithm called MBK-means using the ensemble learning method called Bagging [33], generating k new data sets from the original data set with replacement sampling. For each of the new data set the k-means is run using the MapReduce framework until convergence then finally the k sets of k centroids are merged to form the final k centroids.

Many extensions on the traditional MapReduce framework have been proposed to support algorithms running iteratively [34–39] and incrementally [39–41] efficiently. The incremental MapReduce frameworks are interesting and relates to our work since we are incrementally clustering data but the use of these frameworks is not in this study.

### 3.2.1 Related work vs the study

Our work is most similar to PKMeans [31] described above, a parallel k-means implementation in MapReduce using a Combine function to reduce the intermediate data sent between the mappers and the reducers. We implement a parallel k-means algorithm in MapReduce so that each Map function efficiently calculates the distance to nearest cluster centers by calculating the distance matrix between all data points and the cluster centers instead of processing each data point separately. We show the algorithm in application where it incrementally clusters player behaviors where theoretically large data sets arrive daily. In our work we show k-means is a good approach to provide insights into the general distribution of behaviors in a game, work of Drachen et al. [5, 13] relates to ours when selecting and building features from user telemetry and extracting behavioral profiles by analyzing the centroids (basis vectors) in k-means. We will also experiment clustering a data stream that could theoretically change from day to day and discuss future work.

## Chapter 4

# Methodology

### 4.1 Game Events Data

#### 4.1.1 Building Features

### 4.2 K-Means in Map-Reduce

#### 4.2.1 Euclidean distance

#### 4.2.2 Normalized Histograms

### 4.3 Experimental Set-Up

## Chapter 5

# Results and Discussion

### 5.1 Results

### 5.2 Discussion

## Chapter 6

## Conclusions

## Appendix A

### Appendix Title Here

Write your Appendix content here.

# Bibliography

- [1] Tim Marsh, Shamus Smith, Kiyoungh Yang, and Cyrus Shahabi. Continuous and unobtrusive capture of User-Player behaviour and experience to assess and inform game design and development. In *1st World Conference for Fun 'n Games*, Preston, England, 2006.
- [2] Olana Missura and Thomas Gärtner. Player modeling for intelligent difficulty adjustment. In João Gama, Vítor Santos Costa, Alípio Mário Jorge, and Pavel B. Brazdil, editors, *Discovery Science*, volume 5808 of *Lecture Notes in Computer Science*, pages 197–211. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-04746-6. doi: [10.1007/978-3-642-04747-3\\_17](https://doi.org/10.1007/978-3-642-04747-3_17). URL [http://dx.doi.org/10.1007/978-3-642-04747-3\\_17](http://dx.doi.org/10.1007/978-3-642-04747-3_17).
- [3] C. Thureau, K. Kersting, and C. Bauckhage. Convex non-negative matrix factorization in the wild. In *Data Mining, 2009. ICDM '09. Ninth IEEE International Conference on*, pages 523–532, 2009. doi: [10.1109/ICDM.2009.55](https://doi.org/10.1109/ICDM.2009.55).
- [4] Anders Drachen and Alessandro Canossa. Evaluating motion: Spatial user behaviour in virtual environments. *International Journal of Arts and Technology*, 4(3):294–314, 2011. doi: [10.1504/IJART.2011.041483](https://doi.org/10.1504/IJART.2011.041483).
- [5] A. Drachen, R. Sifa, C. Bauckhage, and C. Thureau. Guns, swords and data: Clustering of player behavior in computer games in the wild. In *Computational Intelligence and Games (CIG), 2012 IEEE Conference on*, pages 163–170, 2012. doi: [10.1109/CIG.2012.6374152](https://doi.org/10.1109/CIG.2012.6374152).
- [6] Jun H. Kim, Daniel V. Gunn, Eric Schuh, Bruce Phillips, Randy J. Pagulayan, and Dennis Wixon. Tracking real-time user experience (true): a comprehensive instrumentation solution for complex systems. In *Proceedings of the SIGCHI Conference*

