Mining Flat Datasets with Evolutionary SQL Statements

Chris Dilger (101133703)
COS20015 – Fundamentals of Data Management

Abstract
A tool for mining datasets of unknown quality and relevance is conceptualised. A GA is built using a fitness function defined by the semantic similarity between words in rows and columns and a question posed by a researcher. How useful generated SQL queries are at summarising flat datasets is investigated. The result is evaluated using a ground truth defined by intuitive human reasoning. The implementation of the GA and associated fitness function were found to be qualitatively insufficient to answer the research questions posed. Further work and refinement on the implantation of the GA and fitness function are proposed.

1. Introduction
A need for tools to reduce the size and scope of flat datasets has been identified. In several disciplines large flat data tables containing large amounts of useful data, useless metadata and often many sparse or unpopulated columns need to be evaluated by a human in the context of research in that discipline. An example research question might be, "How many discoveries has Martin made since 1990?" which a human will interpret, create a relevant SQL query to return the data allowing the user to answer this research question. This human intuition step is a viable candidate for machine learning, as there is a large amount of data that must be processed manually by a human in order to generate the relevant query.

One such approach would use a genetic algorithm (GA), which is a population-based global search technique based on Darwinian evolutionary theory (Holland 1975; Goldberg 2006). A GA has three elementary steps, namely selection, crossover and mutation, each outlined with pseudocode in Figure 1. A genetic algorithm works by defining a population of individuals, each of which are a potential solution to the specified problem. Each generation of the population goes through a the steps of keeping the best individuals as evaluated by some fitness function (selection), generating new individuals using the traits of fit individuals (crossover) and introduction of some random variations to the individuals (mutation). Provided that a fitness landscape can be defined, a GA is suited to finding optimal solutions in that fitness landscape.

In order for a GA to be able to properly transmit useful traits to new generations the genetic information must be directly encoded. If SQL strings were to be composed by randomly assigning AND and OR combinations of the WHERE clauses, this information would be indirectly encoded. Thus the information contributing to a more fit individual would not be passed on to each next generation. Thus, a tree data structure is useful for faster convergence in some domains, and in this case for representing order and Boolean algebra combinations of predicates in the WHERE clause (Palmer & Kershenbaum 1994; Irene Moser 2017; Stanley & Miikkulainen 2002). This awareness of the

representation of the data in the context of SQL strings and the application of a GA is a requisite for a functional GA implementation.

SGA (P_i, NG, SP, F_i, Par_{sel}, Par_{mut}, Par_{cr})
If $[P_i] = []$, Initialize (P_i, SP, NDV)
for $i = 1$: NG
$[P_i^d] = P_i$
If required, $[P_i^d]$ = Decoding (P_i)
If $[F_i] = []$, $[F_i]$ = Fitness_Calculation (P_i^d)
$[P_{i-\text{sel}}]$ = Selection(P_i, F_i, Par_{sel})
$[P_{i+1-\text{sel}}]$ = Selection(P_i, F_i, Par_{sel})
$[P_{\text{new}}] = P_{\text{new}}$ U Crossover ($P_{i-\text{sel}}$, $P_{i+1-\text{sel}}$, Par_{cr})
U Mutation($P_{i-\text{sel}}$, $P_{i+1-\text{sel}}$, Par_{mut})
$[P_i] = [P_{\text{new}}]$
end

Figure 1 - SGA Pseudocode (Talaslioglu 2009)

1.1. Research Question

The broader goal of this project is to implement a pipeline which will take a user's question, and return a subset of a flat database as a query to the user which allows the user to perform some trivial further analysis to answer their research question. For example, returning the count of discoveries that Martin made is not as useful as returning the set of all of Martin's discoveries and allowing the user to first verify the correctness of the solution and then performing a trivial count operation. The solution takes more than 12 hours to find a plateau of the fitness of generated individuals based on 9 preliminary trials on NeCTAR and a workstation computer. As is shown in Figure 1 all GA operations depend on the individual and is thus a viable candidate for optimisation efforts.

This investigation seeks to determine whether a GA can be used to provide useful subsets of a flat database. It would be expected that some useful results can be produced in an extended period of time, as the semantic similarity between a question and the resulting dataset should be high in most cases. A GA is designed to find optimal solutions in a given fitness space, and provided that a fitness landscape can be defined for the GA to operate on, it would be expected that the GA find optima in that fitness landscape.

2. Methods

In order to build and test the proposed system, a variety of technologies are incorporated into the solution. The system outlined has been used to run a number of experiments testing the convergence time of the GA in addition to the resulting subset of the database.

2.1. Technology

This project integrates many supporting technologies to evaluate semantic similarity, and evolve the GA. A layout of these technologies is shown in Architecture below. Each of the technologies has links or references to scripts used to load or manipulate data.

