

Department of Computer Science



Submitted in part fulfilment for the degree of MEng.

Swarm Memory

Harry Burge

Version 1.0, 2020-November

Supervisor: Simon O'Keefe

Acknowledgements

Contents

Executive Summary	vi
1 Introduction	1
2 Literature Review	3
2.1 Cloud/Backup storage policys/schemes	3
2.2 Swarm robotics	5
3 Motiviation	8
4 Methodology/Design	9
5 Conclusion	10
A Some apendix	11
B Another apendix	12

List of Figures

2.1 Example of a hetrogenus ant colony. <https://www.pinterest.co.uk/pin/77736358565153284>

List of Tables

Executive Summary

1 Introduction

Swarm robotics/intelligence/mechanics is becoming an increasingly important area of research for society as the world moves towards a distributed technology future. Swarm intelligence can be viewed as distributed problem solving [2, 5], this is ever becoming more and more relevant as computer systems start to level out in terms of individual performance [7] and parallelism is embraced to be able to satisfy the demand of the age of big data [9]. Swarm mechanics and robotics are on the rise in industry, as society's pace increases and manual labour is automated out, whether its drone delivery to impatient customers or mapping areas in dangerous zones [1].

Another area of swarm robotics research is distributed and local memory of swarm like agents. This area of research has gone down a route more to do with the optimisation of distributed problem solving algorithms rather than practical applications of storage of abstract ideas as a collective. Examples of this is from my understanding there is no research into collective memory on swarm like agents. This research is invaluable due to applications like mapping of dangerous area [2], being able to handle loss of agents and collect data on agents with limited memory. An explanation for this to be a less developed area of study is due to subjects like cloud based and raid based storage systems.

Storage of data on an ever changing network of storage devices is a hard task to complete, handling loss of connection between different servers, reliability to access of data and handling loss of services whether it be non-correlated or correlated failures [9]. This is very applicable to swarm memory handling of data however must be adapted, this is due to current algorithms such as raid not really being designed for highly dynamic systems such as a swarm system. However there are promising papers in this field that suggest approaches that can be adapted with few modifications to be applicable to a swarm based system [10].

The objectives of this report is to merge two different areas of study into one by using different knowledge of each area to create a suitable storage policy for swarm like agents to store collective memories of abstract ideas, then to perform analysis on a variety of simulations to explore the capabilities of said storage policy.

1 Introduction

// Talk about the sections of the report once completed.

2 Literature Review

This chapter will review two areas of relevancy to the project proposed, these are Cloud/Backup storage policys/schemes and Swarm robotics. Ideas and concepts from both areas will have to be relied upon for the completion/design of the storage policy.

2.1 Cloud/Backup storage policys/schemes

As most things in computer science, this area of study used to be very simple, with small datawarehouses and backups on to a medium like magnetic tape following something like a grandfather, father, son method. However as the years have progressed, technology has become much more complex requiring larger files of data to be stored and to be accessed frequently leading to the need of more complex backup systems to provide availability and longevity of data stored. A big component for the complexity of these algorithms other than providing a service better than competitors is the Legal Services Act. 2007 [11], which makes companies cloud storage solutions to have to be reliable and fast into collection of data for users.

Most algorithms used in production are called random replication policys [9] where data is partitioned and randomly distributed among other storage devices usually on different racks of the datacenter. This is an efficient design policy for handling non-correlated errors however lacks the robustness against correlated errors. These algorithms are good for longer term data storage with average popularity in terms of need for collection of that data.

Some of the problems arisen from random replication policys has spawned a new design policys, of which can handle correlated, non-correlated errors. Usually these policys also account for the demand of such item [9, 10]. To tackle these problems two approaches have been taken, one is to go from a higher level approach where you have a sort of manager which chooses which items to replicate and where based off demand, knowledge of other replications and outside factors [9, 12], not as suitable in terms of actual data changes but scheme changes of servers [13]. Working versions of these over a cloud service are tied together usually with a "Distributed

2 Literature Review

key-value store" [14] where you have these key-value pairs on multiple devices on a network and duplication only leads to more fault tolerance of the data stored.

