SIMULATION OF TRUST IN CLIENT – WEALTH MANAGEMENT ADVISOR RELATIONSHIPS

Terry Bossomaier[(a)], Russell K. Standish[(b)]
[(a)(b)]CRiCS Centre for Research in Complex Systems, Charles Sturt University.
[(b)]School of Mathematics and Statistics, University of New South Wales.
[(a)]tbossomaier@csu.edu.au, [(b)]hpcoder@hpcoders.com.au

ABSTRACT

This paper describes a two phase model for simulating trust amongst clients and their wealth management advisors. In phase one an artificial life model was used to assess the dynamics of trust. In phase two the model is extended to utlise real data from a corporate database of client information. The alife model highlighted needs for information not captured directly, requiring sophisticated inference techniques. Fuzzy logic is used to describe client behaviour with rules found through evolutionary optimisation. Analysis of mutual information between time series of clients investments is used to determine links between clients.

1. INTRODUCTION

Today's companies operate in a very complicated and sometimes turbulent environment. Unexpected changes in global resources such as the food crises in 2008 and the escalating oil price can dramatically change market positions. On the other hand the subprime mortgage meltdown has revealed just how complicated and fragile financial systems can be.

Decision support systems and mining of the vast quantities of consumer data in corporate data warehouses are valuable but their capacity to predict future requirements is often limited to narrow extrapolation from the past. More powerful scenario planning systems are needed which can explore new trends, such as, for example, the rapid switch of agricultural land from food to biofuel production, a powerful influence with little prior history.

Agent based models (ABMs) attempt to model people, companies and external forces and to go beyond simple extrapolation. From early minimalist models, some ABMs now incorporate many millions of agents, such as the large Epicast models for studying pandemics in the USA.

Until a few years ago Australian workers were in general locked in to the superannuation fund provided by their employer. But following deregulation, people were allowed to choose to which of many available funds they belonged. Inevitably this created a significant market for financial advisors. At first such services were poorly monitored and advice was not always sound. Worse there were a number of concerns about conflict of interest, where advisors were given hidden trailing commissions or promoted a fund owned by their parent organisation.

Thus trust in financial services became an important issue for banks and other providers to confront. This project focussed on building an ABM with two distinct goals

1. to examine the dynamics of trust, to look for phase transitions and indicators of declining trust. This was essentially an abstract artificial life model.

2. to build a realistic agent based model derived from real data.

The artificial life model (Bossomaier, Jarratt, Anver, Thompson & Cooper 2005) consisted of sets of clients and wealth management advisors (WMAs) and an artificial stock market. The advisors compete with one another to maximise their wealth, while the clients share trust information with one another.

The stock market simulates shares or share portfolios, which are owned by the client and are paid for with upfront consultancy fees, and funds, aggregates of shares from the same market, which pay commissions. The growth of the shares follows a deterministic equation with added noise. Advisors may purchase access to the growth

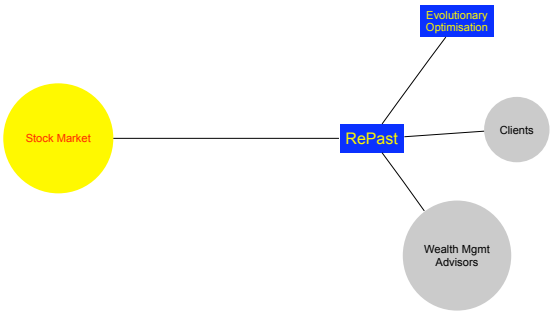

Figure 1: Architecture of the trust ALife model

parameters of this equation, thus trading off costs against better advice to clients.

Clients trust evolves according to their returns on investment, using insights from the neuroeconomics literature. Each client is connected to other clients on one of a number of different types of network, lattice, small world and scale-free. We present results for the wealth of clients and advisors for different networks and different research costs.

To move to the full scale model involves analysing a large corporate database of several million clients. The first stage in doing this requires analysis of the client network but this is not available from any fields in the database itself. Thus indirect methods are needed to infer the connectivity. A novel approach has been adopted which involves looking for common patterns in investment over time among clients.