2.1.1. Architecture

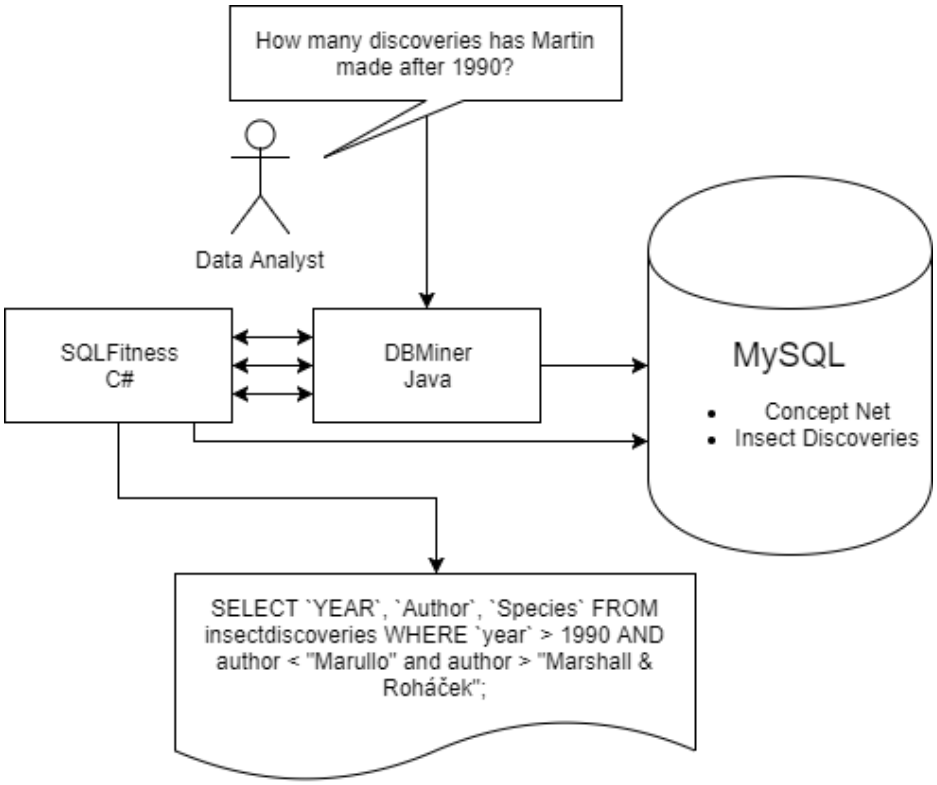

Figure 2 - Architecture of the data summarisation system

2.1.2. Genetic Algorithm

Design of the GA in C# used existing literature to select implementations of the elementary operations best suited to the chosen problem domain. An elitist selection method was chosen for the selection step, as this guarantees an the fittest individual will be at least as fit as the previous generation, while tending to find local maxima (Palmer & Kershenbaum 1994).

The construction of the individual is domain specific and important for a functional GA to have a well optimised chromosomal representation of important factors. Order of operation is encoded in a binary tree inside of the Tree individual. A direct encoding of the selection information in an individual is essential for the GA to be able to crossover new fitter individuals. Otherwise, if an indirect encoding method is used, information that contributes to the fitness of the individual is lost and is not passed down to new individuals. See in Figure 3, the order and encoding in a Object UML diagram.

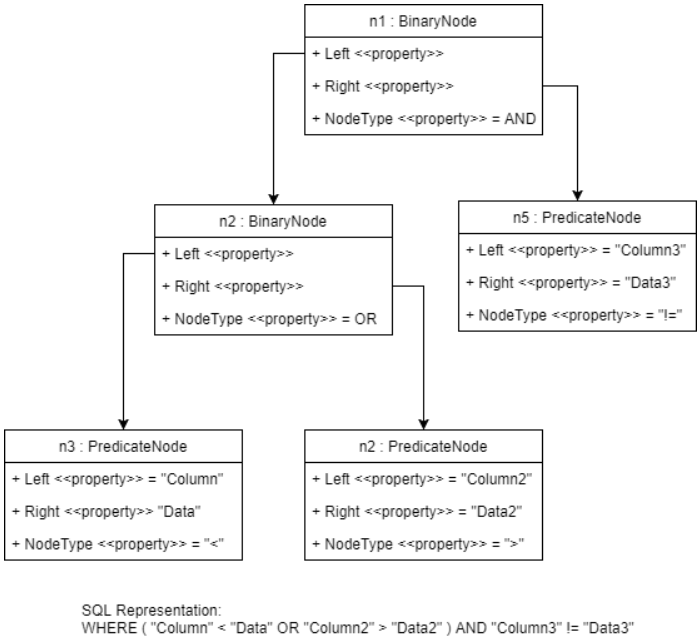

Figure 3 - How an SQL WHERE clause is represented in a TreeIndividual

Mutation in the GA is controlled by a rate parameter, which will cause a random individual to have a random gene in the chromosome reinitialised. In a Flat individual, this replaces the single gene in the chromosome array and a Tree individual will replace the Node and all predicate nodes. The tree structure cannot be modified, so that a the type of the Node is guaranteed not to change.

