Inventory Optimisation with an Interval Type-2 Fuzzy Model

Simon Miller, Mario Gongora, Robert John

***Abstract*—The planning of resources within a supply chain can prove to be a deciding factor in the success or failure of an operation. This research continues the authors' previous work using an extended Interval Type-2 Fuzzy Logic supply chain model, with an Evolutionary Algorithm to search for good resource plans. A set of enhanced experiments is conducted to validate our novel approach with optimal configurations, and determine an appropriate Evolutionary Algorithm set up for the given problem.**

I. Introduction

Optimising inventory levels within a supply chain is an area of ongoing interest for supply chain managers. Planning the allocation of resources within a supply chain (SC) has been critical to the success of manufacturers, warehouses and retailers for many years. Mastering the flow of materials from their creation to the point of sale offers considerable advantages for those within a well managed SC. Poorly managed resources result in two main problems: stock outs and surplus stock. Stock outs occur when a node is unable to satisfy an order that has been placed on it. The consequences of this are lost sales, and potentially lost customers. Surplus stock is when goods are stored by a node from one period to the next, this results in added holding cost and the possibility of stock losing value as it becomes obsolete. Holding some surplus stock is advantageous however; *safety stock* can be used in the event of an unexpected increase in demand or to cover lost productivity.

Various degrees of uncertainty are present in the different data sources used in Supply Chain Management (SCM). This uncertainty is further amplified in demand forecasts by applying methods of analysis with (again) varying degrees of inherent uncertainty. Furthermore, other data that is often used in resource planning such as transportation and other costs, customer satisfaction information, etc. is also uncertain. Therefore, Fuzzy Logic (FL) and especially Type-2 Fuzzy Logic (T2FL) are particularly appropriate for this problem. While Type-1 FL (T1FL) has successfully been used many times for modelling SC operation (see Section II), T2FL has been shown to offer a better representation of uncertainty on a number of problems (e.g., [1] and [2]). Using a fuzzy model we can seek out good resource plans for a supply chain, though the search spaces involved are often very large even for a relatively simple problem. As such, it is not possible to find a resource plan using a exhaustive search, a more efficient search method needs to be selected. In [3] and [4] the authors use an Interval Type-2 Fuzzy Logic (IT2FL) model with an Evolutionary Algorithm (EA) and it is shown to work well.

Simon Miller, Mario Gongora and Robert John are with the Centre for Computational Intelligence, De Montfort University, Leicester, UK. (phone: +44 116 {2478408,2078491,2078226}; email: {smiller,mgongora,rij}@dmu.ac.uk).

In this paper we extend the work seen in [4]. An improved version of the model (see Section III) has been created, and an enhanced set of optimisation setups is used to find good resource plans. Each configuration is tested, and the performance reported. In Section II related work is discussed. Following this is an overview of T2FL and the model being used for the study, the EA, and a description of the test scenario. The experiments focus on the impact varying the parameters and operators has on the performance of the EA.

In Section II EAs and their applications in related work are discussed. Following this is an overview of T2FL and the model being used for the study, with a description of the test scenario. The experiments compare the performance of the optimisation setups on the test scenario problem.

II. Fuzzy resource model optimisation

There are a number of research projects that have explored the use of EAs with T1FL models. In this section we will look at examples of their use.

Evolutionary Algorithms (EAs) are a popular method of optimising fuzzy resource models. An EA is able to find good solutions to a problem while evaluating only a small fraction of the solution space. For an optimisation method to be practical, this is critical, as resource planning tends to involve an extremely large solution space.

In [5] a type-1 fuzzy system is used with a Genetic Algorithm (GA) to model a SC. The GA searches for a configuration that maximises profit, while meeting a target fillrate specified by the user; fuzzy sets are used to describe costs, returns, production capacities, storage capacities and forecasts. The proposed fuzzy method, and a crisp method are compared. The crisp system is unable to produce a feasible configuration if the actual demand is lower than the forecast. In contrast, the fuzzy model presented in [5] is robust and able to cope with fluctuation in demand and production capacity with little impact on profitability.