Another way to handle this which doesn't rely on a more privileged node or duplicated data, and creates a distributed system is by having something like "SKUTE" as proposed in [10]. This is where all individual key-values have a their own choice on what to do in a distributed system. This is described as four sections which are Migration, Suicide, Replication and Nothing; where Migration is the moving of data to lower costing and more redundant servers, Suicide is the removal of itself usually based on amount of duplicates and uses something like Paxos [15] to decide if it should suicide and Replication where it decides that it is being used enough to need to be replicated or there is possibly no other duplicates of this data.

An algorithm like "SKUTE" [10, 16] will be suited best for swarm like agents due to the distributed nature of a swarm, and not wanting to have a static/temporary leader due to issues like communication bottleneck, power loss in the swarm near the leader due to flow of information to the leader, loss of leader (mainly only if static) [1, 5]. Also some of the main reasons for using swarm robotics is the inherent parallel nature that they bring to the table, therefore creating leader based algorithms are not in the spirit of the design of a homogenous swarm, unlike a heterogeneous swarm.

Another area of study which is highly static is storage of data on local disks and how to keep either backups for disk failure or/and improve write performance onto disks, rather than just duplicating data like as described above. An example scheme for this is something like RAID, which have different levels based off the type of attributes that you may need [?]. In terms of cloud based storage RAID arrays are used commonly internally rather than externally to a different NAS or storage server. This is usually because we will have the guarantees of RAID for disk failures if they acquire and for example a server goes down we have the above replication schemes to be able to handle that. Therefore leading to having RAID arrays across multiple nodes to be sort of redundant. However with using a parity [8] based higher level scheme you will get space savings on the duplication of data, this is harder to implement though and most likely not needed due to servers that are lost due to power outages usually come back on pretty soon also known as something like a concurrent-failure.

// TODO: Say what a correlated and non correlated failure is

2.2 Swarm robotics

Within research about swarms there is a split between practical solutions of the agents and use of agents as an algorithm, this split can be seen as swarm robotics and swarm intelligence. Swarm intelligence is where we use swarm like behaviour to a problem, for example traveling salesman problem [5], this means that we use agent like code to compute a task. These tasks have usually been solved using a different algorithm for example for the TSP using a genetic algorithm to solve, and the swarm algorithms like AS-TSP [5] are alternatives to that algorithm. These provide benefits and drawbacks to their counterparts, an area which these algorithms could excel and are researched into is the networking space due to the natural parallelism that can be exploited.

However this is not the route of research that will be needed for this project, rather we will be looking at swarm robotics. Swarm robotics has the same idea as swarm intelligence however it focuses on tasks of which are usually designed to have an agent/agents complete, this is mainly designed for the practical space like moving objects or mapping an area, whether simulated or not. These swarms come in three types: heterogenous, homogenous and a subcategory of homogenous being hybrid swarms [1]. These three types can be mapped onto both the controls of the swarm and the agents body/abilities.

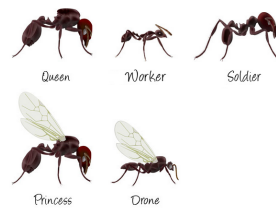


Figure 2.1: Example of a heterogeneous ant colony. <https://www.pinterest.co.uk/pin/777363585651532845/>

A heterogeneous swarm is where there are differences between the agents as in Figure 2.1, this is most commonly occurring in nature and not usually studied into due to the differences in the agents, being a rarely needed property in research based problems. In real world solutions heterogeneous solutions can be of great use for example as described in [1] with a mother ship being a navy boat and a swarm of quadcopters. The reason for less research into this area is due to some key drawbacks of having a heterogeneous swarm. Usually you will have a hive-mind like system if you have a heterogeneous swarm where you have leaders giving commands to subordinates or even one leader commanding the entire swarm. This is less desirable due to if there is a loss of those leaders you lose the ability to control the swarm,

2 Literature Review

in are example this doesn't matter so much due to if you have loss of the mothership something has gone significantly wrong already. Due to the differences in the swarm agents it allows for greater efficiency of the swarm for example having a robot that can mine and one that can farm, however the major loss of one type can lead to the loss of the colony. With swarms like these designs need to be taken so that agents can interchange between the tasks or that if agents fail there is no impact on the colony. Ants usually fit into this type of swarm where you have a queen, worker and major ants, however some ants like Leaf-Cutter Ants also have subcategory of workers like a fungus farmer. If there is a significant loss of workers, majors start doing tasks that normally workers would do [5], and with the farmer subcategory of the workers if a significant loss of them happens other workers can take over that job and learn how to do it, however that leads us more towards homogenous agents within a heterogenous swarm.

