Vol 8. No. 2 Issue 2 – August, 2015

African Journal of Computing & ICT

www.ajocict.net

Determining an Optimal Energy level of the Artificial Ant in the Classical Santa Fe Artificial Ant Problem on the Platform of 'Genetic Programming'

D. Oghorodi
Department of Computer Science,
College of Education
Warri, Delta State, Nigeria,
oghorodiduke@yahoo.com
dukeoghorodi@gmail.com

P.O. Asagba
Department of Computer Science,
University of Port Harcourt,
Port Harcourt, River State, Nigeria
asagba.princeuniport.edu.ng

ABSTRACT

Genetic Programming researchers have used different energy levels for the Artificial Ant on the Santa Fe Trail yielding different results. The need to determine which energy level gives optimal ability of the Artificial Ant to eat more food pallets along the Santa Fe Trail motivated this research. The Evolutionary Methodology was adopted in this research. The Santa Fe Artificial Ant Problem was implemented on Genetic Programming Algorithm. Using the known ant's energy levels in literature, we observed that the Artificial Ant's ability to eat food pallets (program fitness) is highest when ant's energy level is the range of 100, 200 and 300; and beyond these, the ant's ability to eat food pallets began to drop considerably

Keywords: Genetic Programming, Santa Fe Artificial Ant Trail Problem

African Journal of Computing & ICT Reference Format:
M D. Oghorodi & P.O. Asagba (2015): Determining an Optimal Energy level of the Artificial Ant in the Classical Santa Fe Artificial Ant Problem on the Platform of 'Genetic Programming. Afr J. of Comp & ICTs. Vol 8, No. 2, Issue 2. Pp 61-70. .

1. INTRODUCTION

The concept of Darwin Theory of Evolution in computing that started in mid 60s is becoming more and more popular by the day. Evidently, in the 60s, 70s and 90s, four different implementations of Alan Turing's ideas of evolution in his proposed 'evolutionary search' were cultivated. These implementations which came specifically in 1966, 1973, 1975 and 1992 were 'Evolutionary Programming' (EP), 'Evolutionary Strategies' (ES), Genetic Algorithm (GA) and 'Genetic Programming' (GP) respectvely.

These implementations collectively designated as the Evolutionary Algorithms form the backbone of the 'Evolutionary Computation' [22]. Of the different subfields of 'evolutionary computation', Genetic Programming is a comparatively young and a rapidly growing research area[15]. However, evolutionary algorithm's solutions are satisfying 'given current resources and constraints', but not necessarily optimal [5]; but used traditionally for 'solving challenging optimization problems' [13].

However, Genetic Algorithms is the most popular as it provides the finest grained 'model of evolution' by choosing to manipulate bit strings analogous to genes on chromosome [1]. Surprisingly, Koza who was fascinated and indeed inspired by Holland's works in Genetic Algorithm overtly criticized the algorithm as being "difficult, unnatural, and overly restrictive to attempt to represent hierarchies of dynamically varying 'sizes and shapes' with fixed length character strings and for many problems in machine learning and artificial intelligence, the most natural representation for a solution is a computer program."[14]. [22] also lend his voice by saying that Genetic Algorithm has difficulty in handling problems dealing with 'deceptive' fitness functions.

One interesting thing about **Genetic programming is problem-independent algorithm that was used to solve many real life and artificial problems.** It is also an approach for problems that have no well defined efficient solution; and problems that its potential solutions can be adequately measured and compared [29]. It is also well suited to difficult control problems where no exact solution is known or required [12], [13].

One of the areas where GP was rigorously applied is 'Santa Fe Artificial Trail Problem'. Since the classical Santa Fe provides a static environment for foraging artificial ant, one expects that there ought to be an optimal energy level for the ant instead of the 'different energy levels' used in GP literature. Against this backdrop, this research focused on how to find the best energy level of a foraging artificial ant that will yield the best result in the 'Santa Fe Artificial' Trail Problem.

This research work is structured as follows: I: Introduction, II: The Real Ant behaviour and Ant Optimization Algorithms; III: 'Santa Fe Artificial Ant Problem', IV: Genetic Programming: an overview; V: methodology; VI: Results, VII: 'Discussion of results'; and VIII: 'Concluding Remarks' .

2. ANT'S BEHAVIOUR' AND ANT OPTIMIZATION ALGORITHMS

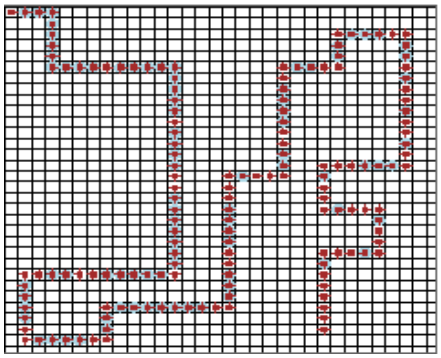
In discussing the behaviour of the artificial ant in the 'Santa Fe Artificial Ant' Problem, it is also pertinent to begin our discussion from the behaviour of the natural ant. The natural or real ants are 'social insect' that lives cooperatively in a group known as the ant colony. The ant is a 'tiny insect' which when in its colony can accomplish some complex task which is not possible with individual ant alone. However, the ant behaviour has been a subject of research in Artificial Intelligence till date.