A similar approach is presented by Wang and Shu [6]. T1FL is used to represent customer demand, processing time and delivery reliability; a GA finds order-up-to levels. The system attempts to find the configuration that incurs the minimum cost. An optimism-pessimism index is set by the user and passed to the system. When optimistic, the model assumes the best case scenario for material response time. A pessimistic attitude presumes the worst. The results show that more pessimistic strategies increase the fill rate, reducing the sales lost through stock outs, and incur higher inventory cost as more stock is kept. More optimistic strategies result

in a drop in fill rate and an increase in sales loss, though inventory cost is also reduced.

Work by Sakawa and Mori [7] describes a method of scheduling jobs that have a type-1 fuzzy due date and processing time. A GA is used to find schedules. The GA takes into account the similarity of the solutions in a given population. When the initial individuals are created they have a similarity of 0.8 or less to ensure diversity. It is shown that ensuring diversity in this way leads to the GA finding an optimal solution on more occasions than a GA without a similarity measure. In testing, the GA is shown to work well, finding solutions with a large correlation between the processing time and the due date.

A. Type-2 Fuzzy Logic (T2FL)

In the examples given we have seen how T1FL has been used to tackle the resource planning problem. However type-1 fuzzy sets represent the fuzziness of the particular problem using a 'non-fuzzy' (or crisp) representation - a number in $[0, 1]$.

As [8] point out:

> "..it may seem problematical, if not paradoxical, that a representation of fuzziness is made using membership grades that are themselves precise real numbers."

This paradox leads us to consider the role of type-2 fuzzy sets as an alternative to the type-1 paradigm. Type-2 fuzzy sets [9] represent membership grades not as numbers in $[0, 1]$, but as type-1 fuzzy sets. Type-2 fuzzy sets have been widely used in a number of applications (see [10] and [11] for examples), and on a number of problems T2FL has been shown to outperform T1FL (e.g., [1] and [2]). Some work has been done regarding the use of Evolutionary Algorithms to optimise Type-2 Fuzzy sets (e.g., [12] and [13]) however, in this work we do not optimise the sets; we use an Interval Type-2 fuzzy model as the means to evaluate resource plans, as this work focuses on optimising the latter. Resource plans naturally take the format of a matrix of values detailing inventory by time period, node and product. Because of this, little adaptation is required to create a chromosome to represent a solution, that we can then operate on to facilitate evolution.

In previous work the authors have shown that Interval Type-2 Fuzzy Logic (IT2FL) [14] is an appropriate method of modelling a 2-tier and multi-echelon supply chain respectively ([15] and [3]). IT2FL has been used because it is computationally cheaper than general T2FL as it restricts the additional dimension, referred to as the secondary membership function, to only take the values 0 or 1. We believe that the extra degree of freedom will allow a better representation of the uncertain and vague nature of data used in SCM. In the next section the IT2FL model will be described.

III. Model

The model used for these experiments is an extension of one used in previous experiments [4]. Additions include:

- Functionality to specify minimum order quantity and unit of order quantity for each product at each node.
- Ability to set batch (transaction) cost by node.

The proposed model represents the interaction of nodes within a multi-echelon supply chain. Figure 1 provides an example of a typical supply chain. In each echelon there are one or more nodes that supply the subsequent echelon with one or more products, and receive stock from the preceding echelon. The first echelon receives goods from an external supplier which is assumed to have infinite capacity, the final echelon supplies the customer. Below the first echelon, capacity is limited by node and product.

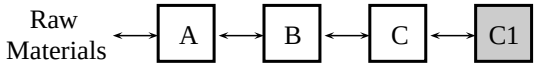

Fig. 1. A typical supply chain

Customer demand is provided by a fuzzy forecast which is given to the model at run-time. This forecast represents the demand placed upon the final echelon in the SC. Echelons above this can see their own demand by looking at the suggested inventory levels at the succeeding echelon, as they will be required to supply these items.

