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

List of Figures

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 technollogy future. Swarm intelligence can be viewed as distributed problem solving[2, 5], this is ever becoming more and more relevent as computer systems start to level out in terms of indurvidual performance [7] and parrelism is inbraced 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 soceitys pace increases and manual labour is automated out, whether its drone delivery to inpatient 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 diffrent 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 algorthims such as raid not really being designed for highly dynamic systems such as s swarm system. However there are promising papers in this field that suggest approahes 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 diffrent areas of study into one by using diffrent knowledge of each area to create a suitable storage policy for swarm like agents to store collective memorys of abstarct ideas, then to perform analysis on a variety of simulations to explore the capabilitys of said storage policy.

// 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 magentic 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 avaliabilty and longevity of data stored. A big component for the complexity of these algortihms other than providing a service better than compettitors is the Legal Services Act. 2007 [11], which makes companies cloud storage solotions 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 radomly distributed amoung other storage devices usually on diffrent racks of the datacenter. This is an effecicent design policy for handiling non-correlated errors however lacks the robstness 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 approches have been taken, one is to go from a higher level approcach 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 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 priverliged node or duplicated data, and creates a distributed system is by having something like "SKUTE" as proposed in [10]. This is where all indurvidual key-values have a their own choose on what to do in a distributed system. This is descirbed 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 usally based on amount of duplicates and uses soemthing 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 their is possibly no other duplicates of this data.

An algrithm like "SKUTE" [10] will be suited best for swarm like agents due to the distributed nature of a swarm, and not wanting to have a static/tempory leader due to issues like communication bottleneck, power loss in the swarm near the leader due to flow of infomation to the leader, loss of leader (mainly only if static) [1, 5]. Also some of the main resons for using swarm robotics is the inherint parrell nature that they bring to the table, therefore creating leader based algorithms are not in the spirit of the design of a homogenus swarm, unlike a hetrogenius swarm.

Another area of study which is highly statc 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 diffrent levels based off the type of attributes that you may need [**?**]. In terms of cloud based storage RAID arrays are used commonaly internally rather than externally to a diffrent NAS or storage server. This is uaully because we will have the gurantees 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 usally 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 soloutions of the agents and ue of agents as an algorithm, this split can be seen as swarm robotics and swarm intelligence. Swarm intelligence is where we use swarm like behavoiur to a problem, for example traveling salesman problem [5], this means that we use agent like code to compute a task. These tasks have usally been solved using a diffrent 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 there counterparts, an area which these algorithms could excell and are researched into is the networking space due to the natrual parrellelism that can be expoited.

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 usally desiged 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. This swarms come in three types: hetrogenus, homogenues and a subcatergoriy of homogenus being hybrid swarms [1]. These three types can be mapped onto both the controls of the swarm and the agents body/abilitys.

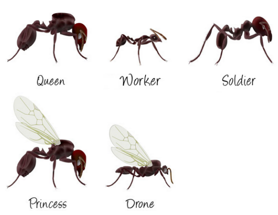

Figure 2.1: Example of a hetrogenus ant colony. https://www.pinterest.co.uk/pin/777363585651532845/

A hetrogenus swarm is where there are diffrences between the agents as in Figure 2.1, this is most commonly aquiring in nature and not usally studied into due to the diffrences in the agents, being a rarely needed property in research based problems. In real world soloutions hetrogenus soloutions can be of great use for example as desribed in [1] with a mother ship being a navy boat and a swarm of quadracopters. The reason for less research into this area id due to some key drawbacks of havin a hetrogenus swarm. Usaully you will have a hivemind like system if you have a hetrogenus 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, 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 diffrences in the swarm agents it allows for greater efficency 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 usally fit into this type of swarm where you have a queen, worker and major ants, however some ants like Leaf-Cutter Ants also have subcatgorey of workers like a fungus farmer. If there is a significant lose 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 homgenus agents within a hetrogenus swarm.

A homogenus 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 efficent approach and due to having learning in the agents can adapt between roles. In homogenus swarms we get sugnificant redundancy due to if an agent goes down we have a swarms worth of replacements for that agent. With this redundancy we gain possible losses of effiecney due to either agents being to simple therefore losing specialism or each agent has parts for speclim 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 effiecent we would have to give them both a hoe and pickaxe if we were using a homogenus model and if they couldn't switch between hoe and pickaxe during runtime of the swarm usage. In a homogenus style of swarm usally we will have homogenus control, this is where all agents decide what they want to do based of what other agents are doing and internal parameters, this can be equivilent to something like an emergernt swarm. This follows a distributed problem solving/commmunication design compared to leader based design like hivemind or structered/heirachical controlled swarms.

With swarm robotics everything gets a bit messy, usally there is no clear cut name or design that can be assigned to swarm models and behavouirs. This is where hybrid approaches come into play. An hybrid approach is mixtures between both hetrogenus and homogenus natures in both communications and agent design.

//Todo: talk about humans be hybrid due to power changes and using tools to be able to speicalize

3 Motiviation

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-

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, Opportunieties 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].