- on *Human Factors in Computing Systems*, CHI '08, pages 443–452, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-011-1. doi: [10.1145/1357054.1357126](https://doi.org/10.1145/1357054.1357126).
- [7] Tim Fields and Brandon Cotton. *Social Game Design: Monetization Methods and Mechanics*. CRC Press, 12 2011. ISBN 978-0240817668.
- [8] M Seif El-Nasr and Canossa A Drachen A. *Game Analytics: Maximizing the Value of Player Data*. Springer, 2013. ISBN 978-1-4471-4768-8.
- [9] G Zoeller. Game development telemetry. In *Proceedings of the Game Developers Conference*, 2010.
- [10] Geogios N. Yannakakis. Game ai revisited. In *Proceedings of the 9th conference on Computing Frontiers*, CF '12, pages 285–292, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1215-8. doi: [10.1145/2212908.2212954](https://doi.org/10.1145/2212908.2212954).
- [11] K. Kersting, M. Wahabzada, C. Thureau, and C. Bauckhage. Hierarchical convex nmf for clustering massive data. In Qiang Yang Masashi Sugiyama, editor, *Proceedings of the 2nd Asian Conference on Machine Learning (ACML-10)*, Tokyo, Japan, Nov 8–10 2010. URL <http://www-kd.iai.uni-bonn.de/pubattachments/477/kersting10acml.pdf>. draft.
- [12] A. Drachen, A. Canossa, and G.N. Yannakakis. Player modeling using self-organization in tomb raider: Underworld. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*, pages 1–8, 2009. doi: [10.1109/CIG.2009.5286500](https://doi.org/10.1109/CIG.2009.5286500).
- [13] A. Drachen, C. Thureau, R. Sifa, and C. Bauckhage. A comparison of methods for player clustering via behavioral telemetry. In *Foundations of Digital Games 2013*, 2013.
- [14] S. Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982. ISSN 0018-9448. doi: [10.1109/TIT.1982.1056489](https://doi.org/10.1109/TIT.1982.1056489).
- [15] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. In *Proceedings of the 6th conference on Symposium on Operating Systems Design & Implementation - Volume 6*, OSDI'04, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association. URL <http://dl.acm.org/citation.cfm?id=1251254.1251264>.



- [16] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O’Callaghan. Clustering data streams: Theory and practice. *Knowledge and Data Engineering, IEEE Transactions on*, 15(3):515–528, 2003. ISSN 1041-4347. doi: [10.1109/TKDE.2003.1198387](https://doi.org/10.1109/TKDE.2003.1198387).
- [17] M. Charikar and S. Guha. Improved combinatorial algorithms for the facility location and k-median problems. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 378–388, 1999. doi: [10.1109/SFFCS.1999.814609](https://doi.org/10.1109/SFFCS.1999.814609).
- [18] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. *SIGMOD Rec.*, 25(2):103–114, June 1996. ISSN 0163-5808. doi: [10.1145/235968.233324](https://doi.org/10.1145/235968.233324).
- [19] Nir Ailon, Ragesh Jaiswal, and Claire Monteleoni. Streaming k-means approximation. *Advances in Neural Information Processing Systems*, 22:10–18, 2009. URL <http://www1.cs.columbia.edu/~rjaiswal/ajmNIPS09.pdf>.
- [20] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA ’07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics. ISBN 978-0-898716-24-5. URL <http://dl.acm.org/citation.cfm?id=1283383.1283494>.
- [21] Marcel R. Ackermann, Christiane Lammersen, Marcus Märtens, Christoph Raupach, Christian Sohler, and Kamil Swierkot. Streamkm++: A clustering algorithms for data streams. In *ALENEX*, pages 173–187, 2010. URL [http://www.siam.org/proceedings/alnex/2010/alx10\\_016\\_ackermannm.pdf](http://www.siam.org/proceedings/alnex/2010/alx10_016_ackermannm.pdf).
- [22] Michael Shindler, Alex Wong, and Adam W. Meyerson. Fast and accurate k-means for large datasets. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2375–2383. NIPS, 2011. URL [http://books.nips.cc/papers/files/nips24/NIPS2011\\_1271.pdf](http://books.nips.cc/papers/files/nips24/NIPS2011_1271.pdf).
- [23] A. Meyerson. Online facility location. In *Proceedings of the 42nd IEEE symposium on Foundations of Computer Science*, FOCS ’01, pages 426–, Washington, DC, USA, 2001. IEEE Computer Society. ISBN 0-7695-1390-5. URL <http://dl.acm.org/citation.cfm?id=874063.875567>.