In order to use the model the following information must be provided:

- Number of echelons (not including the end customer)
- Number of nodes in each echelon
- Number of end customers
- Number of products
- Number of periods
- Service level required (as a percentage of orders filled completely)
- Capacities for each product at each node (amount that can be produced in one period)
- Lead time (in periods) for production/supply of each product at each node
- Minimum order and unit of order quantites for each product at each node
- Initial stock levels for each node
- Distance matrix containing distances between nodes in successive echelons
- Forecast of customer demand
- Suggested inventory levels
- Costs including:
  - Batch cost (by node)
  - Production cost (by product)
  - Transport cost (by product)
  - Holding cost (as a percentage of purchase price)
  - Purchase price (by product)

Using this information the model will calculate the cost of the given resource plan as shown in Equation 1 where $F(x)$ is the fitness of the solution x and:

- T = total number of periods to be evaluated
- I = total number of inventory locations
- b = batch cost
- p = production cost
- tr = transport cost
- h = holding cost
- s = stockout cost
- sr = service level
- tsr = target service level

$$F(x) = \left(\sum_{t=0}^{T} \sum_{i=0}^{I} (b_{i,t} + p_{i,t} + tr_{i,t} + h_{i,t} + s_{i,t}) \right) + \max\left((tsr - sr)\, 1000000, 0\right) \quad (1)$$

The total cost of a plan is made up of the following:

(a) The cost of setting up an order is called the *batch cost*. This represents the cost of administration, setting up any machines that are required, and picking the items for dispatch. There is a flat fee for each batch (that can vary by node) which is charged once at each warehouse for the production of a particular item for a particular customer.

(b) Each product is assigned an individual *production cost*. The total production cost for each batch is calculated by multiplying the number of items by their production cost.

(c) *Transport cost* is produced using a matrix of distances between nodes in neighbouring echelons, and a list of transport costs per km/mile. The product of the relevant cost and distance gives the overall transport cost for a batch of product.

(d) A *holding cost* is charged if a product is kept at a node for more than one period. The cost is calculated by taking a specified percentage of the purchase price of the goods held, for items carried over from one period to the next. The purpose of the charge is to represent the cost of storing items, the depreciating value of stock and the losses incurred by tying up capital in unsold stock.

(e) A *stock out cost* is charged for the shortfall of a product in a particular period. In this model we make the assumption that the end customer is always provided with an item. If it is not in the warehouse, it is obtained at purchase price from a competitor. The stock out cost is the sum of the value of items that had to be purchased. To discourage discovery of solutions that have a shortfall of stock, a stock out penalty is added by taking the product of the stock out cost and a multiplier provided by the user. Stock out penalty is applied to all but the final echelon, which supplies the end customer. In the final echelon, service level is used to determine how good a solution is. To administer a stock out penalty as well would be to penalise a solution twice for the same shortfall, leading to the EA being pressured to find solutions that satisfy 100% of customer demand, regardless of the service level required by the user.

(f) To discourage solutions that do not meet service level requirements, a *service penalty* is added to the cost of poor solutions in proportion to a solution's distance from the target service level. Service level is calculated by taking the percentage of customer demand that is completely satisfied. To measure satisfaction the fuzzy sets for the current stock level and the forecast are compared. An agreement index is calculated by looking at where the sets intersect, or is set to 1 if the inventory level exceeds the forecast. As stated before, stock out penalities are not applied here. It may appear that a more satisfactory solution would be to simply measure service level throughout the chain and not use stock out penalties. However in practice applying service penalty throughout the model results in the EA finding solutions in which the nodes within the chain placed little or no orders on each other, enabling solutions to achieve a good service level without satisfying a significant amount of customer demand. Stock outs need to be charged within the chain however, else the EA finds solutions in which only the final echelon before the customer supplies any product.

A. Interval Type-2 Fuzzy Logic

IT2FL [14] has been used to represent some of the values within the model. Other than the authors' previous work, examples in the literature (as discussed in Section II) have focused on the use of T1FL; we believe there exists an opportunity to exploit the extra degree of uncertainty provided by IT2FL in a model of this type.