2. THE ALIFE MODEL

The ALife model consists of several components as shown in figure 1.

Wealth Management Advisors invest money on behalf of their clients taking some profit along the way. They invest the clients' money in a mixture of shares and funds[1]. Shares return a percentage of the investment as a once off fee. Funds provide an ongoing trailing commission.

Clients split their wealth into a fraction invested with their WMA and leave the rest in the bank, the fraction being determined by their trust level. If their trust falls significantly below the trust of their neighbours,

[1]To make the simulation computationally tractable, portfolioss are relatively small and each share may be thought of as more an assest class than an individual equity.

The stock markert is not intended to represent the real stock market in any detail and the many diverse investment options it provides. It merely provides an investment framework.

Simulations of this model allowed the study of the evolution of trust under various conditions. One important issue is the nature of the client networks. If the clients are unconnected then their trust will go up and down with WMA performance and the WMA can trade off the additional investment he gets with increased trust against the potential loss of fees and commissions. But if the clients can talk to each other then the WMAs now have to outperform each other to avoid loss of clients to others who provide better net returns.

Much recent interest in networks has led to three commmon types in social systems: simple local connectivity, such as a lattice; small world networks in which additional long range connections are added (Watts 1999); and scale free networks characterised by a power law distribution in the connectivity of nodes (Barabási 2002). Different social networks will lead to different trust flow behaviours, hence modelling these networks is important. However, this information may not be readily available.

2.1. The Stock Market

There are very many stock market models aournd in the literature and this paper neither tries to improve on them, nor to evern select the best. Two factors drive the approach

- the model must be efficient in comuting resources
- it must have a natural transition to real financial instruments or products and must reflect investment in research activity into the value of different sorts of investment.

A simple model satisfying these requirements is used data from the NY Stock Exchange obtained via Yahoo finance (Yahoo 2008). An exponential fit to this data was lperturbed by an additional noise term using Brownian motion as in equatiion 1.

$$y = A_0 exp(a_1 * t + a_2 * c) \quad (1)$$

where A_0, a_1, a_2 are constants and c is a cummulative uniformly distirbuted random number in the range $[0 - 1]$.

The full bank model uses the financial instruments constructed by the bank and their variation over time as measured on the real stock market.

3. THE DATASETS

A large dataset of 456 million records describing 14 million customers over a five year period, and a smaller more detailed dataset of 42 million records describing the investment profiles of 1.5 million customers over a three year period were supplied to the project.

The larger dataset's records contained demographic details, as well as aggregate account balances over the three categories of cash, loans and investments. Since these account balances will depend in an intricate way upon the account product performance, customer income and expenditure, details which are not available in the dataset, it was decided to model the discrete events when existing customers enter or leave a particular account class. This includes (for example) existing cash or investment customers taking out a mortgage, or investors closing all their investments accounts, which is of particular interest to the trust project.

Customer behaviour is represented by values drawn from $\{-1, 0, 1\}$ where -1 indicates that all accounts of the class are closed, 0 means no change and 1 indicates that that customer has entered that class. Thus three timeseries are available in the dataset, representing the behaviour in the three classes. A fourth timeseries is generated from change in the number of dependents, perhaps due to the birth of child, or through marriage, or conversely through children growing up and leaving home.

From the smaller dataset, it was possible to establish the investment profiles of the 1.5 million investment customers. Whilst product performance was not available in this database, it was possible to match the internal product identifiers to published product information, and to download the relevant product performance from the bank's website. By defining a product's risk as the variance of its performance (historical volatility), one can estimate a customer's risk profile by taking the product balance weighted average of the product's risk. This will fluctuate over time as the product balances change (unless the customer is invested in a single product only), but if the customer performs active portfolio balancing, this will reasonably accurately reflect the customer's risk preference. The resulting timeseries has six independent variables (age, length of customer relationship, gender, marital status, deceased and number of dependents), and one dependent variable (risk).