This fitness landscape is defined by an associated Java program DBMiner. This fitness evaluation program has been developed alongside the GA with the role of evaluating the fitness of individuals in the population, using Stanford's CoreNLP (Manning et al. 2014) to extract keywords from the question and ConceptNet (Liu & Singh 2004) to compare the semantic meaning of resulting columns and rows from the resultant dataset after executing the SQL generated by the individual. Parallel inter-application communication with DBMiner is done using TCP Sockets to evaluate individuals. This process is outlined in Figure 2.

2.1.3. Supporting Technologies

Expanding on the architecture diagram, a number of technologies were used to load data, keep track of source code or distribute runtimes. Docker was used to run the MySQL server (Anon 2017), which was useful as the server could be shared between development computers. The Dockerfile which was written using the documentation as a guide is available on the associated GitHub Repository (see Code Release).

NeCTAR was used to run the code, making use of Mono to run the C# code. Binaries were uploaded using FTP. Git was used to distribute source code between developers, and maintain history. This has a variety of benefits as bugs can be debugged easier, when changes to the code are tracked with fine granularity. To run the code on NeCTAR the Unix tool 'screen' was used to detach terminals and keep sessions running for many hours.

2.1.4. Domain Data

Only one domain has been investigated to date, using the insect discoveries dataset from (Mesibov 2014). This dataset was chosen because it was reasonably small to test on, was a real dataset and had actual applications our given problem domain. This dataset has been used for research and it is feasible that a researcher would ask questions about this dataset. To load this dataset, a CSV file was loaded via MySql Workbench into MySQL.

2.1.5. Code Release

SQLFitness: https://github.com/cdilga/SQLFitness

DBMiner: https://github.com/cdilga/DBMiner

3. Results

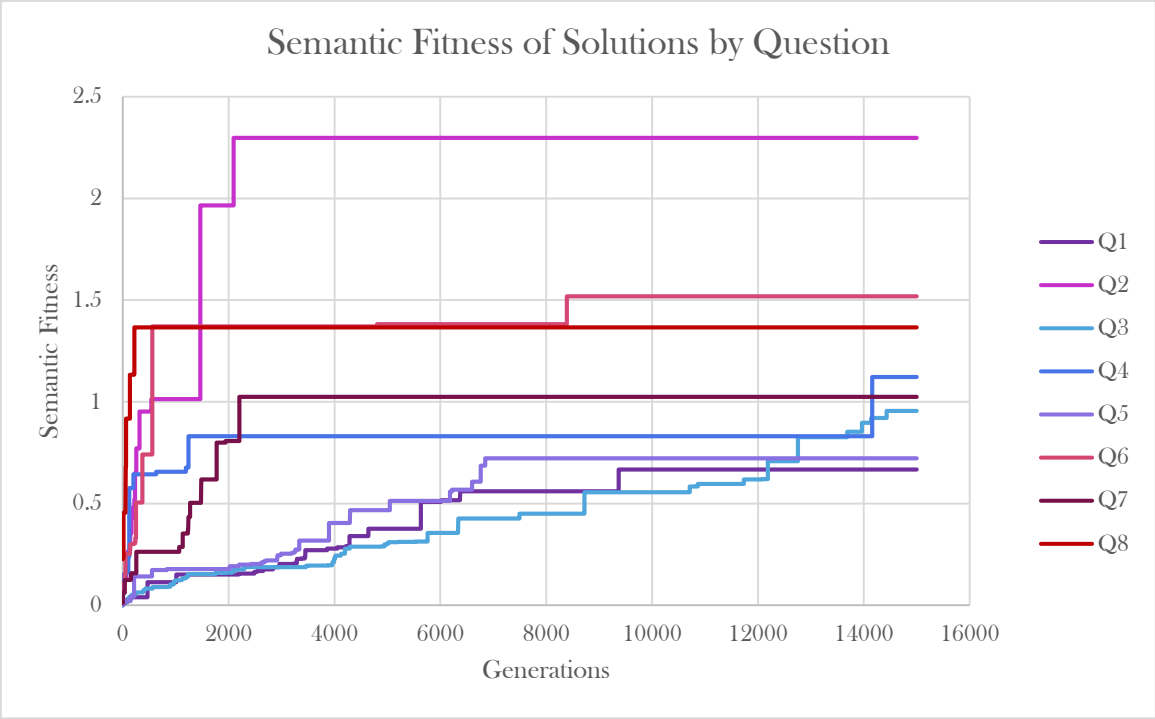

Figure 4 - Semantic fitness of each question by iteration. Only one experiment was done for each individual

Figure 4 shows the increase in fitness by iteration. We see a wide variation in the amount of time taken to find a fitness plateau by question, between 442 generations in Q8 and greater than 15000 generations for Q3. Literal natural language queries are outlined in Appendix I with their corresponding values at 1500 iterations.

