Planning and Research Report

Paul-Michael Sorhaindo

October 2011

Abstract

Abstract goes here..

Chapter 1

Introduction

1.1 About this paper

In this paper an outline for the project will be put forward, research will be carried out and reported on, a plan will be proposed as to how to carry out the rest of the project with small targets and goals and finally areas of further research will be identified. The paper will be divided into four significant chapters the introduction, in which background will be provided as to how the topic for the project was chosen, relevance to the course and the project's objectives and aims. This will be followed by a research chapter which begins by outlining research aims and continues to report on my initial research into some of these aims. The chapter following this will consist of aims and goals in a chronological order that will be viewed as form of schedule for the project. Finally the report will conclude with a discussion about the areas that are as of yet not fully researched but will be read into further as the project progresses.

1.2 Producing a Topic

The process of settling on a topic for a project took quite some time the initial idea was to produce something which was in the Natural Language Processing field. The primary aim was to implement an intelligent agent that would parse text, recognizing relationships between pronouns and the nouns they refer to. Then storing everything in a database that it parsed so that it could analyze the data statistically and self optimize the parsing rules I initially gave it in order to parse more efficiently and accurately. Furthermore I wanted to extend that idea further and include text generation functionality to the proposed project and allow it to communicate with a user via a keyboard and screen so it would effectively become a chat-bot which "learnt" from its mistakes. After further research into the NLP field I realized how ambitious these plans were and that just the subject of data mining a corpus of descriptive text in isolation is a broad field. I also interested exploring the field of AI in particular genetic

algorithms. The main concept of the idea was to recreate a race track and racing car and implement an intelligent agent with a limited view of the track such as a human would have and after exposing functionality of the car such as a human would have give the agent the goal of completing the track in the least amount of time possible. It turned out I didn't go with this idea because it would have been an incredible amount of work to develop the track and car and to accurately simulate all the physics involved that it was possible that the AI element wouldn't get time required to produce a finished artifact.

After reading around the field of AI I came across the topic of swarm intelligence, which was first used in reference to robots. The definition given by Beni, G. and Wang, J. (1989) (!cite) of Swarm Intelligence is "systems of non-intelligent robots exhibiting collectively intelligent behaviour evident in the ability to unpredictably produce 'specific' ([i.e.] not in a statistical sense) ordered patterns of matter in the external environment" the idea of implementing a system that could simulate a group of simple objects that together worked intelligently appealed to me. I looked to the real world for something to use as a model, so I could simulate on a computer with AI, and I realized that ants and an ant colony as a whole would offer a great model to simulate after analyzing the problem and how it would be implemented (with my supervisor), I realized that again the problem consisted of many issues the first of which was ant colonies often consist of tens of thousands, if not more, ants. Simulating that many objects which are all moving independently each with their own location and each with their own task, has the potential to be very processor intensive. As the number of ants in the simulation increase, depending on the implementation used, the simulation model could easily slow significantly and stop, producing incorrect values no longer in real time resulting in something with high latency and therefore inaccurate. In the context of a casual simulation the accuracy and need for it to be real time aren't imperative but, in the computer games industry this is increasingly becoming not the case. The games produced by the big development houses which really turnover money, in terms of the amount spent developing them and the amount they sell for, come under more and more scrutiny from the average game player as they expect their gaming experience to become more immersive and realistic. Although the realtime in games, often classified as soft realtime because the usefulness of information recieved after it is due degrades the longer the delay is, the market has become such that game development houses who allow their products too push the leaniancy of the soft realtime requirements will suffer at the hand of bad reviews. As the major companies in the industry who produce AAA titles sell alot of releases based of the reputation their previous products have a good reputation, as in most industries, is vital.