4. MODELLING CLIENT BEHAVIOUR

Human behaviour may be represented in several ways, each extremely diverse (Fulcher 2008). Artificial neural networks (ANNs) are loosely linked to the structure and operation of the human brain but there are very many architectures and training or learning algorithms from which to choose. At the other extreme to ANNs are formal rule based systems. But representing human behaviour with rules is tricky, often requiring very large rule sets. Yet these two extremes ultimately have to converge to the same outcomes since both are capable of arbitrarily accurate representations.

Fuzzy logic falls somewhere in between. Its advantage in ABMs arises from the interdisciplinary nature of socio-economic modelling. Qualitative research outcomes and judgements from domain experts can be readily transcribed into fuzzy logic and its conception was in part motivated by the semi-quantitative style of much human thinking. Thus fuzzy logic is the methodology used herein.

4.1. Fuzzy Inference Systems

Fuzzy sets are sets whose elements have a degree of membership in the range $[0, 1]$. More precisely, a fuzzy set F is a pair $F = (A, m)$, where A is a set, and $m : A \rightarrow [0, 1]$ is the membership function. If $m(x) = 0$, then x not considered to be a member of the fuzzy set F, and if $m(x) = 1$ then x is considered fully included in the fuzzy set.

Fuzzy logic extends the notion of propositional logic to fuzzy sets with the fuzzy logic operators AND, OR and IS. Fuzzy logic rules are of the form:

$$\begin{aligned} &IF\ x_1\ IS\ I_{11} \\ &\quad AND\ (x_2\ IS\ I_{21}\ OR\ x_2\ IS\ I_{22}) \ldots \\ &\qquad THEN\ y\ IS\ O_1. \end{aligned} \tag{2}$$

There are a variety of *fuzzy inference systems* (FIS). The *Mamdani* type, used here , consists of a number of input variables x_i, whose ranges are partitioned into fuzzy sets I_{ij}, an output variable y whose range is partitioned into fuzzy sets O_j, and a set of fuzzy rules of the form (2). The FIS

takes a vector of input values, and outputs an inferred value $\hat{y}$.

The *matching degree* $w(x_1, x_2, ...x_n)$, or *weight*, for a rule is constructed from the antecedent, where the IS operator is replaced by the membership function, AND is replaced by a binary operator $\wedge$ called a *t-norm*, and OR is replaced by its t-conorm, $\vee$, where $a \vee b = 1 - (1 - a) \wedge (1 - b)$. Simple examples of t-norms are the minimum of the two argument and the product of its arguments, e.g. with the rule (2):

$$w(x_1, x_2, \ldots) = m_{I_{11}}(x_1) \wedge (m_{I_{21}}(x_2) \vee m_{I_{22}}(x_2)) \ldots \quad (3)$$

For the purposes of this work, $\wedge$ =min and $\vee$ =max were used to initially generate the FIS, but the evolutionary algorithm was allowed to mutate these to the product t-norm or the Łukasiewicz t-norm ($x \wedge y = \max(0, x + y - 1)$).

For converting the weight value w computed in (3) into the output value $\hat{y}$ of the inference system, one needs to aggregate the weights of all rules with the same consequent (the part following THEN in rule (2)), and then apply a *defuzzification operator*. Aggregation involves taking either the maximum of, or the sum of the weights of all rules with the same consequent. In this work, we used the sum aggregation rule, ie:

$$W_j = \sum_{\{r | c_r = \text{“}y \ IS \ O_j\text{”}\}} w_r(x) \quad (4)$$

where c_r is the consequent of the rth rule, w_r is the weight of the rth rule.

Finally, to produce an inferred output $\hat{y}$, one needs to defuzzify the aggregate weights W_j. A number of possible operators can be employed for this task, but the one we use herein is known as *area defuzzification*. Form new functions

$$\mu_j(x) = \begin{cases} m_{O_j}(x) & \text{if } m_{O_j}(x) < W_j \\ W_j & \text{otherwise} \end{cases}, \quad (5)$$

as shown in Figure 2. Then the inferred output is the centroid of the area underneath the sum of the μ_j curves:

$$\hat{y} = \frac{\int x \sum_j \mu_j(x) dx}{\int \sum_j \mu_j(x) dx} \quad (6)$$

4.1.1. Fuzzy Logic Source Code

FISPRO is an open source software package available from INRA. implementing fuzzy inference systems, allowing a wide variety of membership functions, different forms of weight computation, aggregation and defuzzification to be specified. It is implemented as a C++ library, with a Java interface provided through JNI that enables an interactive Java program that users can use to design fuzzy inference systems, and experiment with the inference engine.