As the model operates on IT2 fuzzy numbers, fuzzy arithmetic is used to calculate costs. This involves taking fuzzy sets, discretising them, performing the arithmetic operation, and then reconstructing the fuzzy set. In this model, fuzzy sets are represented using a series of α-cuts. Each set is an array of pairs of intervals. Each pair shows the area of values covered at a particular value of μ, the first interval is the left hand side of the set, and the second the right. Storing the sets in this way removes the need to discretise before fuzzy arithmetic is performed, and then reconstruct the result. Operations on the IT2 fuzzy sets are performed at the interval level, corresponding intervals (at the same μ) are taken from two sets, the operation performed and the result stored in a third fuzzy set.

Forecast demand, inventory level, transportation distances, transportation cost, stock out level, stock out cost, carry over and holding cost are represented by IT2 fuzzy numbers. For each of these values we can use the linguistic term *'about n'*, e.g., forecast demand of product 1 for customer 1 in period 1 may be *'about 200'*. Figure 2 shows how the set *'about 200'* may look with the α-cut representation used, where x is the scale of values being represented.

We have seen that values in the resource planning model are represented using IT2FL. Within the model this is useful as we can describe the uncertainty present in the supply

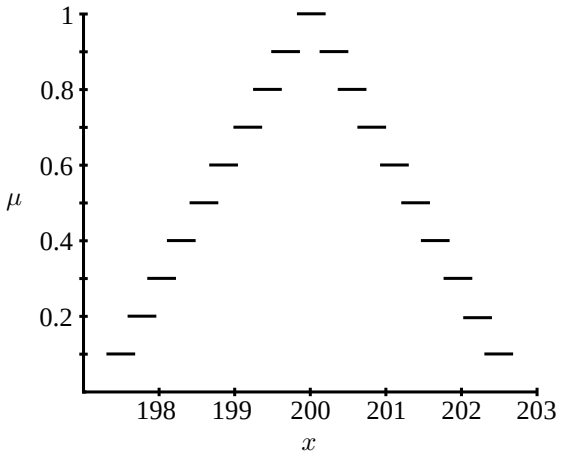

Fig. 2. Interval representation of IT2 fuzzy set *'about 200'*

chain. For the user however it is not explicit enough to state 'This resource plan will have a total cost of *about £1,000,000*' or 'Warehouse A should stock *about 300* of Product 1'. In order to produce an output that can be applied to a real-world supply chain, some of the IT2 fuzzy numbers need to be defuzzified. Defuzzification is the process of taking a fuzzy set (in this instance an IT2 fuzzy set) and deriving a single crisp value from it. To do this, the Karnik-Mendel method proposed in [16] is used. This is a widely used method that finds an interval representing the centroid of an Interval Type-2 fuzzy set. The interval can then be used to obtain a crisp number by finding its centre.

B. Optimisation

The purpose of these experiments is to find a set of parameters for the EA that result in the discovery of good resource plans. The base setup of the EA has been taken from the authors' previous related work [4] in which it was shown to work well. The EA has a population of 250 individuals and is executed for 500 generations, in all 125,000 solutions are evaluated out of a total possible search space of 1.424×10^{89} solutions. New generations consist of: 1% individuals produced with elitism, 10% copied individuals, 60% individuals created with crossover and 29% of individuals created using mutation. A description of the chromosome, operators and processes employed follows.

(a) The *chromosome* used to describe potential solutions is a 5 dimensional matrix of inventory levels. The dimensions are ordered as follows: echelon, period, source, destination and product. Each element of the matrix contains a value representing the number of items held in an echelon, in a time period, by a source node, for a destination node of a particular product.

(b) The *initial population* is randomly generated. Inventory levels are constrained so that they take into account the minimum order quantity, unit of order quantity and capacity of a node. This has been done to reflect the fact that in industry, products are usually manufactured in round quantities. If the model suggests that a warehouse should make 102 of product 1, this could lead to difficulties and extra expense. Limiting the valid inventory numbers also has the side effect of reducing the search space.