1.3 Relevance to degree

Making games has been described by many as an art, (Wired reference) and just as in the art of story telling it is important for characters to have life and feel real so too it's important for many games because of the aesthetic path they've chosen to present every object and every movement within as realistically as the end users machine will allow in order to fully immerse the player. Ofcourse there are many tricks a programmer can employ to fake the feeling of real physics, true artificial intelligence and produce graphics that stun the gamer. But these tricks often get swapped out for the real deal when technology catches up with designers imaginations. Way back games were displayed on monochrome screens and made use of coloured film over the display to create the illusion that different colours were on the screen, while today most people think nothing of their 1080p widescreen displays which absorb our natural field of view with high definition. There is always a demand for accuracy within computer games and research tries to meet these demands. Take for example one of the most accurate forms of rendering in computer graphics, ray tracing, in August of 2009 Nvidia launched what they dubbed 'The World's First Interactive Ray Tracing Engine' the engine OptiX (¡?better source) which is just the result of one of many projects which will ultimately allow developers to bring games and many other application of computers to another level of accuracy and realism.

1.4 Overall Project Aims and Objectives

The fundemental aim of this project is to produce a simulation of an ant colony, exploiting the use of parallel programming, which reperents ants as individual units working independently without an overarching, all-knowing method which controls each one. However this aim can be split up into several smaller aims. The aims are detailed below because it is important to highlight different areas under which the project can be assessed. I have split the aims up in under the following headings, Technical, Qualititive and Learning outcomes. It is important that any project has each of these as they are all measured in different ways. Technical aims are absolute while Qualititve aims are subjective. Finally whether or the learning aims have been achieved or not can only be decided by the person conducting the project.

1.4.1 Technical Aims

- To explore paralellism and implement parallel and sequential algorithms.
- To produce functional code that addresses the problem.
- To implement accurate collision detection.
- To implement a data structure that holds the information in the simulation.

- To develop custom algorithms suitable for this project.

1.4.2 Qualititive Aims

- To produce ants which have a sense of position and can navigate their environment
- To produce ants which appear to interact through the use of phermones
- To produce an environment that provides obstacles that tests the ants behaviour
- To produce clear concise and understandable code.

1.4.3 Learning Outcomes

- To be able to program functionally
- To have a better understanding of parallelism
- To have a good understanding of the programming language(s) that I use to implement my solution.
- To be able to analyze and critque algorithms effectively.

1.5 Project overview

In this project the problem of simulating the independant movements of a group of ants is tackled, instead of taking a short-cut approach to the problem and allowing every ant to see the whole 'world' the ants will have a realistic limited field of sense each going about their own tasks independantly but while sharing the common goals of the colony. This will provide several problems which the project will attempt to address over the course of the project there will be a particular focus on the amount ofants working individually yet simultaneously, while trying to keep the nature of their interaction as accurate as possible. However it is important to mention here that when the terms realistic and accurate are used they will be used with the meaning "accurately conforming to a list of given assumptions", as detailed below, because the finer details of how an ant or ant colony functions is not the point of this project. However some brief reading of secondary sources will be done in order to produce a list of assumptions which will define the realistic (to a given definition of realistic) behaviour of an ant and ant colony which hopefully should be agreeable for the average non-entomologist. After this we will start analyzing and discussing various approaches to tackling the problem that has been posed, firstly abstractly then programatically looking at various programming paradigms. Then research will continue further on methods mentioned in this disscussion that seem most likely to provide a solution and various ways of implementing the solution will be presented using the techniques discovered during this initial research period.