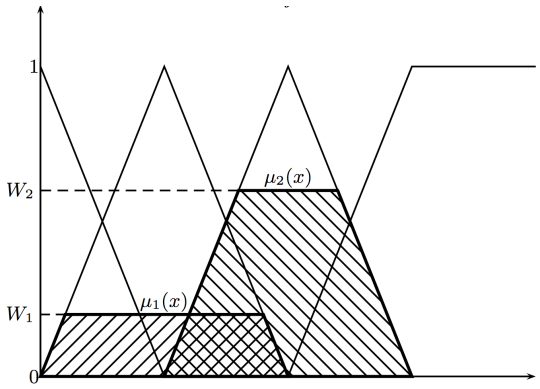

Figure 2: Membership functions

Additionally, FISPRO provides a number of functions for learning rules from training datasets. These were used to initially seed the evolutionary algorithms.

Various performance enhancements were added to FISPRO 3.0 and submitted to the FISPRO maintainers for inclusion into the next release of FISPRO.

4.2. Parametrising the FIS from Real World Data

Since around 1990, people have sought to combine the knowledge representation power of fuzzy inference systems with the learning power of evolutionary algorithms (EA), particularly genetic algorithms. Alander noted some 280 papers have been published on the topic by early 1996 (Alander 1997). A ten year survey by Cordón et al notes the different types of approaches taken to evolving fuzzy inference systems (Cordón, Gomide, Herrera, Homann & Magdalena 2004). Different aspects of the FIS are available to be evolved: the type of FIS (whether Mamdani, or Takagi-Sugeno), the t-norm used in the calculation of the rule weight (3), the rules and their consequents, the number and shapes of membership functions for the inputs and outputs, and the actual parameters of the membership functions. Most commonly, the parameters of the membership functions are evolved, or the rule base is

evolved. The evolutionary algorithm used is usually a genetic algorithm (parameters converted into a bitstring representation, which is evolved), although evolutionary strategies (working directly with floating point representations) are also deployed (eg (Cordón & Herrera 1999)), as we do here. When evolving the rule base, individuals of the EA may either be complete FISes (called the Pittsburgh approach) as used in this work, or individual rules (Michigan or Iterative approaches).

In the EvoNF framework (Abraham 2002) all of these aspects can be tuned, but in practice evolving all levels of this framework is computationally prohibitive. Normally, domain knowledge is used to constrain the optimisation search space. We had initially hoped to evolve just the membership function parameters, with the fuzzy rule base being given by domain knowledge. However, the domain knowledge turned out insufficient for the task, so we chose to inform the rule base from the data, by seeding the evolving population using the FPA algorithm (Glorennec 1996), and then further evolving the rule base.

4.3. Representation of the FIS

An evolutionary algorithm requires a representation of the solution, a sequence of evolutionary operators (genetic operators) to generate variation and a selection criterion for removing unsuccessful solutions from the pool.

In this work, we use a direct representation in the form of a list of the parameters for all the membership functions of the input and output fuzzy sets. We also vary which rules are active, and what their consequents are.

For computational efficiency (avoiding the extensive computation of exponentials or other such functions), we use piecewise linear trapezoidal membership functions. This includes triangular and semi-trapezoidal membership functions as a special case. Figure 3 shows the general form of the membership function, and defines the term *core*, where the membership function is 1, and the *support* where the membership function is greater than 0.

4.4. Evolutionary Operators

The operators we implemented were mutation, insertion (splitting), deletion (merging) and crossover, each controlled by a separate parameter.