(c) *Fitness* is evaluated using the IT2FL model described. A solution's fitness is judged by the cost and service level achieved. For those solutions that meet the service level requirement specified by the user, the fitness is the cost of the solution; for those that don't a penalty is added to the cost as discussed in Section III. This encourages the EA to find cheap solutions, that meet a target service level.

(d) *Selection* is performed using a fitness ranking proportionate method similar to *roulette wheel selection*. First, all solutions in the population are ranked by fitness. They are then given a number of elements of an array in proportion to their fitness ranking. For example, if we have a population of 250 the fittest individual would be allocated 250 elements in the array, the second fittest 249 and so on. An element of the array is then selected at random, and the identification number of the individual it contains is used to retrieve a parent. This tombola style approach ensures that it is possible for any individual to be selected, while weighting in favour of those with greater fitness.

(e) *Crossover* is achieved with a uniform crossover. Two parents are selected using the method of selection described. Then, a new individual is created by selecting elements from each parent with a probability of 0.5. The probabilistic nature results in a variable crossover producing a diverse range of possible children from any two parents.

(f) *Mutation* takes a parent and then replaces a variable amount of elements with other valid values to create a child. The number of changes is decided in a similar way to the tombola selection used for selecting parents. This time the array is populated so that it is most likely that a small number of changes will be made, large numbers of changes are possible, but unlikely.

C. Test Scenario

To evaluate optimisation peformance a test case has been created. Table I describes the configuration of the supply chain, and Figure 3 shows the topology. Each customer requires 100 of each product in each period. It takes one period for a node to process and deliver an order.

Table II shows how each node is setup. The capacity refers to the amount of each product that a node can handle in one period. The minimum order quantity is the smallest order a node can place for a product; the order units are the increments in which orders are placed. For example, Node A has a capacity of 600, a minimum order size of 200 and orders in 100 item units. So valid orders for Node A are 0, 200, 300, 400, 500 and 600. The order sizes decrease as we move down the chain to reflect the fact that nodes nearer the source of the chain (e.g., warehouses and

manufacturers) tend to order in larger quantities than those near the end (e.g., retailers). It is possible to set capacity, minimum order quantity and order units by product within each node, however in this example it is assumed that it is the same for all products a node supplies.

Initially, each node has enough stock to meet one month's demand. Table III gives the operating costs of the supply chain. To calculate transport costs, the model needs to know the distance between nodes. Table IV shows the distances between nodes in successive echelons.

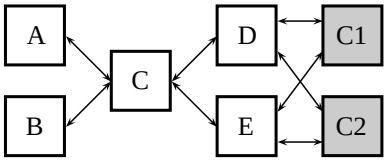

Fig. 3. Test scenario supply chain topology

TABLE I
TEST SCENARIO SUPPLY CHAIN SETUP

Products	2
Periods	6
Service level required	100%

TABLE II
NODE SETUP

Node	Batch cost	Capacity	Min. Order	Order units
A	100	600	200	100
B	150	600	200	100
C	100	300	100	50
D	50	150	50	10
E	100	150	50	10

TABLE III
SUPPLY CHAIN COSTS

	Product	
	1	**2**
Production cost	£2	£4
Transport cost	£0.50	£0.75
Purchase value	£4.50	£6
Holding cost	£0.45	£0.60
Stock out multiplier	5	

A service level of 100% has been chosen to simplify creation of an ideal solution for comparison. A resource plan was created that matched demand in each period, using nodes that are closest to one another. When put into the model, a cost of £21,865.98 was produced.

IV. RESULTS

Using the test scenario described, a series of experiments were conducted to test the effect of altering the size of the population, number of generations and the proportions of the operators used by the EA. All tests were run 10 times. In this section we will look at the results of each type of test in turn.