Determined by our intuition, the best resulting dataset from the question, Q1 ("How many discoveries Martin has made since 1990") would be similar to the resultant set from Query 1.

```
SELECT `YEAR`, `Author`, `Species` FROM insectdiscoveries WHERE `year` > 1990
AND `Author` < "Marullo" and `Author` > "Marshall & Roháček";
```

Query 1

This result would assume that the GA has determined that "Year", "Author" and "Species" is related, semantically to the components in the question, "since", "discoveries" and "Martin".

The result found, from Appendix II is equivalent to Query 2, which is very loosely related to Q1

```
SELECT * FROM insectdiscoveries WHERE"Taxon" <= "Auchenorrhyncha";
```

Query 2

This result is qualitatively irrelevant to the query and thus is given a poor qualitative result, despite the resultant dataset having a high semantic similarity to Q1. Note that the simplified Query 2 has many

repeated projection elements as is shown in Appendix II. DBMiner will therefore count the elements returned multiple times and get an artificially higher fitness because duplicates are included.

4. Discussion

The results in indicate the system is largely dysfunctional and is unable to produce results related to the given natural language query. Largely driven by the fact that duplicates are allowed, which are rewarded due to the inclusion of more words of high semantic similarity. Additionally, the DBMiner fitness evaluation doesn't penalise invalid data in its current state. The graphs shown generally jump when a crossover operation includes a large number of duplicate projection statements. As the resulting dataset is a set, duplicates make little sense in the context of most conceivable research questions with the fundamental operations of projection and selection.

Questions like Q3, "In which decade were the most discoveries made" are currently unanswerable with the selection and projection operations included. Group by statements should be introduced to group by year and thus return a more useful resultant dataset. It would be predicted that the semantic information of returned "COUNT(x)" column names would be sufficient to allow the semantic similarity test of the fitness function to assign an appropriate fitness.

Finally, some bugs have been identified in which some race condition involving MySQL connections in the DBMiner application cause individuals to be incorrectly evaluated. The bug will return a fitness of 0 even when the query is valid due to exceptions being thrown. As a result, the experiment must be run again as the integrity of the data is uncertain.

5. Conclusion

The results were inconclusive, and further improvements to the DBMiner and SQLFitness program are necessary before a valid conclusion can be reached. First and foremost, the duplicates must be removed. Following from this, race conditions and a penalty for including irrelevant data must be imposed on the fitness function in DBMiner. In addition to these basic improvements, there are many architectural opportunities for vast optimisations. From a data mining perspective, waiting 12 hours for a query which can be intuitively determined is less time is impractical, and thus several optimisations including the reading of large data structures like ConceptNet from a binary file and centralising intensive computations rather than using TCPSockets may improve performance dramatically.

To decide on which method would best suit the SQLFitness GA, more extensive testing must be done in order to determine the difference in convergence time for each implementation which can be done on the NeCTAR cloud research platform.

6. Bibliography

Anon n.d., 'BigQuery - Analytics Data Warehouse', *Google Cloud Platform*, viewed 6 November, 2017a, <https://cloud.google.com/bigquery/>.

Anon n.d., 'c# - Using LINQ to remove any value that is a duplicate - Stack Overflow', viewed 6 November, 2017b, <https://stackoverflow.com/questions/20906701/using-linq-to-remove-any-value-that-is-a-duplicate>.

Anon n.d., 'Cloud Natural Language API', *Google Cloud Platform*, viewed 6 November, 2017c, <https://cloud.google.com/natural-language/>.

Anon n.d., 'Google Testing Blog', *Google Testing Blog*, viewed 8 November, 2017d, <https://testing.googleblog.com/>.

Anon n.d., 'Multithreaded Applications (C# and Visual Basic)', viewed 22 October, 2017e, <https://msdn.microsoft.com/en-us/library/ck8bc5c6(v=vs.110).aspx>.

Anon 2017, 'mysql', *Docker Documentation*, viewed 14 November, 2017, <https://docs.docker.com/samples/library/mysql/>.

Anon n.d., 'Natural Language Interface for Databases | KUERI.ME', viewed 6 November, 2017f, <http://kueri.me/>.

Anon n.d., 'Resolvr', *Devpost*, viewed 6 November, 2017g, <http://devpost.com/software/resolvr>.

Anon n.d., 'StringBuilder Class (System.Text)', viewed 3 November, 2017h, <https://msdn.microsoft.com/en-us/library/system.text.stringbuilder(v=vs.110).aspx>.