In the case of mutation, with probability given

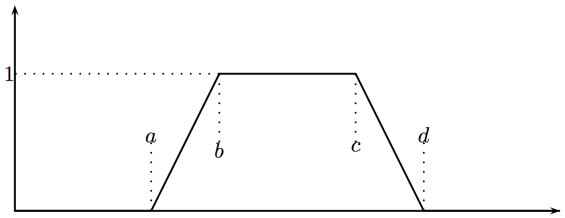

Figure 3: A piecewise linear *trapezoidal* membership function. The area between b and c is known as the *core* and the area between a and d the *support*. If $b = c$, the function is called *triangular*, if $a = b = -\infty$, or $c = d = \infty$ it is known as *infimum* or *supremum semi-trapezoidal* respectively

by the mutation probability parameter `mutConc` (0.01), a number was drawn from $\{0, \ldots, N_c\}$, where N_c is the number of fuzzy sets in the FIS output. If the number drawn was N_c, the rule was toggled between active and inactive, otherwise the rule's conclusion was set to the fuzzy set corresponding the number drawn. Similarly, with probability `mutConj` (0.01), the conjunction operator $\wedge$ was mutated between minimum, product and Łukasiewicz t-norms (which are the t-norms supported by FISPRO).

Mutating membership function parameters is a little more complex. We need to ensure that the fuzzy sets do not expand to engulf neighbouring fuzzy sets, so we require that (a) the support of a membership function does not overlap the core of its neighbours (b) that the the support to overlap the support of its neighbour to ensure complete coverage and (c) that $a \leq b \leq c \leq d$. We also limit the maximum variation to an evolutionary parameter `mutrange` (=0.1). In practice, this means that we calculate a range that maximally satisfies all those constraints, and chose a value randomly from that range. More precisely, if a_i, b_i, etc. represent the parameters of the ith membership function, the core is given by $[b_i, c_i]$ and the support by $[a_i, d_i]$. Let μ=`mutrange`. Then we form:

$$\begin{aligned} a_i^- &= \max\{a_i - \mu, c_{i-1}\} \\ a_i^+ &= \min\{a_i + \mu, b_i, d_{i-1}\} \\ a_i' &\in [a_i^-, a_i+] \\ b_i^- &= \max\{b_i - \mu, a_i'\} \\ b_i^+ &= \min\{b_i + \mu, c_i\} \\ b_i' &\in [b_i^-, b_i^+] \\ c_i^- &= \max\{c_i - \mu, b_i'\} \end{aligned} \tag{7}$$

$$
\begin{aligned}
c_i^+ &= \min\{c_i + \mu, d_i\} \\
c_i' &\in [c_i^-, c_i^+] \\
d_i^- &= \max\{d_i - \mu, c_i', a_{i+1}\} \\
d_i^+ &= \min\{d_i + \mu, b_{i+1}\} \\
d_i' &\in [d_i^-, d_i^+]
\end{aligned}
$$

The a_i', b_i', d_i' and d_i' become the new mutated value of the parameters. If any of the $x_i^- > x_i^+$, it is not possible to draw a new value for parameter x_i, so the parameters are left unchanged.

The insertion operator was implemented by replacing a membership function with two new membership functions that split the original core between them. If a, b, c, d are the original function's parameters, the new functions' parameters are:

$$
\begin{aligned}
a_1 &= a \\
b_1 &= c_1 = b \\
d_1 &= c \\
a_2 &= b \\
b_2 &= c_2 = c \\
d_2 &= d
\end{aligned}
\tag{8}
$$

The resulting membership functions are triangular, but need not remain that way after further mutation.

The deletion operator was implemented as replacing a pair of consecutive membership functions with a single merged function:

$$
\begin{aligned}
a &= a_1 \\
b &= b_1 \\
c &= c_2 \\
d &= d_2
\end{aligned}
\tag{9}
$$

These operators were applied with probability `split` (0.01) and `merge` (0.01), which are parameters of the evolutionary algorithm. Because these operations change the number of membership functions describing an input variable, we end up with oddities such as rule bases with 3 genders, so in practice we also specified a Boolean input array to indicate which inputs, and whether the output could have split/merge applied to them. Furthermore, changing the numbers of membership functions invalidates the ruleset. Rather than renumbering the ruleset (which would require resolving the issue of which of the two previous fuzzy sets maps to the new fuzzy set when a merge has happened), we took the approach of reapplying the FPA algorithm to regenerate a new rule base from scratch.