A foraging ant leaves its nest to explore and exploits its environment for food; once it finds one, it leaves a trail of secreted chemical called trail pheromone along the source of important food to its nest. Once ant arrives its nest, other ants follow the trail to food source. As the ants move from its nest to its source of food and back, a high concentration of pheromones chemical is laid on its trail thereby stimulating both stigmergetic behavour [20] and autocatalytic process [21] of the ant. In the popular and impressive 'Diamond-Shaped Bridge' experiment denominated as 'the double bride' or 'binary bridge experiment, path optimization behaviours of stigmergetic and autocatalytic of the ant was demonstrated. 'Stigmergy' is indirect 'communication mediated' by modifications of the environment. It is the influence of another ant's due to the environment modification with its pheromone. Autocatalytic behaviour of ant is a collective behaviour. It means that as more 'ants follow a trail, it becomes more attractive for more ants to followed' such trail which causes 'very rapid converges' of the ant along the trail.

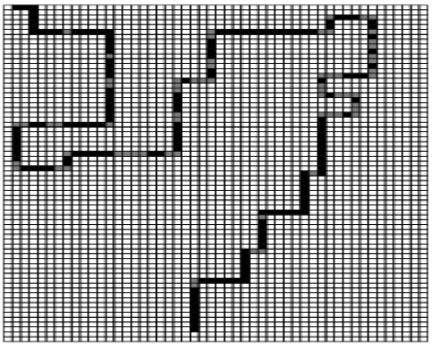
The ant optimizing behaviour is an act of an ant to follow a shorter and more reinforce path with pheromone from 'source of food' to nest or vice versa. Apart from pheromone that guides the path of the ant, [2] also demonstrated in his experiment that position of sun, gravity, slope and reference objects can also guide its direction. This experiment and others on the behaviour of the real or natural ants in their colonies stimulated several 'ant optimization algorithms' in 'Artificial Intelligence' [21]. These optimization algorithms are Ant System, System Elitist Ant, Ant-Q, Ant Colony System, Max-Min Ant System, Ranked-based Ant System, Ants and Hyper Cube-ACO[27] that use the basic ideas of search and optimization techniques.

3. THE 'SANTA FE ARTIFICIAL ANT PROBLEM'

The Artificial ant Problem is a simulation of the natural ant behaviour in a digital environment [21]. It is a 'multi-agent' method from behaviors of real ants and local search algorithm. The Artificial Ant Problem is an optimization problem developed by Jefferson et al in 1991 [9] [27] but popularized by [11]. The task of the artificial ant in the simulated ant environment is to navigate along some paths or trails in attempts to forage for all food pellets along such trails. Certain benchmark problems used in Genetic Programming are based on some of the known trails [7]; amongst which are San Fe Trail, Los Altos Hills, John Muir Trail, and Auxiliary Trail [27]. Earlier on, [7] also mentioned the San Matco Trail which was built upon the ideas behind the Santa Fe Trail which is the most famous and widely used in Evolutionary Computation. See figure1

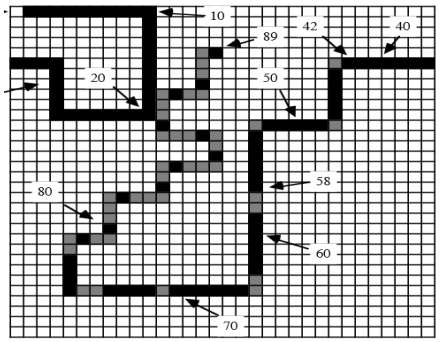


(a) Santa Fe Trail (32x32 grid)
Source: http://dev.heuristiclab.com

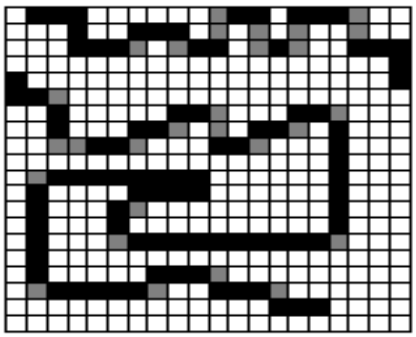


(b) Los Altos Hills trails (100x100 grid)
Source: http://citeseerx.ist.psu.edu

www.ajocict.net



(a) John Muir Trail (32x32 grid)
Source: http://www.genetic-programming.com



(b) Auxiliary Trail
Source: http://www.researchgate.net

Figure 1: Artificial Ant Trails.

The 'Santa Fe Trail Problem' using Genetic Programming is a metaphor for the historic 19th-Century transportation route of 1,400 km between America and Santa Fe in 'New Mexico' where travelers were faced with (problem or) hardship [35]. Hence it is appropriately known in computing as Santa Fe Artificial Ant Trail Problem [36]. Figuratively, the Santa Fe Artificial Ant Trail Problem is a 32 x 32 toroidal square grid having a trail length of 144 cells with 89 food pellets distributed randomly along the trail with a total of 55cells being gaps within the 89 food pellets and 21 turns with 10 left and 11 right turns along the trail [15]; [36]. Generally, the objective of the 'Santa Fe Artificial' Ant Problem is to evolve programs to control the artificial ant that will find all 89 food pellets that are located on the discrete trail [28] using a particular energy level. In traversing the trail for food pellets, the artificial ant begins from the cell identified by (0, 0) coordinate on the west of the grid and moves towards the east of the trail.