1.6 Ant Colony Generalizations

Listed here are the aforementioned generalizations or preconceptions about ants and their colonies, that will be assumed as factual for the purposes of this project. All ants need both food and water to survive, according to the Center for Insect Science Education Outreach at the University of Arizona ants can go for quite some time without food but without water they will be dead within a day. For this reason water will be a stronger priority for all ants over food and ants within the simulation will die without water or food, I will determine the amount of time a satiated ant can survive for as three simulation days without food and six simulation hours without water. After reading an article on the BBC News website about sleeping patterns in ants I learnt that that queen ants live for years where as worker ants live for months because of this statement I'm giving queen ants within my simulation a lifespan of three to four simulation years and likewise worker ants will have a life span of three to four simulation months. The same article went on to talk about how ants sleep in short power naps lasting just over a minute two hundred and fifty times a day and the queen sleeps for longer periods of around six minutes nintey times a day, I also will reflect this in the context of my simulation. The concept of insects having different roles or jobs within a hive is explored in the work of Oster and Wilson (1978) (!cite) although I won't get into too much detail on this point I will outline different roles that ants within the simulation can take. Firstly there is a queen, in my simulation this will be dramatically simplified to one per colony, her job will to simply lay eggs to produce more workers.

In the simulation the queen will be allowed to produce eggs at varying frequency depending on how much food and water is available to her and she will only produce offsring while there is room in the nest. Her main priority will be ensuring the colony doesn't run out of workers. Rhe 'worker' class of ants will then be subdivided into four seperate castes. Firstly a builder caste of ant will be created who's main priority is to build, maintain and repair the colony's nest. A builder ant's work will never be done as even when the nest is in excellent condition the ants will always strive to make the nest bigger so the queen can continue producing ants and for increased space for food storage. Builder caste ants will also be responsible for cleaning the nest. The second class of ants to exist within the simulation will be be the gatherers this castes' number one priority is to go out to known sources of food and water and carry it back to the colony's nest for storage. Slightly different from this class is the scout whose main purpose is to explore new territory and communicate to other ants where new sources of food and water are aswell as building materials useful for the builder caste of ant. Finally onr last caste will be outlined as the soldier ants whose main job is to protect the the nest and provide security, they will attempt to achieve this by patrolling the perimiter of the nest, the perimiter of the ant colony's territory as defined by the scout caste, and other important areas to the colony such as the food store and the queen.

11	12	13
21	22	23
31	32	33

Table 1.1: Table of Foos

Within my artifact I'll have to represent each of these castes by varying values of attributes assigned to each ant. I will go into further detail about the attributes I will use to represent ants within my system in further detail later. But I will also have to represent communication between the ants, to keep communication between ants to some level of realism I will allow each ant within the simulation to produce pheremones. Pheremones will vary in strength All ants will have the capability to release a pheremone indicating danger. When ants die they will release a very strong pheremone signaling general danger, but scout ants will be able to release strong pheremones indicating whether this is a danger all ants should flee from or if soldier ants should move in and either protect the nest. Situations will often occur where scout ants will call in the aid of soldier ants to attack a larger insect and once this attack is successful the target is now a food source. Scout ants will notify gatherer ants of this by again releasing a pheremone which indicates a food source has been discovered while returning to the nest. I will also allow builder ants to communicate with a similar system detecting scout pheremones indicating building resources or an area of damaged nest and react accordingly. Scout ants will be able to release most pheremones that all castes of ant are capable of producing while other castes range of pheremones will be more specialized. Details of of the planned pheremone structure will be shown in the table below (Table fig. 1.1)

1.7 Problem Area

When the concept of the simulation is broken down it is immediately evident that a crutial element of the system to be produced is the ant. As mentioned before in a real world colony amount of ants in a colony can be very large, within my simulation however in order to produce a working final product an incremental approach will be taken to solve the problem loooking at first representing one ant then two ants and look at getting collision detection working between them before increasing the complexity of an ant and increasing the amount of ants in the system. Even though starting small will allow me to take a numerous amount of approaches to the problem The project will be aiming to take an approach that scales well, so that when we add more ants to the simulation the performance doesn't drop to a point where it is no longer accurate and realistic to a given definition of realism. This is important to note as when we look for a programming approach in the research stage of the paper it would be sensible choose one that inherently produces solutions capable of scaling well so that performance doesn't dip as the demand on the system is increased. With this in

mind a notable observation that can be made of the problem is that it is rich in concurrency. Concurrency is a property attributted to a system which performs more than one "possibly unrelated tasks at the same time." [4] When holding this definition against my problem it is clear that my problem is a concurrent one.