TABLE IV
SUPPLY CHAIN DISTANCES

	Dest. node	
Src. node	**1**	**2**
Echelon 1		
Node A	100	-
Node B	250	-
Echelon 2		
Node C	200	150
Echelon 3		
Node D	150	300
Node E	300	150

A. Population

In these experiments the base setup is used for all settings except the population, which is altered in each test. Table V and Figure 4 show the results.

TABLE V
MEAN COST OF SOLUTIONS BY POPULATION

Test	Pop.	Mean Fit.	Std. Dev.	Mean Time (secs)
1	50	38630.39	1902.95	169
2	100	34912.68	2166.84	343
3	150	33050.36	1551.41	515
4	200	31210.02	1934.65	703
5	250	30758.25	782.65	897
6	300	29499.98	1708.25	1105
7	350	30399.86	1825.44	1313
8	400	29772.10	1325.07	1546
9	450	29390.24	1154.76	1788
10	500	28964.21	961.79	2070

As we'd expect, for the most part the results show improvement as the population size increases. Fitness begins to

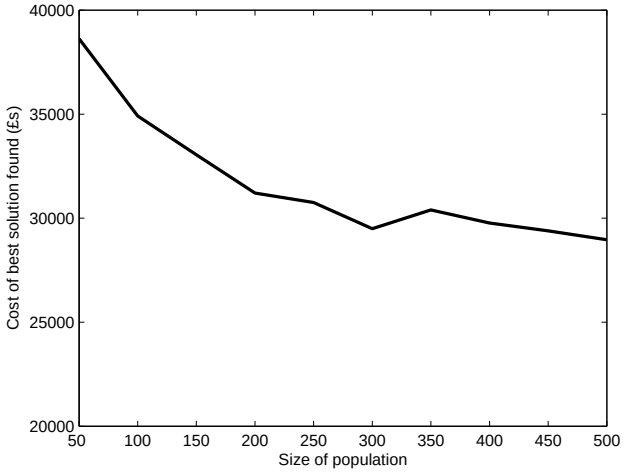

Fig. 4. Mean fitness by population size

level out when the population reaches 250-350 individuals, after this the improvements are less significant and the fitness actually decreases slightly between 300, 350 and 400 individuals. For the remaining tests we will keep the base level of 250 individuals, this seems an appropriate balance of time and performance and has the smallest deviation in solution quality.

B. Generations

To test the effect of altering the number of generations, the EA is run for 1000 generations and the fitness recorded after every 100. Table VI and Figure 5 contain the results.

TABLE VI
MEAN EVOLUTION AT 100 GENERATION INTERVALS

Gens.	Mean Fit.	Std. Dev.	Mean Time (secs)
100	49311.01	2506.48	191
200	39031.23	1616.81	369
300	34401.91	1250.73	541
400	31840.69	1145.29	708
500	30758.25	782.65	873
600	30221.16	890.31	1036
700	30039.20	825.63	1198
800	29897.99	839.88	1360
900	29569.97	717.38	1522
1000	29544.34	694.56	1683

As with population, we expect the fitness to improve as generations elapse, as elitism makes it impossible for the best solution in a generation to be worse than the best from the previous generation. In this case the fitness levels out at around 500-600 generations, which also have relatively low standard deviations. After this the improvements gained by continuing to run the EA are minimal.

C. Operators

In the final set of tests the base population and generations are used, and the proportions of the evolutionary operators are altered to see the effect. Each block of 5 tests fixes one operator at its base level, and alters the other two operators. The parameters are bound, they have to add up to 0.99. This approach allows us to cover the space of possible configurations in an ordered and logical manner, we will test all possible combinations at a reasonable level of discretisation. Table VII provides the results.