The final evolutionary operator was the crossover operator. This simply selected two parents at random from the pool, and crossed the membership functions for each input with 50% probability, and also crossed rules with matching antecedents with 50% probability (or performed an insertion of a parent 2 rule if its antecedent doesn't exist in parent 1. A later step in the evolutionary algorithm removes identical FISes from the pool.

4.5. Evolutionary Algorithm

A pool of FISes is seeded with a FIS generated using the *Fast Prototyping Algorithm* (FPA) (Glorennec 1996). The FIS pool is iterated over, with the various evolutionary operators described in the previous section applied according to the controlling probabilities. Once the number of FISes in the pool reached `maxPop` (10), selection is applied.

The primary measure of fitness of the FIS is *performance* P, which is the root mean square error of the FIS with respect to the training dataset σ. Some of the items in the training dataset may not match any of the rules well, particularly as the input fuzzy partitions evolve. If the maximum w computed according to equation (3) is less than m (`matchThresh`=0.01), then the item is dropped from the training set. Let $\sigma' = \{j \in \sigma : w(j) \geq m\}$, then performance is calculated from

$$
P = \sqrt{\frac{1}{|\sigma'|} \sum_{i \in \sigma'} (\hat{y}_i - y_i)^2}, \tag{10}
$$

and the coverage C as

$$
C = \frac{|\sigma'|}{|\sigma|}. \tag{11}
$$

The advantage to using this is that the FIS can report when its predictive ability is poor, and one can substitute an alternative inference rule (such as random selection from a probability table).

Using performance (eq (10) directly as a fitness function encourages the algorithm to find solutions that diminish coverage. By eliminating difficult to predict conclusions, P can be made arbitrarily small, even zero. If one sets `matchThresh` to zero (ensuring all of the training set is used), then P is dominated by the poorly performing rules, and the evolutionary algorithm has difficulty finding improvements.

An improvement to using P directly is to combine coverage, for instance as a ratio P/C

or as a linear combination $P + \alpha/C$. The former fitness function is particularly prone to finding a FIS that reduces coverage to the point that $P = 0$, which then dominates the evolutionary pool. The latter fitness function has the troublesome α parameter, and the algorithm stagnates once $P \leq \alpha$.

This is a problem of multi-objective optimisation (simultaneously minimising P at the same time as keeping C as high as possible). An alternative approach to multi-objective evolutionary algorithms is Pareto optimisation (Abbass 2006), whereby only Pareto-dominated FIS candidates are eliminated (those for which other FISes in the population are better at both performance and coverage). In practice, this algorithm worked the best of all.

4.6. Implementation and Results

The evolutionary algorithm was coded in C++ as a model running under *EcoLab* (Standish & Leow 2003), available from the *EcoLab* website.[2] OpenMP (OpenMP 2002) was used to parallelise the computation of performance and coverage, as well as repopulating the rule base after a a change in the number of fuzzy sets describing the inputs or output. The source code is the NCR.D5 release and relies on the modified version of FISPRO 3.0 (fispro.3.0.D10 see section 4.1.1.) Both are available from the same (*EcoLab*) website.

he smaller dataset had both advantages and disadvantages but on alance proved the most useful and all the results reported in this paper refer to he product risk timeseries obtained from this dataset. It was further downsized by sampling the customers with a requency 0.01 and of 0.001. This resulted in 285,660 records for the .01 sampling frequency, and 30,627 for the 0.001 sampling frequency. arious rule induction options available within FISPRO were tried, but nly the FPA option could handle such large datasets.

Using a single objective function, P/C could in exceptional cases give good coverage but tended to achieve coverage of only around coverage was around 80–85%.

With the matching threshold parameter was set to 0, coverage is 100% by definition. The evolutionary algorithm still improved the starting ruleset found by FPA, but it doesn't produce as good a solution as the multi-objective methods.