Chapter 2

Research

2.1 Research Aims

Throughout the following section a breakdown of the topics that will be important to my project will be provided. Questions will then be derived from these topics in order to produce a set of aims that the research part of the paper will aim to address. As research will be an on going area of focus in the project not all the aims will be covered in this paper. When tackling a concurrent problem it would profitable to research what are the most popular and promising programming paradigms used to approach concurrent problems, taking this further to look at why these paradigms are more efficient at dealing with my problem. In the process of getting more specific, the focus will then be turned to what programming languages are used for this type of problem, again looking at why certain languages are perferred or avoided. Other important aspects of the project will also be further researched such as collision detection, the visual representation of the simulation and artificial intelligence concepts such as genetic algorithms and neural networks. However not all of the research will be included witin this paper as this will be an on going process. Most importantly a programming paradigm will be chosen to focus on "the choice of programming paradigm can significantly influence the way one thinks about problems and expresses solutions" to problems. [5]

To summarise the research aims of this project are as follows:

- Identify a programming paradigm that is suitable for the problem.
- Compare languages that could be used in the implementation of the project.
- Review algorithms and Data Structures (represented in languages which may be used throughout this project) that tackle concurrency and other problems found in my project.
- Analyze other large scale simulations.

- Compare approaches to large scale collision detection.
- Look at possible ways to represent the information in the simulation visually.

2.2 Programming Paradigms

"Over the last decades, several programming paradigms emerged and profiled. The most important ones are: imperative, object-oriented, functional, and logic paradigm."[8] The next few paragraphs will look breifly at these four paradigms and go on to mention a few others which might be of interest considering the problem.

2.2.1 The Imperative Paradigm

The imperative programming paradigm is based on the Von Neumann architecture of computers, introduced in 1940s. [8] This means the programming paradigm similar to the Von Neumann machine operate by performing one operation at a time, on a certain pieces of data retrieved from memory, in sequential order. According to Backus [2] the man who cointed the term "The von Neumann bottleneck", "there are several problems created by the word-at-a-time von Neumann style of programming, with its primitive use of loops, subscripts, and branching flow of control."

2.2.2 The Object Orientated Paradigm

The Object orientated paradigm is as its name suggests center around objects. Its aim is to group similar data and functionality related to that data into objects. Blocking access to data within objects this way is commonly called encapsulation and is common in the Object orientated paradigm. One advantages of programming in the Object Orientated paradigm

2.2.3 The Functional Paradigm

Object orientated programming has been described as "the antithesis of functiontional programming" [7] From the two programming paradigms above it can be deduced that a good programming language is modular. But John Hughes claims that it is not enough for a language to just support modularity, it needs to go a step further and make modular programming easy. To do this a programming language needs to provide flexible functionality in order to bring modules together. In an an article on Functional Programming Hughes writes "Functional programming languages provide two new kinds of glue - higher-order functions and lazy evaluation."[3]

2.2.4 The Logic Paradigm

The logic programming paradigm - talk about inference and goals and pedicates. refer Prolog.

2.3 Why Parallelism

Over the last few years hardware manufactures have shifted their focus from producing processors with faster clockspeeds. Traditionally a programmer would be able to produce a program which ran on the current processor, and to get it to run faster he would just wait for faster hardware to be released and run their code on that.