The trails is marked by black or grey fields; the black fields represent food pallets, while the grey fields represent obstacles which the ant must overcome to eat a food pellet [25]. In a related study of the artificial ant using the program landscape and schema analysis, it was concluded that the artificial ant following the 'Santa Fe Trail' is a difficult task. In agreement with this,[36] stated that the 'Santa Fe ant problem is seemingly simple problem, but with complicated dynamics; and that its hardness is as a result of 'difficulty of searching' its fitness landscape. However [10] opined that the solution to Santa Fe Artificial Ant Problem's hardness is resolvable with a reduction in the search space through the removal of ineffective operations that consume resources like energy of the ant.

The 'artificial ant' moves along the trail using a sensor that enables it to see only adjacent cell in its current direction. In other words, for the artificial ant to locate food pellets along the trail, the ant uses the food sensing function, IfFoodAhead to check the field the ant is currently facing, if it senses food pellets, the function returns the Boolean value of 'true' and then the ant moves forward into the field containing food pellet to eat it, otherwise it returns the value 'False' meaning the field is an obstacle which must be overcome. The amount of food eaten by the ant is the fitness measure of the ant controlling program. Apart from the Move operation, the artificial ant can also perform the operations of turn *right*, or *turn left* as the case may be. The speed at which artificial ant eats all '89' food pellets is important. In estimating the speed at which the ant traverses the trail for food pellets, it is important to know the ant's energy unit which is quotient of the total steps the ant takes to eat the furthest pellets over the trail length.

While trail lengths are fixed for all grids, the total steps the artificial ant must take to consume all the food pellets in the trail are predetermined for all problems. For instance in a 'Santa Fe Trail' which has a trail length of 144 cells and with a predetermined steps of 400 will have approximate 2.7 energy unit per cell (ie 400/144). Therefore whenever the ant takes a step, it consumes a unit less of its total energy[30]; [36]. [15] estimated the speed of the ant as the distance along the trail to the furthest pellet the ant eats; divided by the energy it consumed to get to that food pellet - if it does not eat any, then its speed is zero. The total speed of the ant therefore is the quotient of squares in the trail over the total steps the ant must take to eat the furthest pellets. Therefore, for 'Santa Fe Trail' that has 144 fields to be traversed with 400 steps, the speed of the ant will approximately be 0.36 units per field. Various researches used different initial energy in their works. Some used 400 steps as in [17], [30], 545 steps as in [25]; 600 steps as in [36], [15], and 615 steps as in [10]. In determining the behaviour of the artificial ant in dynamic environments, Murphy, [23] used five different energy levels: 20, 42, 60, 100, and 140 in their experiment.

The pertinent question at this juncture is: is there an optimal energy level for the artificial ant in Santa Fe Artificial Ant Problem since environment is static.

4. GENETIC PROGRAMMING (GP): AN OVERVIEW

The objective of the 'Santa Fe' Artificial Ant Problem is to evolve programs to control the artificial ant in trail [28]. The only efficient means through which one can evolve programs to direct the movement of the artificial ant to find and eat all the '89' food pellets located on the trail is by GP. Genetic Programming automatically generates program codes, rather than using lines of codes that are hand-written by programmers, in the familiar method that is time-consuming, tedious and expensive.

GP was based on the principles of Darwin's Evolution Theory. The motivation for the concept of evolution to be applied to artificial systems like dynamic program structures stem from the fact that evolution is as a non-guided, goal-directed and a preprogrammed form of growth in natural systems[11].

Historically, GP is an extension of Genetic Algorithms [6]; It is a mathematical formulation of simulated evolution as a type of search algorithm [11]. Even though Genetic Programming is an extension of Genetic Algorithm, they are similar but only differ in the ways they represent problems [11]. Unlike Genetic Algorithm, GP uses variable length computer program [17]. This makes GP more expressive as it has the flexibility needed to express solutions to different problems using computer programs; but it is 'a-time-intensive' algorithm [32].

It uses dynamic structure that evolves 'computer programs' to perform tasks by 'means of natural selection' [11]. GP is a technique of generating programmed solutions to problem automatically[29] using the genetic operations of reproduction, crossover, and mutation [17]. Genetic programming aspires to induce a population of computer programs that improve automatically as they experience the data they knew. [24] said it a machine learning method that could automatically solve problems and producing solutions with complex structure, including executable code. [19] asserted that it is a set of algorithm that mimic survival of the fittest, genetic inheritance and variation, and selectively 'breeding' of parent population of program codes or structures and replacing them with more fit 'offspring' of programs codes or structures. Gustafson (2004) added that GP iterates as it finds solution close to or equal to the ideal solution in a solution space. These computer programs that evolved are all likely 'candidate solutions'.