Anon n.d., 'Text Classification made easy with Elasticsearch | Elastic', viewed 6 November, 2017i, <https://www.elastic.co/blog/text-classification-made-easy-with-elasticsearch>.

Anon n.d., 'When to use Dependency Injection', *Google Testing Blog*, viewed 8 November, 2017j, <https://testing.googleblog.com/2009/01/when-to-use-dependency-injection.html>.

Butey, PK, Meshram, S & Sonolikar, RL 2012, 'Query Optimization by Genetic Algorithm', *Journal of Information Technology and Engineering*, vol. 3, no. 1.

dotnet-bot n.d., 'Pattern Matching - C# Guide', viewed 3 November, 2017, <https://docs.microsoft.com/en-us/dotnet/csharp/pattern-matching>.

Elsevier n.d., '11 steps to structuring a science paper editors will take seriously', *Elsevier Connect*, viewed 10 November, 2017, <https://www.elsevier.com/connect/11-steps-to-structuring-a-science-paper-editors-will-take-seriously>.

Gamma, E, Helm, R, Johnson, R & Vlissides, J 1995, *Design Patterns: Elements of Reusable Object-oriented Software*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Goldberg, DE 2006, *Genetic algorithms*, Pearson Education India.

Holland, JH 1975, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI.

Irene Moser 2017, 'Research Meeting'.

Knoernschild, K 2002, *Java Design: Objects, UML, and Process*, Addison-Wesley, viewed <https://books.google.com.au/books?id=4pjbgVHzomsC>.

Liu, H & Singh, P 2004, 'ConceptNet &Mdash; A Practical Commonsense Reasoning Tool-Kit', *BT Technology Journal*, vol. 22, no. 4, pp. 211–226.

Manning, CD, Surdeanu, M, Bauer, J, Finkel, J, Bethard, SJ & McClosky, D 2014, 'The Stanford CoreNLP Natural Language Processing Toolkit', *Association for*

Computational Linguistics (ACL) System Demonstrations, pp. 55–60, viewed <http://www.aclweb.org/anthology/P/P14/P14-5010>.

Mesibov, R 2014, 'A dataset for examining trends in publication of new Australian insects', *Biodiversity Data Journal*, vol. 2, p. e1160.

Palmer, CC & Kershenbaum, A 1994, 'Representing trees in genetic algorithms', *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, IEEE, pp. 379–384.

rpetrusha n.d., 'Managed Execution Process', viewed 23 October, 2017, <https://docs.microsoft.com/en-us/dotnet/standard/managed-execution-process>.

Salim, M & Yao, X 2002, 'Evolving SQL queries for data mining', *International Conference on Intelligent Data Engineering and Automated Learning*, Springer, pp. 62–67.

Stanley, KO & Miikkulainen, R 2002, 'Evolving neural networks through augmenting topologies', *Evolutionary computation*, vol. 10, no. 2, pp. 99–127.

Talaslioglu, T 2009, 'A New Genetic Algorithm Methodology for Design Optimization of Truss Structures: Bipopulation-Based Genetic Algorithm with Enhanced Interval Search', *Modelling and Simulation in Engineering*, Research article, viewed 11 November, 2017, <https://www.hindawi.com/journals/mse/2009/615162/>.

tutorialspoint.com n.d., 'C# - Multithreading', *www.tutorialspoint.com*, viewed 22 October, 2017, <https://www.tutorialspoint.com/csharp/csharp_multithreading.htm>.

Watson, T & Rakowski, T 1995, 'Data mining with an evolving population of database queries', *Proceedings of MENDEL*, pp. 26–28.

Whitley, D 1994, 'A genetic algorithm tutorial', *Statistics and computing*, vol. 4, no. 2, pp. 65–85.