A homogenous swarm is where you have each agent being the same, this is less often in nature and is more towards man-made agents, this is due to nature taking to the more efficient approach and due to having learning in the agents can adapt between roles [1, 5]. In homogenous swarms we get significant redundancy due to if an agent goes down we have a swarm worth of replacements for that agent. With this redundancy we gain possible losses of efficiency due to either agents being too simple therefore losing specialism or each agent has parts for specialism but may never need a part. An example of this is a robot that has hands can farm and dig, however if we want them to be more efficient we would have to give them both a hoe and pickaxe if we were using a homogenous model and if they couldn't switch between hoe and pickaxe during runtime of the swarm usage. In a homogenous style of swarm usually we will have homogenous control, this is where all agents decide what they want to do based on what other agents are doing and internal parameters, this can be equivalent to something like an emergent swarm. This follows a distributed problem solving/communication design compared to leader based design like hive mind or structured/hierarchical controlled swarms.

With swarm robotics everything gets a bit messy, usually there is no clear cut name or design that can be assigned to swarm models and behaviours. This is where hybrid approaches come into play. A hybrid approach is mixtures between both heterogenous and homogenous natures in both communications and agent design. In terms of communication usually when taking a hybrid approach to a swarm usually you will have a swarm leader or leaders designated by a swarm of robots, this is handled with a consensus algorithm like paxos [?]. Also in hybrid models if we want to gain the efficiency of heterogenous models and the adaptability/reliability of homogenous swarms we can use something like tools. Humans themselves are a great example of a hybrid based swarm. Though humans have variations in characteristics they can be seen as pretty homogenous in terms of the tasks

2 Literature Review

that they perform, obviously removing edge case actions that humans do. Tools and knowledge can be spread between humans to make the swarm more efficient and an agent can specialize in a certain area however if some agents are lost other agents/humans can replace them by using the same tools and learning from the remaining agents of that task. Also the natural power based structure of humans fits a hybrid model in terms of leadership of some kind, however the leaders aren't needed for every single action so fits into a usually hierarchical power structure, compared to something of a hive mind model for swarm like agents.

3 Motivation

The reason why I want to undertake this project of merging two/three research areas into one is because I believe that swarm robotics hasn't had much research into areas like this for storage of data on the swarm. The application of such a technology and algorithm is widely applicable to robots and cloud storage on a volatile network. There are lots of applications for this technology ranging from surveillance, cooperative task completion and the future of human space exploration. The push for an optimal distributed storage system seems to me to be a key for the future of mankind, being able to parallelise programs efficiently, this is especially true in quantum computers being able to handle parallelised algorithms efficiently, obviously this is very simplified.

Examples of such applications that the merger of these two fields bring together are mainly either just new algorithms for already handled cloud storage which is super needed but could be made more efficient with algorithms that are distributed, same sort of idea as block chain however that focuses more on security. Useful applications of swarms fit into usually the military, surveillance, delivery or exploration space. If talking future, future technologies you could be talking medical, terraforming however this is very vobatum and speculative.