- [24] Vladimir Braverman, Adam Meyerson, Rafail Ostrovsky, Alan Roytman, Michael Shindler, and Brian Tagiku. Streaming k-means on well-clusterable data. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, pages 26–40. SIAM, 2011. URL <http://dl.acm.org/citation.cfm?id=2133036.2133039>.
- [25] Charu C. Aggarwal. An intuitive framework for understanding changes in evolving data streams. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 261–, 2002. doi: [10.1109/ICDE.2002.994715](https://doi.org/10.1109/ICDE.2002.994715).
- [26] Charu C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, SIGMOD '03, pages 575–586, New York, NY, USA, 2003. ACM. ISBN 1-58113-634-X. doi: [10.1145/872757.872826](https://doi.org/10.1145/872757.872826).
- [27] Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th international conference on Very large data bases - Volume 29*, VLDB '03, pages 81–92. VLDB Endowment, 2003. ISBN 0-12-722442-4. URL <http://dl.acm.org/citation.cfm?id=1315451.1315460>.
- [28] Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for projected clustering of high dimensional data streams. In *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30*, VLDB '04, pages 852–863. VLDB Endowment, 2004. ISBN 0-12-088469-0. URL <http://dl.acm.org/citation.cfm?id=1316689.1316763>.
- [29] Charu C. Aggarwal, Joel L. Wolf, Philip S. Yu, Cecilia Procopiuc, and Jong Soo Park. Fast algorithms for projected clustering. In *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, SIGMOD '99, pages 61–72, New York, NY, USA, 1999. ACM. ISBN 1-58113-084-8. doi: [10.1145/304182.304188](https://doi.org/10.1145/304182.304188).
- [30] Aoying Zhou, Feng Cao, Weining Qian, and Cheqing Jin. Tracking clusters in evolving data streams over sliding windows. *Knowl. Inf. Syst.*, 15(2):181–214, May 2008. ISSN 0219-1377. doi: [10.1007/s10115-007-0070-x](https://doi.org/10.1007/s10115-007-0070-x).

- [31] Weizhong Zhao, Huifang Ma, and Qing He. Parallel k-means clustering based on mapreduce. In *Proceedings of the 1st International Conference on Cloud Computing, CloudCom '09*, pages 674–679, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-10664-4. URL [http://dx.doi.org/10.1007/978-3-642-10665-1\\_71](http://dx.doi.org/10.1007/978-3-642-10665-1_71).
- [32] Hai-Guang Li, Gong-Qing Wu, Xue-Gang Hu, Jing Zhang, Lian Li, and Xindong Wu. K-means clustering with bagging and mapreduce. In *System Sciences (HICSS), 2011 44th Hawaii International Conference on*, pages 1–8, 2011. doi: [10.1109/HICSS.2011.265](https://doi.org/10.1109/HICSS.2011.265).
- [33] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. ISSN 0885-6125. doi: [10.1023/A:1018054314350](https://doi.org/10.1023/A:1018054314350).
- [34] Tyson Condie, Neil Conway, Peter Alvaro, Joseph M. Hellerstein, Khaled Elmeleegy, and Russell Sears. Mapreduce online. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation, NSDI'10*, pages 21–21, Berkeley, CA, USA, 2010. USENIX Association. URL <http://dl.acm.org/citation.cfm?id=1855711.1855732>.
- [35] Jaliya Ekanayake, Hui Li, Bingjing Zhang, Thilina Gunarathne, Seung-Hee Bae, Judy Qiu, and Geoffrey Fox. Twister: a runtime for iterative mapreduce. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, HPDC '10*, pages 810–818, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-942-8. URL <http://doi.acm.org/10.1145/1851476.1851593>.
- [36] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, HotCloud'10*, pages 10–10, Berkeley, CA, USA, 2010. USENIX Association. URL <http://dl.acm.org/citation.cfm?id=1863103.1863113>.
- [37] Yingyi Bu, Bill Howe, Magdalena Balazinska, and Michael D. Ernst. Haloop: efficient iterative data processing on large clusters. *Proc. VLDB Endow.*, 3(1-2): 285–296, September 2010. ISSN 2150-8097. URL <http://dl.acm.org/citation.cfm?id=1920841.1920881>.
- [38] Yingyi Bu, Bill Howe, Magdalena Balazinska, and Michael D. Ernst. The haloop approach to large-scale iterative data analysis. *The VLDB Journal*, 21

- (2):169–190, April 2012. ISSN 1066-8888. URL <http://dx.doi.org/10.1007/s00778-012-0269-7>.
- [39] Cairong Yan, Xin Yang, Ze Yu, Min Li, and Xiaolin Li. Incmr: Incremental data processing based on mapreduce. In *Proceedings of the 2012 IEEE Fifth International Conference on Cloud Computing, CLOUD '12*, pages 534–541, Washington, DC, USA, 2012. IEEE Computer Society. ISBN 978-0-7695-4755-8. URL <http://dx.doi.org/10.1109/CLOUD.2012.67>.
- [40] Pramod Bhatotia, Alexander Wieder, Rodrigo Rodrigues, Umut A. Acar, and Rafael Pasquin. Incoop: Mapreduce for incremental computations. In *Proceedings of the 2nd ACM Symposium on Cloud Computing, SOCC '11*, pages 7:1–7:14, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0976-9. URL <http://doi.acm.org/10.1145/2038916.2038923>.
- [41] Pramod Bhatotia, Marcel Dischinger, Rodrigo Rodrigues, and Umut A Acar. Slider: Incremental sliding-window computations for large-scale data analysis. *MPI-SWS-2012-004*, September 2012. URL <http://www.mpi-sws.org/tr/2012-004.pdf>.