Therefore, we define Genetic Programming as the gradual evolution of computer programs that generate intermittent solutions known as candidate solutions in a solution space until the 'best-so-far' solution is attained. "Genetic Programming executes iteratively. It begins with initial guesses at a solution and successfully improves a solution over time. Once a termination criterion is attained, GP returns the best individual so far as the solution to the problem" [4]

Genetic Programming is a probabilistic, non-deterministic, an optimization and heuristic search technique [19]; [26]; [17]; [4]. It is probabilistic because it rarely gets a solution in precisely the form you contemplated and same result is rarely obtained twice as 'anything can happen and nothing is guaranteed'[11]. It is an optimization technique because it searches through the 'space of all possible' programs for one that has the optimal fitness [4], It is a heuristic search technique because no obvious straight and easy path exist to the best or optimal solution [17]. As a 'search technique', GP initially explores the 'solution space' for good solution and later exploits for better solution as evolution progress [3]; and hence it is generally considered to be a very time-consuming algorithm [31]. Consequently, [33] asserted that the massive disadvantage of GP is the phenomenal computing resources required before any real-world problem can be tackled. It was observed that the need for more 'computational resources' may reduce GP performance with difficult problems. However, the search for the optimal solution can only be attained by using randomization and brute-force.

Genetic Programming is specified from Koza's GP algorithm thus:

```
Procedure GP; {
 time t = 0;
 initialize population P(time);
 evaluate P(time);
  until (done) {
        t= t + 1;
        parent selection P(t);
        recombine P(t);
        mutate P(t);
        evaluate P(t);
        survive P(t);
        }
}
```

A simplified flow-diagram of Genetic Programming process is shown in figure2:

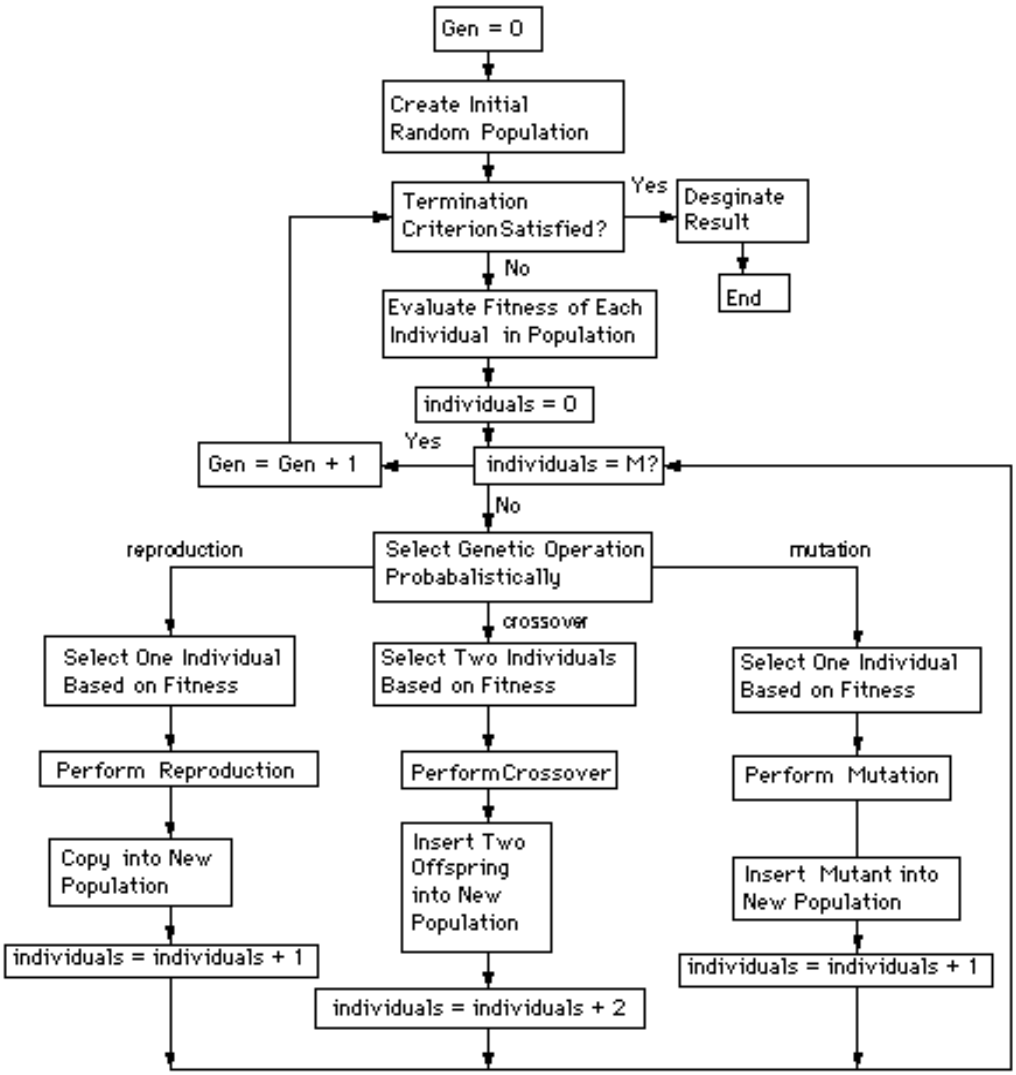


Figure2: Genetic Programming Flow-diagram.
Source: http://www.geneticprogramming.com/Tutorial