4 Methodology/Design

5 Conclusion

A Some apendix

B Another apendix

Bibliography

- [1] J. C. Barca and Y. A. Sekercioglu, "Swarm robotics reviewed," *Robotica*, vol. 31, no. 3, pp. 345–359, 2013.
- [2] V. Kumar and F. Sahin, "Cognitive maps in swarm robots for the mine detection application," *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483)*, Washington, DC, 2003, pp. 3364-3369 vol.4, doi: 10.1109/ICSMC.2003.1244409.
- [3] H. Wang, D. Wang and S. Yang, "Triggered Memory-Based Swarm Optimization in Dynamic Environments," in *Applications of Evolutionary Computing*, M. Giacobini, Ed. Berlin, Germany: Springer-Verlag Berlin and Heidelberg GmbH & Co. K, 2007, pp. 637–646.
- [4] D. A. Lima and G. M. B. Oliveira, "A probabilistic cellular automata ant memory model for a swarm of foraging robots," 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), Phuket, 2016, pp. 1-6, doi: 10.1109/ICARCV.2016.7838615.
- [5] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. Cary, NC, USA: Oxford University Press, 1999.
- [6] L. Xiang, Y. Xu, J. Lui, Q. Chang, Y. Pan, and R. Li, 'A Hybrid Approach to Failed Disk Recovery Using RAID-6 Codes: Algorithms and Performance Evaluation', *Association for Computing Machinery*, vol. 7, p. 11, 2011
- [7] C. Mims, 'Why CPUs Aren't Getting Any Faster', *MIT Technology Review*, 2010. [Online]. Available: <https://www.technologyreview.com/2010/10/12/199966/why-cpus-arent-getting-any-faster/>. [Accessed: 01-Dec-2020].
- [8] U. Troppens, W. Müller-Friedt, R. Wolafka, R. Erkens, and N. Haustein, 'Appendix A: Proof of Calculation of the Parity Block of RAID 4 and 5', in *Storage Networks Explained: Basics and Applic-*

Bibliography

- ation of Fibre Channel SAN, NAS, ISCSI, InfiniBand and FCoE, U. Troppens, Ed. Chichester: Wiley United Kingdom, 2009, pp. 535–536.
- [9] J. Liu and H. Shen, "A Low-Cost Multi-failure Resilient Replication Scheme for High Data Availability in Cloud Storage," 2016 IEEE 23rd International Conference on High Performance Computing (HiPC), Hyderabad, 2016, pp. 242-251, doi: 10.1109/HiPC.2016.036.
- [10] N. Bonvin, T. G. Papaioannou, and K. Aberer, A Self-Organized, Fault-Tolerant and Scalable Replication Scheme for Cloud Storage. New York, NY, USA: Association of Computing Machinery, 2010.
- [11] Legal Services Act. 2007.
- [12] A. Prahlad, M. S. Muller, R. Kottomtharayil, S. Kavuri, P. Gokhale, and M. Vijayan, 'Cloud gateway system for managing data storage to cloud storage sites', 20100333116A1, 2010.
- [13] B. Czejdo, K. Messa, T. Morzy, M. Morzy, and J. Czejdo, 'Data Warehouses with Dynamically Changing Schemas and Data Sources', in Proceedings of the 3rd International Economic Congress, Opportunities of Change, Sopot, Poland, 2003, p. 10.
- [14] 'Key-Value Scores Explained', HazelCast. [Online]. Available: <https://hazelcast.com/glossary/key-value-store/>. [Accessed: 02-Dec-2020].
- [15] L. Lamport, 'The Part-Time Parliament', in Concurrency: The Works of Leslie Lamport, New York, NY, USA: Association of Computing Machinery, 2019, pp. 277–317.
- [16] D. Agrawal and A. E. Abbadi. The tree quorum protocol: An efficient approach for managing replicated data. In VLDB'90: Proc. of the 16th International Conference on Very Large Data Bases, pages 243–254, Brisbane, Queensland, Australia, 1990.
- [17] S. Lynn, 'RAID Levels Explained', PC Mag, 2014. [Online]. Available: <https://uk.pcmag.com/storage/7917/raid-levels-explained>. [Accessed: 06-Dec-2020].
- [18] J. Hu et al., Eds., HiveMind: A Scalable and Serverless Coordination Control Platform for UAV Swarms. ArXiv, 2020.
- [19] D. Calvaresi, A. Dubovitskaya, J. P. Calbimonte, K. Taveter, and M. Schumacher, Multi-Agent Systems and Blockchain: Results from a Systematic Literature Review. Cham, Switzerland: Springer Interna-

Bibliography

tional Publishing, 2018.