7. Appendix I

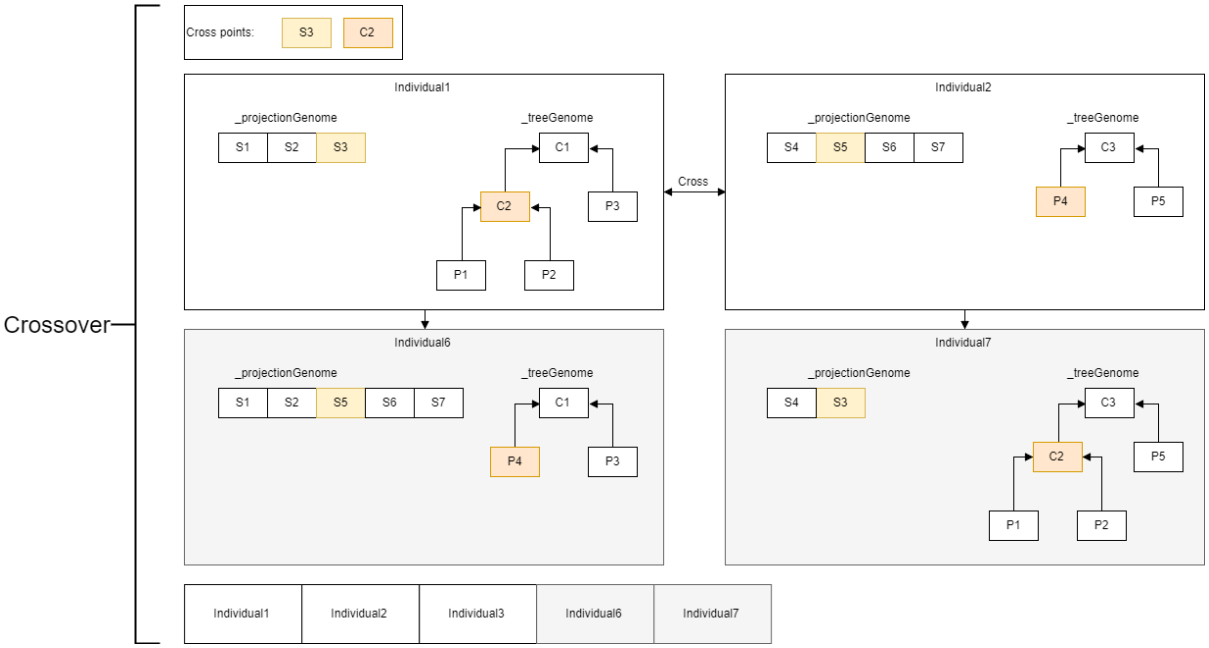

Figure 5 - A tree individual in this GA has a hybrid chromosome, with both a list based collection of projections as genes and a tree based representation of selection with nodes as genes. Crossover must therefore operate on each of these structures in isolation.