5. METHODOLOGY

In a 'traditional software development' process in which *what to process* and *how to process it* has to be clearly specified, a choice of methodology could be made from standard methodologies like the 'waterfall methodology', spiral methodology, prototyping, incremental methodology, 'Object-oriented methodology', Simulation or Structured Systems 'Analysis and Design' Methodology (SSADM). For this research where the program structures are made to evolve, appropriate methodology therefore is the Evolutionary Methodology with inclination to Object-Orientation. This methodology uses the iterative process that evolves a final solution from an initial specification of well-defined and well-understood requirements by adding new features as the evolution progresses until a termination condition is met. However, we implemented the Santa Fe Ant Problem on Genetic Programming as in Figure3

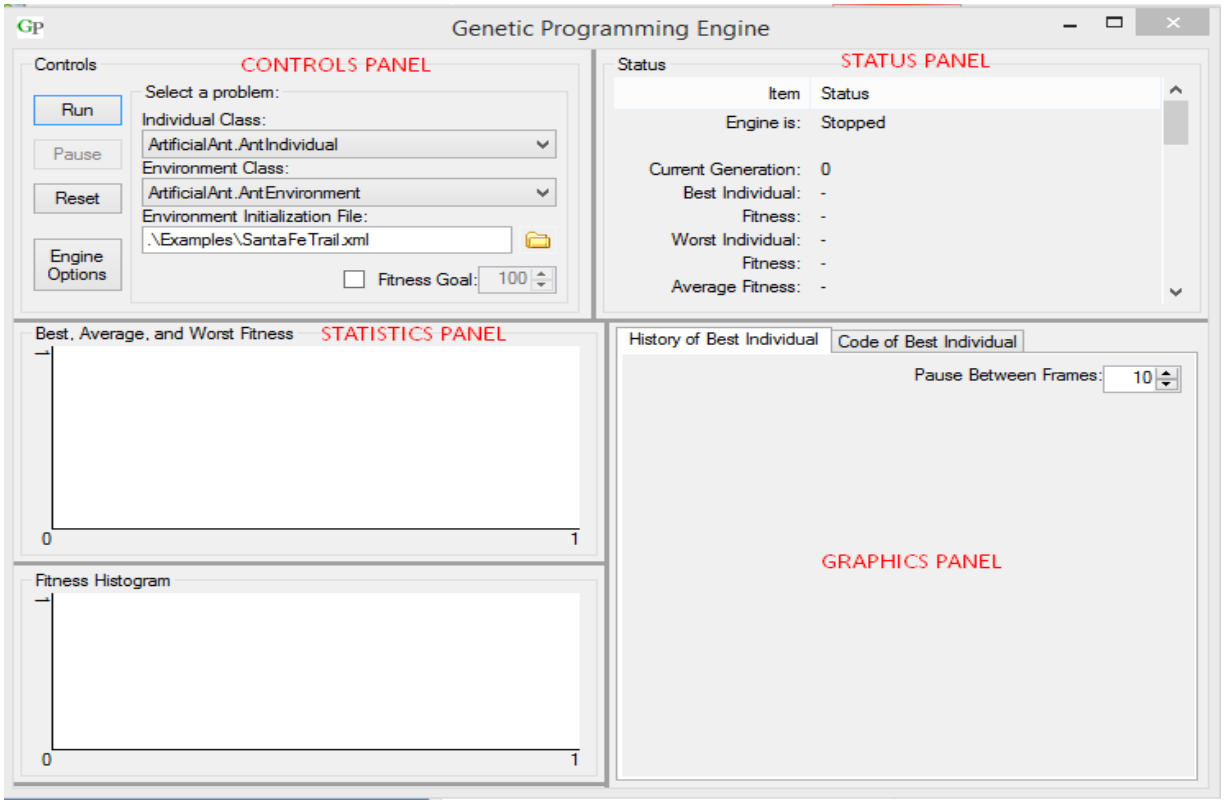


Figure3: Genetic Programming Engine.

The Santa Fe Artificial Ant Problem using Genetic Programming was implemented on a Microcomputer notebook with an Intel Pentium 250 quad core processor running on 2.16 GHz speed, with a Ram size of 4GB, a Hard disk capacity of 500GB and on Windows 8 operating system of 64-bit bus size. The Microsoft 'Integrated Development Environment' known as 'Visual Studio' was used to develop the initial software. Specifically, we used the Microsoft 'Visual Studio 2010' Ultimate ; and Visual C# (pronounced Visual C sharp) as the language of implementation. This language is Microsoft's new-generation object-orientated programming language suitable for Window and Web applications**.** The parameters used are in Table1.