Parallelism is also viewed as a programming paradigm, it takes the concept that large problems can usually be broken down into smaller simpler problems. If these problems can be solved independant of one another then different processors can compute these problems at the same time. Parallelism can take place at different levels, there are levels of parallelism pertaining to hardware while others concepts in parallelism deal with the code. The two levels of parallelism that apply to code are the ones of particular concern here and are referred to Data and Task parallelism. Parallelism can also be see at the Instructional level. For instructions to be executed in parallel they need to be data independant. Take fo Another area where parallelism can be identified is at the Bit level although this is more relevant to hardware development. There are also two approaches to programming in parallel implicit parallelism is all down to the language and the compiler. Programming in languages such as pH and NESL don't need specific functions or takes Explicit parallelism,Concurrency) Explain differences

This project will focus on parallelism through functional langauges. It will persue this path because due to functional programming language features such as lazy evaluation it easier to write code that allows more computations to be evaluated seperatly on either different threads, different cores and potentially different machines. In functional languages it is easier (...? Reuesable code, functional code scales better...?) However similar algorithms maybe generated in imperative, object orientated languages to show differences in behaviour and performance.

2.4 Choice and Use of Programming Language

When looking for a language to develop this project in it hard not to consider Haskell. Haskell

2.5 Brief History of Parallelism

People as early as, Flynn were thinking about the power that could be harnessed from being able to compute things in parallel. Flynn's Taxonomy is a classification of the different methohd of processing data. Flynn defined four methods these were, SISD- Single Instruction Single Data, SIMD- Single Instruction Multi data, MISD- Multi Instruction Multi Data and MIMD- Multi Instruction Multi Data. Even today it is still good approach to look at software an Models Kuck and its

2.6 What problems can be parallelized?

In theory any problem which has the ability to perform more than one calculation at the same time has the potential to be parallelized. However some calculations in an algorithm may have Data dependencies, Dave Patterson. Computer Architecture.

2.6.1 The Limits of Parralelization

It has been proved by Gene Amdahl that increasing the amount of processors working on a program only increases speed for so long. Amdahl's law shows that the speedup of a program is limited by the sequential portion of the program. This is clear because sequential processes need to be performed in sequence so by definition they are prevented from benifiting from multiple processors. In 1967 at the AFIPS Spring joint Computer Conference, while talking about the overhead of "data management housekeeping" which can be found in any computer program, Amdahl reportedly said that "The nature of this overhead appears to be sequential so that it is unlikely to be amenable to parallel processing techniques. Overhead alone would then place an upper limit on throughput."[1] This upper limit has been deduced from his conference talk and given rise to the widely known formula below. The formula calculates the potential speedup that could be brought to a program by adding more processors work on the parallel portion of a problem. The sequential overhead generated in arranging the parallel computation that Amdahl mentions is represented by (1-P) and therefore P representing the parallel section of the program makes up the whole. The forumula then takes the sequential portion and adds it to the parallel portion of the program over N which represents the number of processors this symbolizes the division of the parallel portion of the program between the processors. The upper limit that Amdahl referrs to is clear as you increase the number of processors N, if you had infinity processors working on the parallel part the speed up would simply be 1 over (1-P). [1]

There have however been criticisms of Amdahl's law Gustafson's law, Cost efficiency, Karp-Flatt metric

Non-blocking algorithm. slowdown and speedup

2.7 Algorithms and Data Structures

The vast majority of of programs or as Skiena puts it any reasonable computer program has a procedure to to accomplish a specific task. These procedures are called algorithms and play an important role throughout any programming project.[6] As the project progressing a close look will be taken at the different known algorithms which tackle the problems that arise and particular attenton will be be taken in order to use algorithms that scale well with the quantity of data passing through them. Also I will

2.8 Space Partitioning

Binary Space Partitioning and KD Trees

2.9 Collision Detection

There are several algorithms for collision detection as it is a well researched field. Collision detection is not only found in games but in many areas such as physics and other scientific simulations as well as robotics. Posteri Vs Priori ?Hybrid parallel continuous collision detection

2.10 Artificial Intelligence

2.11 Graphical Style

Chapter 3

Planning

3.1 Software Development Model

This project will take an iterative and modular development approach as it works well with the nature of the program. It is important to small modules to develop

3.2 Important Tasks

The following is a break down of the project into smaller iterations this can then be used alongside the various project deadlines I have, to produce a Gantt chart of how I plan and how the workflow will go.