8. Appendix II

Table 1 - Questions and their results from the SQLFitness GA

ID	Question	Fitness	Query
Q1	How many discoveries Martin has made after 1990?	0.67	SELECT `Subgenus`, `A in B`, `Taxon`, `Pub ento`, `Name`, `Full name`, `Author`, `Full name`, `Full name`, `Order`, `Pub ento`, `Rank`, `A in B`, `Year`, `Publication country`, `Citation`, `A in B`, `Publication`, `Publication type`, `Full name`, `Journal`, `Taxon`, `Full name`, `Subspecies`, `Citation`, `A in B`, `Subspecies`, `Publisher class`, `Subspecies`, `Publication`, `Number authors`, `Journal`, `Journal`, `Citation`, `Subspecies`, `Publisher class`, `Subgenus`, `Taxon`, `Publisher class`, `Publication type`, `Genus`, `Subgenus`, `Pub ento`, `Citation`, `Rank`, `Genus`, `Genus`, `Original combination`, `Publication`, `Author`, `Journal`, `Order`, `Pub ento`, `Publisher class`, `Subspecies`, `Full name`, `Genus`, `Publication type`, `Author`, `Publication type`, `Rank`, `Publisher class`, `Rank`, `Citation`, `Journal`, `Subgenus`, `Author`, `Publication country`, `Taxon`, `Number authors`, `Subspecies`, `Full name`, `A in B`, `Subgenus`, `Author`, `Year`, `Journal`, `Publisher`, `Number authors`, `Subgenus`, `Publisher`, `Citation`, `Order`, `Journal`, `Year`, `Pub ento`, `Original combination`, `Citation`, `Publisher`, `Publication country`, `Year`, `Publication type`, `Original combination`, `Full name`, `Order`, `Publication`, `A in B`, `Original combination`, `Journal`, `Publisher class`, `Number authors`, `Year`, `Pub ento`, `Number authors`, `Year`, `Full name`, `Subgenus`, `Original combination` FROM insectdiscoveries WHERE"Taxon" <= "Auchenorrhyncha";
Q2	Who has discovered the most new insect species?	2.30	SELECT `Publisher class`, `Publication`, `Rank`, `Name`, `Rank`, `Publisher`, `Publisher`, `Publication type`, `Genus`, `Taxon`, `Author`, `Order`, `Name`, `A in B`, `Journal`, `Publication country`, `Publisher`, `Species`, `Publication type`, `Rank`, `Rank`, `Subspecies`, `Genus`, `Publication type`, `Species` FROM insectdiscoveries WHERE ("A in B" = "no") OR ("Species" < "rugifrons");
Q3	In which decade were the largest numbers of discoveries made?	0.95	SELECT `Publication country`, `Publisher`, `Name`, `Rank`, `Number authors`, `Year`, `Journal`, `Publisher class`, `Name`, `Number authors`, `Subgenus`, `Citation`, `A in B`, `A in B`, `Pub ento`, `Journal`, `Taxon`, `Pub ento`, `Rank`, `Genus`, `Rank`, `Order`, `Journal`, `Rank`, `Journal`, `Publisher`, `Order`, `Citation`, `Subspecies`, `Journal`, `Original combination`, `Rank`, `Journal`, `Journal`, `Author`, `Taxon`, `Order`, `Name`, `Name`, `Name`, `Year`, `Full name`, `Year`, `Full name`, `Original combination`, `Publisher`, `Author`, `Citation`, `Journal`, `Publisher`, `Publisher class`, `Subspecies`, `Journal`, `Subgenus`, `Citation`, `Year`, `Journal`, `Year`, `Journal`, `Publication`, `Genus`, `Publication country`, `Rank`, `Author`, `Order`, `Species`, `Citation`, `Full name`, `Subgenus`, `Species`, `Original combination`, `Rank`, `Subspecies`, `Subgenus`, `Order`, `Pub ento`, `Publication type`, `Publication country`, `Species`, `Number authors`, `Number authors`, `Order`, `Subspecies`, `Year`, `Subgenus`, `Number authors`, `Author`, `Publisher class`, `Journal`, `Year`, `A in B`, `Order`, `Species`, `Pub ento`, `Pub ento`, `A in B`, `Publication`, `Name`, `Order`, `Name`, `Species`, `Order`, `Citation`, `Rank`, `Publisher class`, `Number authors`, `Publication type`, `Pub ento`, `Year`, `Name`, `Journal`, `Rank`, `Publication`, `Journal`, `Pub ento`, `A in B`, `Author`, `Original combination`, `Pub ento`, `Subgenus`, `Publication`, `Author`, `Author`, `Citation`, `Pub ento`, `Citation`, `Publication`, `Name`, `Name`, `Name`, `Taxon`, `Publication country`, `A in B`, `Subspecies`, `Name`, `Name`, `Year`, `Citation`, `Original combination`, `Order`, `Publisher class`, `Citation`, `Original combination`, `Publication`, `Genus`, `Publisher class`, `Publisher`, `Rank`, `Publication type`, `Order`, `Subgenus`, `Rank`, `Taxon`, `Subgenus`, `Rank`, `Genus`, `Number authors`, `Rank`, `Publication type`, `Rank`, `Pub ento`, `Subspecies`, `Subgenus`, `Original combination`, `Publication`, `Publisher class`, `Genus`, `Subgenus`, `Year`, `A in B`, `Citation`, `Publication`, `Publication country`, `Author`, `Publisher`, `Full name`, `Publication country`, `A in B`, `Publisher class`, `Citation`, `Subspecies`, `Rank`, `Taxon`, `Name`, `A in B`, `Taxon`, `Name`, `Genus`, `Original combination`, `Rank`, `Publication type`, `Publication country`, `Subspecies`, `Publication type`, `Rank`, `Species`, `Taxon`, `Taxon`, `Publisher`, `Genus`, `A in B`, `Subgenus`, `Full name`, `Publication country`, `Order`, `Order`, `Publisher`, `Citation`, `Original combination`, `Publication`, `Species`, `Publication country`, `Publisher`, `Rank`, `Journal`, `Order`, `Rank`, `Publication`, `Publisher`, `Genus`, `Pub ento`, `Rank`, `Publication country`, `Species`, `Original combination`, `Rank`, `Rank`, `Citation`, `Publisher`, `Journal`, `Publication`, `Full name`, `Rank`, `Original combination`, `Year`, `Species`, `Subspecies`, `Number authors`, `Original combination`, `Rank`, `Subspecies`, `Genus`, `Citation`, `Publisher class`, `Year`, `Name`, `Citation`, `Full name`, `Subspecies`, `Rank`, `Full name`, `Subgenus`, `Original combination`, `Original combination`, `Pub ento`, `Publication`, `A in B`, `Publication country`, `Taxon`, `Publisher class`, `A in B`, `Publication country`, `Name`, `Number authors`, `Name`, `Order`, `Pub ento`, `Subgenus`, `Rank`, `Author`, `Order`, `Original combination`, `Taxon`, `Taxon`, `Rank`, `Author`, `Taxon`, `Name`, `Author`, `Name`, `Publisher class`, `Pub ento`, `Subspecies`, `Author`, `Citation`, `Publisher`, `Publication`, `Original combination`, `Rank`, `Full name`, `Genus`, `Original combination`, `Publisher`, `Original combination`, `Full name` FROM insectdiscoveries WHERE ((("Name" = "Valid") AND ("A in B" < "no")) AND ("Subgenus" = "")) OR ("Citation" <> "Papp J. 1981. Three new Orgilus Haliday species from the Indo-Australian Region (Hymenoptera: Braconidae: Mimagathidinae). Annales Historico-Naturales Musei Nationalis Hungarici (Zoologica) 73: 253-261");