Table1: Parameters for 'Santa Fe Artificial Ant Problem'

Problem	'Santa Fe Artificial Ant Problem'
Objective	To find a program that could direct an 'artificial ant' to find '89' food pallets on the 'Santa Fe Trail'.
Terminal Set	(Left), (Right), (Move)
Function Set	If-Food-Ahead, Prog2, Prog3
Fitness cases	As in See table 2
Raw fitness	Number of food pellets picked up before the artificial ant time out at its specified energy level
Parameters	Maximum population size = 500 and Generation = 51
Success Predicates	A score of '89' food pellets

6. RESULTS

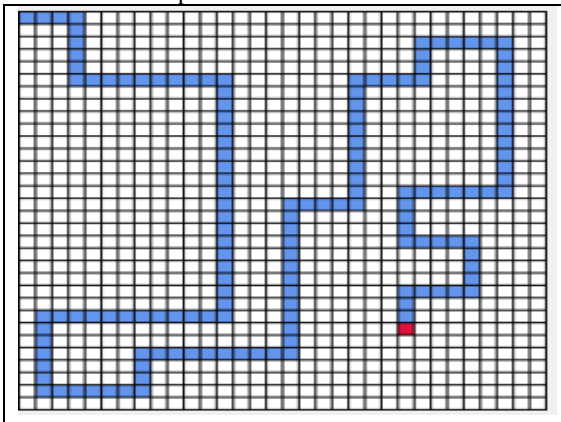
Using the algorithm on Artificial Ant problem at the different ant's energy levels identified in the literature, the following results were obtained from Figure3 and presented in Table2

Table2: Artificial Ant Problem Data

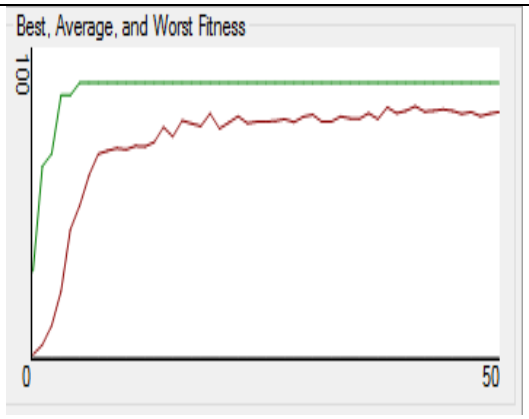
Initial Energy Level (Fitness cases)	Unit Energy (approx.)	Trail Length	Fitness goal (Food Pellets)	Number of Obstacles	Best Program Fitness	Average Program Fitness
100	0.69	144	89	55	89	79.618
200	1.38	144	89	55	88	77.194
300	2,.08	144	89	55	89	77.022
400	2.77	144	89	55	36	4.41
540	3.75	144	89	55	25	3.636
545	3.78	144	89	55	26	3.092
600	4.16	144	89	55	65	4.248
615	4.27	144	89	55	26	2.978

7. DISCUSSION OF RESULTS

In the first run of the algorithm, the Artificial Ant energy level of 100 is used with a fitness goal of 89 (ie. Number of food pallets to be eaten). Using this energy level, the ant navigated along the blue trail in Figure4 with a best program fitness at 89 meaning that the ant ate all 89 food pallets along the trail. This is shown in associated line graph in Figure 5 with the best program fitness indicated by a green line and the average program fitness with red waving line. This means that at the 'ant energy level' of 100, the 'artificial ant' performed well.



(a) Figure 4. Santa Fe Artificial trail 1



(b) Figure 5 Line graph1

In the second run, the artificial ant's energy level was increased to 200. Using this energy level, the ant navigated along the blue trail in Figure 6 the best program fitness became 88 meaning that the ant ate 88 food pallets along the trail while the average program fitness was '77. 194'. Both the best program and average program fitnesses are in Figure 7.

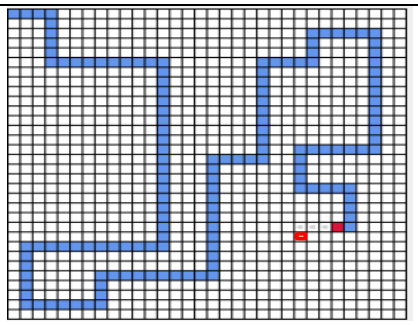
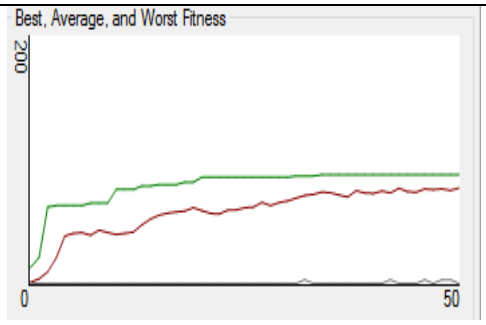


Figure 6 Santa Fe Artificial Ant trail 2



(b) Figure 7 Line graph2

In Figure 7, the green line shows the best program performance while the red waving line shows average program performance.

In third run, the artificial ant's energy level was increased to 300. Using this energy level, the ant navigated along the blue trail in Figure8 the best program fitness is 89 meaning that the ant ate all food pallets along the trail while the average program fitness was '77.022'. Both the best program and average program fitness are in Figure 9.

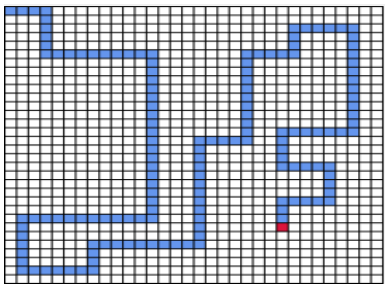


Figure 8 Santa Fe Artificial Ant trail 3

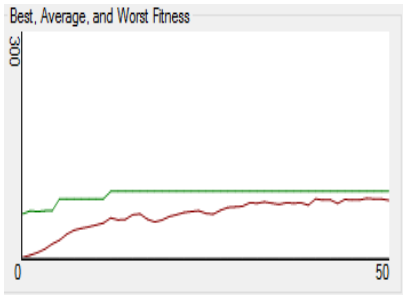


Figure 9 Line graph 3

In Figure 9, the green line indicates the best program fitness while the red waving line is the average program fitness.