- Representing an Ant
- Representing the World
- Representing the Pheremone Levels
- Individual Ant Intelligence
- Collision Detection
- Parallelize Algorithms further
- Distributing Problem

3.3 A Schedule of Activities

The produced Gantt chart attached to this report visualizes the following information. Throughout the following paragraphs I will describe my plan for the project and in the form of a schedule.

October-November Throughout the course of October and November the project will focus mainly on gathering resources and research. Much of research will be conducted in the functional languages Haskell and Erlang and small experiments will be made to demonstrate any concepts that have been learnt. Further research will be carried out in areas of the project that will pose potential problems that need to be dealt with later on in the implementation stage of the project. Even at this point in the project the implementation process will begin. As development will be iterative small aims will be set such as representing a single ant and then these aims will be expanded upon.

December-Janurary In the months of December and Janurary the code produced in the first section will be expanded upon to produce a working base system. At the end of this this period a simulation of the ant colony should be able to be run. The simulation should include the all the basic functionality of the final system. Ants will be able to move and collide. Each ant will respond to its surrounding by referencing its own basic behaviour tree.

February-March In Feburary and March I will focus on improving the algorithms I have used within the system. Making them more efficient and try and get more of them working in parallel. At the end of this period the project's delieverable, the simulation, will be in a state that is presentable with as few bugs as possible. Towards the end of this period if things are on schedule the project's focus will shift from improving the base simulation to adding extentions and making it more usable.

April During the final few weeks of the project I will be fixing any bugs in the simulation and finishing up any extentions that may be added to the project. During this time the projects focus will be documentation and producing an informative and evaluative technical report.

3.4 Risk Analysis

Throughout this project there are several things that could go wrong, such as loss of work, hardware failure and the inability to surpass certain bugs I encounter throughout the project. To minimize the chances of loss of work I will use git source control tools to not only backup my project but keep a track of changes and monitor its evolution. In the event of hardware failiure all the software I require for the project is freely available and I would be able to set up for work either in the University or on a replacement system with minimal effort. There is also a small chance that I may encounter bugs which are impossible to over come this is a risk that can only be managed through planning and research. Should I encounter difficulties there are a huge range of resources for functional programming and a Brighton functional programming user group, this would provide a great opportunity to increase my knowledge

but should I not be able to find a solution the only course of action would be to alter the future plans of the project.

3.5 Ethics

Before embarking on this project it was essential to look into any ethical issues that may arise throughout the course of the project or as a result of using the project's delieverable. So far none have no ethical issues have been encountered however ethics will be kept in mind throughout the course of the project and any issues will be documented in the project log during the project and the technical report at the end of the project.

Bibliography

[1] Gene M. Amdahl? Validity of the single processor approach to achieving large scale computing capabilities. *AFIPS Spring joint Computer Conference*, 1967.

[2] John Backus. Can programming be liberated from the von neumann style?: a functional style and its algebra of programs. *Commun. ACM*, 21:613–641, August 1978.

[3] John Hughes. Why functional programming matters. *The Computer Journal*, 32:98–107, 1984.

[4] Bryan O'Sullivan, John Goerzen, and Don Stewart. *Real World Haskell*. O'Reilly Media, Inc., 1 edition, December 2008.

[5] Andrew McGettrick Rene McCauley. Computer science curriculum 2008: An interim revision of cs 2001report from the interim review task force, 2008?

[6] Steven S. Skiena. *The Algorithm Design Manual*. Springer, second edition edition, 2010.

[7] Antero Taivalsaari. On the notion of object. *Journal of Systems and Software*, 21(1):3–16, 1993.

[8] M Vujoevi-Janii and D Toi. The role of programming paradigms in the first programming courses. *The Teaching of Mathematics*, XI(2):6383, 2008.