9. Appendix III

Table 2 - Questions and their results from the SQLFitness GA

ID	Question	Fitness	Query
Q4	How many species were discovered in the last decade?	1.12	SELECT `A in B`, `Citation`, `Publication type`, `Pub ento`, `Full name`, `Author` FROM insectdiscoveries WHERE (("Author" > "Hendrich") AND ("Publication country" >= "UK")) AND ("Rank" >= "Species");
Q5	How many discoveries Martin has made in Australia?	0.72	SELECT `Publisher class`, `Author`, `Citation`, `Publication country`, `Citation`, `Name`, `Publication`, `Genus`, `Publication`, `Publisher`, `Publisher`, `Full name`, `Rank`, `Year`, `A in B`, `Journal`, `Number authors`, `Publication country`, `Subgenus`, `Publisher`, `Name`, `Pub ento`, `Publisher`, `Number authors`, `Number authors`, `Publisher`, `Pub ento`, `Subspecies`, `Year`, `Citation`, `Author`, `Author`, `Taxon`, `Rank`, `Taxon`, `Order`, `Author`, `Author`, `Taxon`, `Genus`, `Publisher class`, `Subspecies`, `Publication type`, `Year`, `A in B`, `Subgenus`, `Publication`, `Publisher class`, `Original combination`, `Publisher class`, `Pub ento`, `Citation`, `Publisher class`, `Publication type`, `Publication country`, `Genus`, `Order`, `Order`, `Rank`, `Taxon`, `Publisher class`, `Subgenus`, `Publisher class`, `Publisher class`, `Author`, `Taxon`, `Citation`, `Publication type`, `Publication type`, `A in B`, `Rank`, `Rank`, `A in B`, `Order`, `Subgenus`, `Original combination`, `Publisher class`, `Subspecies`, `Year`, `Publisher`, `Full name`, `Original combination`, `Genus`, `Author`, `Publication`, `Taxon`, `Subspecies`, `Name`, `Species`, `Full name`, `Citation`, `Journal`, `Year`, `Year`, `Year`, `Name`, `Journal`, `Original combination`, `Genus`, `Genus`, `Order`, `Number authors`, `Citation`, `Order`, `A in B`, `Number authors`, `Publication country`, `Genus`, `Taxon`, `Genus`, `Subgenus`, `Subspecies`, `Pub ento`, `Publication country`, `Taxon`, `Publisher`, `Author`, `Publication country`, `Subgenus`, `Subgenus`, `Publication country`, `A in B`, `Pub ento`, `Rank`, `Pub ento`, `Journal`, `Species`, `Full name` FROM insectdiscoveries WHERE"Publication" > "561";
Q6	Which species has most of its specimen discovered last?	1.52	SELECT `Order`, `Publication type`, `Publication type`, `Full name`, `Subspecies`, `Number authors`, `Rank`, `Taxon`, `Rank`, `Publication type`, `Author`, `Order`, `Full name`, `Order`, `Publication country` FROM insectdiscoveries WHERE ("Name" = "Valid") OR ((("Publication type" < "Article in Journal") OR ((("Number authors" < "2") OR ("Full name" <> "Brailovsky H.")) AND ("Subspecies" <> ""))) OR (("Subspecies" < "") AND (("Taxon" > "Ichneumonoidea") OR ("Publisher class" > "museum"))));
Q7	How many species discovered by Martin after 1990 in Australia have been published in the Acta Coleopterologica journal?	1.02	SELECT `Publisher`, `A in B`, `Publication`, `Number authors`, `Species`, `Species`, `Publication`, `Citation`, `Pub ento`, `Subspecies`, `Publication country`, `Publication country`, `Pub ento`, `Publication type`, `Pub ento`, `Citation`, `Publication`, `Taxon`, `Publication country`, `A in B`, `Publication country`, `Publisher`, `Subspecies`, `Genus`, `Pub ento`, `Publisher`, `Name`, `Publisher`, `Rank`, `Species`, `Genus`, `Author`, `Publication`, `Species`, `Publication`, `Publisher class`, `Journal`, `Publication type`, `Genus`, `A in B` FROM insectdiscoveries WHERE"Rank" <> "Species";
Q8	How many species in Heteroptera taxonomy have been discovered?	1.37	SELECT `Rank`, `Citation`, `Genus`, `Year`, `Full name`, `Publisher`, `Author`, `Publisher`, `Number authors`, `Publication type`, `Citation` FROM insectdiscoveries WHERE ("Original combination" <> "Y") OR ((("Rank" >= "Species") OR (("Pub ento" < "no") OR ("Publication type" = "Article in Journal"))) OR ("Genus" < "Ophiomyia"));