The trend in the above run is that at the three energy levels of the artificial ant at 100, 200 and 300 there are no much difference in best program fitness and average program fitness and the artificial ant performance is at optimal level of performance. Beyond these level of 400, 540, 545, 600 and 615, the ant ability to eat food pallets began to drop. This is clear in figure 10.

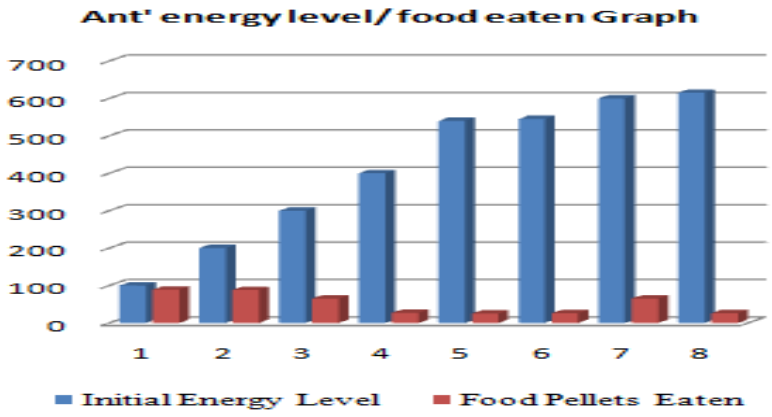


Figure 10. Ants energy level/food pallets eaten graph

In Figure 10, the first three twin bars indicate the best energy levels and food pallets eaten. Beyond these three, the food pallets eaten drops considerably as the ant energy level began to increase.

8. CONCLUDING REMARKS

Different energy levels were used in GP literature. These energy levels have yielded different experimental results. Hitherto, researchers used these energy levels because others have used them without reasons. However, this research shows that artificial ant's ability to eat food pallets along the Sante Fe Ant trail is best when the ant's energy is at 100, 200 or 300; beyond these levels the ant ability is drastically affected.

REFERENCES

[1] Angeline, P.J. (1994). "Genetic Programming and emergent intelligence". Retrieved from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.15.5594&rep=rep1&type=pd

[2] Bethe, A. (1898). Recognition of nestmates, trails. Arch. Gesamt, Physiology. 70, 15-100.

[3] Burke, E., Gustafson, S., and Kendall, G. (2002). "A Survey and 'Analysis of DiversityMeasures' in 'Genetic Programming' ". Retrieved from:http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.2.3757

[4] Dolan, K. (2009). "Beginners' Guild to Genetic Programming". Retrieved on 19-03-2013 from: http://www.geneticprogramming.us/Further_Reading.html.

[5] Eberbach, E. (2005). "Toward a theory of evolutionary computation". BioSystems, 82(1): 1-19. Retrieved from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.61.9316&rep=rep1&type=pdf

[6] Folino, G., Forestiero, A., and Spezzano, G. (2006). "A JXTA based 'Asynchronous Peer-to-Peer Implementation' of Genetic Programming ". Retrieved from: http://www.academypublisher.com/jsw/vol01/no02/jsw01021223.pdf

[7] Fagan, D, Nicolau, M., Hemberg, E., O'Neill, M., and Brabazon, A. (2011). "Dynamic Ant: Introducing a new benchmark for Genetic Programming in Dynamic Environments' ". Retrieved from: https://csiweb.ucd.ie/files/UCD-CSI-2011-04.pdf

[8] Gustafson, S.M. (2004). "Analysis of Diversity in Genetic Programming".(Doctoral dissertation, University of Nottingham). Retrieved from: http://www.gustafsonresearch.com/research/publications/ phdthesis-gustafson.pdf

[9] Jefferson, D., Collins, R., Cooper, C., Dyer, M., Korf, M. F.R., Taylor, C., and Wang, A. (1991). "Evolution as a theme in artificial Life': 'The genesys/tracker system' ". Retrieved from: http://ftp.cs.ucla.edu/tech-report/1990-reports/900047.pdf

[10] Karim, M. R. and Ryan, C. (2012). "Sensitive Ants Are Sensitive Ants". Retrieve from: https://www.lri.fr/~hansen/proceedings/2012/GECCO/proceedings/p775.pdf

[11] Koza, J. (1992a). "Genetic Programming: On the Programming of Computers by Means of Natural Selection". Retrieved from:http://www.amazon.com/Genetic-Programming-Computers-Selection-Adaptive/dp/0262111705

[12] Koza, J.R.(1992d). "A genetic approach to finding a controller to backup a 'tractor-trailer' ". American Control Conference, Chicago, IL, USA: IEEE, (3): 2307-2311.