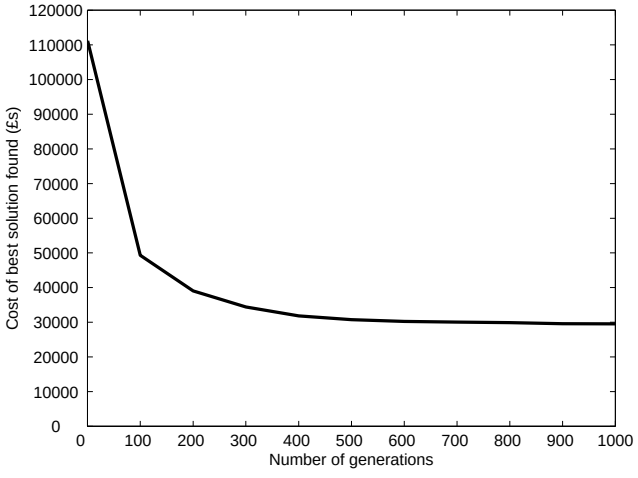

Fig. 5. Mean fitness over 1000 generations

TABLE VII
EFFECT OF ALTERING OPERATOR PROPORTIONS

Test	Copy	Cross.	Mut.	Mean Fit.	Std. Dev.	Mean Time (secs)
1	0.0	0.6	0.39	42971.79	2840.44	919
2	0.1	0.6	0.29	30758.25	782.65	861
3	0.2	0.6	0.19	32722.92	1516.10	841
4	0.3	0.6	0.09	36137.04	1388.93	849
5	0.38	0.6	0.01	46987.76	2500.28	876
6	0.1	0.0	0.89	61877.87	3085.90	953
7	0.1	0.2	0.69	61378.35	2765.94	953
8	0.1	0.4	0.49	53728.40	2871.07	942
9	0.1	0.8	0.09	33166.27	1520.47	857
10	0.1	0.88	0.01	42422.68	2583.42	871
11	0.0	0.7	0.29	29949.99	1146.56	886
12	0.2	0.5	0.29	31055.94	1832.60	860
13	0.4	0.3	0.29	33093.89	1680.54	862
14	0.6	0.1	0.29	36682.85	3128.32	886
15	0.7	0.0	0.29	45282.95	2527.94	891
16	0.0	0.8	0.19	30419.89	1617.58	856
17	0.0	0.9	0.09	33046.67	947.55	865

Looking at the results we can see that the copy operator has little effect on the fitness of solutions, one of those with no copy at all is the best configuration found. The reason for this is that the best solutions are already being copied into the next generation through elitism. Also, the selection method being used gives a significant advantage to the fittest individuals meaning that they are often used for crossover and mutation operations. Because of this, creating identical copies of individuals offers little advantage, other than good individuals that were not selected for crossover or mutation in this generation may be selected in the next generation if they are copied. The best solutions are those with a (relatively) small amount of mutation and a high proportion of crossover. Overall a setup with 1% elitism, no copy, 70% crossover and 29% mutation was shown to work best. The results of the third block of tests shows us that as crossover rises and mutation falls, fitness improves. With this in mind 2 extra tests were run, continuing the trend of increasing crossover and reducing mutation (tests 16 and 17).

Increasing the proportion of crossover further reduced the quality of the results gained. Knowing this, we can say that while a high proportion of crossover is beneficial, a significant amount of mutation (around 29%) is required to get the best results.

V. CONCLUSIONS

In this paper we have extended the work described in [4]. Using an improved model, and an enhanced set of

optimisation configurations it has been shown that by varying the setup of the EA, better resource planning solutions can be found. In general, increasing population and generations improves the results yielded. However, in both instances there comes a point where a significant amount of extra time brings about only a small improvement. The decision of how long to allow the algorithm to run should be made on a case-by-case basis. If time is not a factor, then it may be decided that it is worth running the algorithm for a long time to achieve the minimal improvements. When altering the proportions of the operators the best solutions were found with 1% elitism, no copy, 70% uniform crossover and 29% variable mutation. The EA favours the fittest solutions when selecting for crossover and mutation, this, coupled with elitism make the copy operator redundant. Crossover and mutation however have the ability to create new individuals increasing diversity in a population.