[2] http://ecolab.sourceforge.net

In the multi-objective case the a Pareto front of the best solutions occurs near the edge of the "cliff" where coverage drops off precipitously. The best solution at the end of the run has $P = 3.39 \times 10^{-11}$ and $C = 1.0$.

Extending the dataset to the 0.01 sample rate achieves a performance rate of 0.077 with complete coverage.

Thus optimising just on performance was not nearly as effective as optimising both P and C.

5. NETWORK ANALYSIS

The trust model requires understanding the social networks of clients. Although there is plenty of empirical evidence for small world (Watts 1999) or scale free connectivity (Barabási 2002), there are no fields in the data warehouse which capture the links. Furthermore there are no obvious proxies, e.g. children attending the same school, membership of the same sports clubs might all be harbingers of interactions, but no data of this kind is available. Hence a deeper inference system was needed. The solution is in looking at the time series of investments and determining how one client's investments correlates with another. Correlation between time series is a well understood metric but it can sometimes miss nonlinear interactions commpletely. Mutual information is a more powerful, although computationally more demanding technique (Cellucci, Albano & Rapp 2005).

5.1. Methods

The investment timeseries of the approx 180,000 customers investing in products with known performance data was computed using the technique described in section **??**. Since the absolute balances of the product investment do not carry meaningful information, and even relative balance (monthly investment divided by product balance) is distorted, the timeseries were converted to the range $\{-1, 0, 1\}$, representing withdrawal, no investment and investment respectively in a product. Mutual information was calculated in the usual way between these reduced investment timeseries $I(c, pr, t)$ for customer c, product pr at time t:

$$\begin{aligned} MI(c_1, c_2; \Delta) &= \sum_{x.y} p(x, y) \ln \frac{p(x, y)}{p_1(x) p_2(y)} \\ p(x, y) &\equiv p(I(c_1, pr, t) = x, \\ &\quad I(c_2, pr, t + \Delta) = y) \\ p_1(x) &\equiv p(I(c_1, pr, t) = x) \end{aligned}$$

$$p_2(y) \equiv p(I(c_2, pr, t+\Delta) = y)$$
$$(x,y) \in \{-1,0,1\}^2 | (x,y) \neq (0,0) \qquad (12)$$

Here the probability distributions were computed by histogramming over the 20 investment products and all timeseries points t (Jan 2002 – Dec 2003). The $(x,y) = (0,0)$ points were excluded because there were a lot of data points where no activity occurred, leading to spurious mutual information. Offsetting the timeseries by Δ in the range $[-3,+3]$ months allows for possible causality to be inferred. By maximising the mutual information over Δ,

$$MI(c_1,c_2) \max_{\Delta} MI(c_1,c_2;\Delta) \qquad (13)$$

the sign of Δ for which the mutual information is maximised induces a direction to the network link, pointing from c_1 to c_2 is Δ is positive, and vice versa.

Using the above technique of classifying customers passive investors are largely excluded by the $(x,y) = (0,0)$ exclusion, but regular customers will also tend to add spurious correlation between the timeseries, that is not due to causal influence. Therefore, the dataset was further reduced to just those customers classified as active investors. In the graphs presented with this report, the threshold was taken as a conservative 5%, meaning that only the 120,000 customers classified as active were considered. Further, this dataset was decimated to a final collection of approx 12,000 customers. A mutual information threshold of 1.05 was chosen to capture the most important links between customers. The networks generated in this fashion, illustrated in figure 4 (LGL 2008), typically have around 4000-7000 nodes.

The networks show substantial clustering, as well as restructuring as a function of time. Probably the next phase of research would be to apply standard network metrics such as degree centrality, clustering coefficients and categorising the type of degree distribution.

6. CONCLUSION

This paper identified fuzzy inference systems which could be parametrised against very large real world datasets. A multi-objective evolutionary algorithm significantly improves the performance and coverage of a fuzzy inference system trained on a large dataset over what was obtained by the fast prototype algorithm of FISPRO. This fuzzy system can now be used in the agent based model of client investment behaviour.