[13] Koza, J.R (1992e). "A genetic approach to the 'truck backer upper Problem' and the 'inter-twined spirals problem' ". Retrieved from: http://ieeexplore.ieee.org/iel2/632/5902/00227324.pdf

[14] Koza, J. (1994h). " 'Genetic Programming' as a Means for 'Programming Computers by Natural Selection' ". Retrieved from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.127.6987

[15] Langdon, W. B and Poli, R (1998a). "Better Trained Ants for Genetic Programming". Technical Report CSRP-98-12, University of Birmingham. Retrieved from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.46.634&rep=rep1&type=pdf

[16] Liang W. and Huang, C. (2009). " The 'generic genetic algorithm' incorporates with 'rough set theory'- An application of the 'web services composition' ". Retrieved from: http://isiarticles.com/bundles/Article/pre/pdf/29508.pdf

[17] Luke, S. (2000). " Issues in 'Scaling Genetic Programming': 'Breeding Strategies', 'TreeGeneration', and 'Code Bloat' ", PhD Dissertation, University of Maryland, Computer science department. Retrieved from: https://cs.gmu.edu/~sean/papers/thesis1p.pdf

[18] Maumita B. , Ajith A., and Baikunth N. (2001). "A Linear Genetic Programming Approach for Modeling Electricity Demand Prediction in Victoria." Retrieved from:http://www.researchgate.net/publication/220981219

[19] McConaghy, T., ladislavleva, E., and Riolo, R. (2010). "Genetic ProgrammingTheory and Practice : An Introduction". Retrieved from: http://www.cs.mun.ca/~tinayu/Publications_files/gptp2005-1.pdf

[20] Merloti, P., E. and Lewis, J. (2005). "Simulation of Artificial Ant's Behaviour in a Digital Environment" Retrieved from:. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.116.7759&rep=rep1&type=pdf

[21] Merloti, P., E. (2004). "Optimization Algorithms Inspired by Biological Ants and Swarm Behaviour". Retrieved from: http://www.merlotti.com/EngHome/.../AntsSim/AntOptimizationAlg.pdf

[22] Mitchell, M.(1995). "Genetic Algorithm: Overview". Complexity,1(1)31-39. Retrieved from: http://ohm.ecce.admu.edu.ph/wiki/pub/Main/ResearchProjects/mitchell_GA_tutorial.pdf.

[23] Murphy E., O'Neill, M., and Brabazon, A. (2011). "A 'comparison of GE and TAGE' in 'dynamic environments' ". Proceedings of the 13th annual conference on Genetic and evolutionary computation held in Dublin, Ireland. p431-438.'

[24] O'Neil, M. and Brabazon, A. (2009). "Recent Patents on GeneticProgramming". Retrieved from: http://www.benthamscience.com /cseng/samples /cseng2-1/0005CSENG.pdf

[25] Oplatková, Z. and Zelinka, I (2006). "Santa Fe Trail For Artificial Ant With Simulating Annealing – Preliminary Study ". Retrieved from: http://www.scs-europe.net/services/ecms2006/ecms2006%20pdf/77-is.pdf

[26] Poli, R. and Langdon, W.B. (1998). "On the Search of Different 'Crossover Operators' in Genetic Programming". Retrieved from: http://dces.essex.ac.uk/staff/rpoli/papers/Poli-GP1998.pdf

[27] Roy, S. (2013). "Bio-'inspired Ant Algorithms': A review". Retrieved from http://www.mecs-press.org/ijmecs/ijmecs-v5-n4/IJMECS-V5-N4-4.pdf

[28] Salehi-Abari, A., and White T. (2009). "The Uphill Battle Of Ant Programming Vs. GeneticProgramming". Retrieved from: http://www.cs.toronto.edu/~abari/papers/UphillBattle.pdf

[29] Soule, T. (1998). "Code Growth in Genetic Programming". PhD Dissertation, College of Graduate Studies, Computer Science Department, University of Idaho, USA. Retrieve from: http://www,citeseerx.ist.psu.edu/viewdoc/doi=10.1.1.38.8938&rep=rep1&type=pdf

[30] Suguira, H., Mizuno, T., and Kita, E. (2012)." Santa Fe Trail Problem Solution Using Grammatical Evolution". International Conference on Industrial and Intelligent Information. 5 (4): 36-40. Retrieved from: http://www.ipcsit.com/vol31/007-ICIII2012-C0015.pdf

[31] Tsutsui, S, and Collet, P. (2013). "Massively Parallel Evolutionary Computation onGPGPUs". Retrieved from: https://www.books.google.com.ng/books?id

[32] Wei, Y. (2003). "Profitable, Return Enhancing' Portfolio Adjustments- An Application of Genetic Programming with Constrained Syntactic Structures". M.Sc. degree project of the University College, London. Retrieved from:http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.57.4166

[33] Walker, M. (2001). "Introduction to Genetic Programming". Retrieved from: http://www.cs.montana.ed/ ~ bwall/cs586/introduction-to-gp.pdf

[34] Wikipedia (2015)." Artificial Ants". Retrieved from: https://en.wikipedia.org/wiki/Artificial_ants

[35] Wikipedia (2015). "Santa Fe Trail". Retrieved from: https://en.wikipedia.org/wiki/Santa_Fe_Trail .

[36] Wilson, D. and Kaur, D. (2013). "How Santa Fe Ants Evolve". Retrieved from: http://arxiv.org/ftp/arxiv/papers/1312/1312.1858.pdf