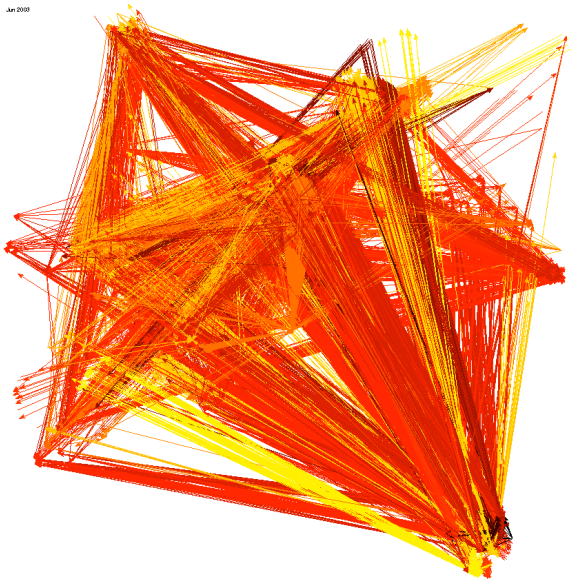

Figure 4: Illustrative client network. using large graph layout techniques to map related nodes close to one another. The client network shows a pronounced hub structure.

ACKNOWLEDGMENTS

This work was supported by grant LP0453657 from the Australian Research Council and a grant of computer time from the Australian Centre for Advanced Computing and Communications.

REFERENCES

Abbass, H. 2006. Pareto-optimal approaches to neuro-ensemble learning, *in* Y. Jin, ed., 'Multi-Objective Machine Learning', Vol. 16 of *Studies in Computational Intelligence*, Springer, Berlin, chapter 18, pp. 407–427.

Abraham, A. 2002. EvoNF: A framework for optimization of fuzzy inference systems using neural network learning and evolutionary computation, *in* 'Proceedings of the 2002 IEEE International Symposium on Intelligent Control', pp. 327–332. arXiv:cs/0405032.

Alander, J. T. 1997. An indexed bibliography of genetic algorithms with fuzzy logic, *in* W. Pedrycz, ed., 'Fuzzy Evolutionary Computation', Kluwer Academic, Boston, pp. 299–318.

Barabási, A.-L. 2002. *Linked*, Perseus, Massachusetts.

Bossomaier, T., Jarratt, D., Anver, M., Thompson, J. & Cooper, J. 2005. Optimisation of client trust by evolutionary learning of financial planning strategies in an agent based model, *in* 'Proc, IEEE Conf. on Evolutionary Computing', pp. 856–863.

Cellucci, C., Albano, A. & Rapp, P. 2005. Statistical validation of mutual information calculations: Comparison of alternative numerical algorithms, *Physical Review E* 71.

Cordón, O., Gomide, F., Herrera, F., Homann, F. & Magdalena, L. 2004. Ten years of genetic fuzzy systems: Current framework and new trends, *Fuzzy Sets and Systems* 141: 5–31.

Cordón, O. & Herrera, F. 1999. A two-stage evolutionary process for designing tsk fuzzy rule-based systems, *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybetics* 29: 703–715.

Fulcher, J. 2008. Computational intelligence: an introduction, *Studies in Computational Intelligence* 115: 1.

Glorennec, P.-Y. 1996. Quelques aspects analytiques des systèmes d'inférence floue, *Journal Européen des Systèmes automatisés* 30: 231–254.

LGL 2008. Large graph layout.
URL: *http://bioinformatics.icmb.utexas.edu/lgl/*

OpenMP 2002. *OpenMP C and C++ Application Program Interface*, OpenMP Architecture Review Board. Version 2.0.

Standish, R. K. & Leow, R. 2003. EcoLab: Agent based modeling for C++ programmers, *in* 'Proceedings SwarmFest 2003'. arXiv:cs.MA/0401026.

Watts, D. 1999. *Small Worlds*, Princeton University Press.

Yahoo 2008. Yahoo finance.
URL: *http://www.yahoo.com.au*