VI. Future Work

Although the results found using the more complex model are promising, there is still room for improvement. The best solution found in any test was £26,610.98, still 21.7% higher than the ideal solution created for comparison. Future research in this area could look further into the relationship between population and generations (e.g., very small population run for many generations and vice versa), the crossover, mutation, initial population generation and selection methods being used, and how solutions are represented within the chromosome. Extensions to the model will include looking at how alternative methods of representing type-2 fuzzy sets (e.g. geometric [17]) could be used to improve the model's description of the uncertainty in the supply chain, and increasing the complexity of the model so that it is possible for individual nodes to have different review periods as often happens in real-world supply chains.

Also, having seen that an interval type-2 fuzzy model can be used with an EA to find good resource plans, further research needs to be done directly comparing the performance of the interval type-2 model, with that of an equivalent type-1 model. Doing this will allow us to confirm or disprove our intuition that interval type-2 fuzzy logic offers a better representation of the uncertainty in the supply chain.

Acknowledgment

The research reported here has been funded by the Technology Strategy Board (Grant No. H0254E).

References

[1] H.A. Hagras. A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. *IEEE Transactions on Fuzzy Systems*, 12(4):524–539, August 2004.

[2] N. Karnik and J. Mendel. Applications of type-2 fuzzy logic systems to forecasting of time-series. *Information Sciences*, 120:89–111, 1999.

[3] S.M. Miller and R. John. An interval type-2 fuzzy multiple echelon supply chain model. *Knowledge-Based Systems*, 2010. In Press.

[4] S.M. Miller, M. Gongora, and V. Popova. Optimising interval type-2 fuzzy resource plans. In *Proceedings of Fourth International Workshop on Genetic and Evolutionary Fuzzy Systems (GEFS 2010)*, Mieres, Asturias, Spain, March 2010. In Press.

[5] R.A. Aliev, B. Fazlollahi, B.G. Guirimov, and R. R. Aliev. Fuzzy-genetic approach to aggregate production-distribution planning in supply chain management. *Information Sciences*, 177:4241–4255, 2007.

[6] J. Wang and Y-F. Shu. Fuzzy decision modelling for supply chain management. *Fuzzy Sets and Systems*, 150(1):107–127, 2005.

[7] M. Sakawa and T. Mori. An efficient genetic algorithm for job-shop scheduling problems with fuzzy processing time and fuzzy duedate. *Computers & Industrial Engineering*, 36:325–341, 1999.

[8] G.J. Klir and T.A. Folger. *Fuzzy Sets, Uncertainty and Information*. Prentice Hall, 1988.

[9] J.M. Mendel and R.I. John. Type-2 fuzzy sets made simple. *IEEE Transactions on Fuzzy Systems*, 10(2):117–127, April 2002.

[10] R.I. John and S. Coupland. Type-2 fuzzy logic a historical view. *IEEE Computational Intelligence Magazine*, 2(1):57–62, February 2007.

[11] J. M. Mendel. Advances in type-2 fuzzy sets and systems. *Information Sciences*, 177(1):84–110, January 2007.

[12] C. Wagner and H. Hagras. A genetic algorithm based architecture for evolving type-2 fuzzy logic controllers for real world autonomous mobile robots. In *Proceedings of the IEEE International Conference on Fuzzy Systems*, London, UK., July 2007.

[13] N.R. Cazarez-Castro, L.T. Aguilar, and O. Castillo. Genetic optimization of a type-2 fuzzy controller for output regulation of a servomechanism with backlash. In *Proceedings of 5th International Conference on Electrical Engineering, Computer Science and Automatic Control (CCE 2008)*, pages 268–273, Toluca, Mexico., November 2008.

[14] J.M. Mendel, R.I. John, and F. Liu. Interval type-2 fuzzy logic systems made simple. *IEEE Transactions on Fuzzy Systems*, 14(6):808–821, December 2006.

[15] S.M. Miller, V. Popova, R. John, and M. Gongora. An interval type-2 fuzzy distribution network. In *Proceedings of 2009 IFSA World Congress/EUSFLAT Conference*, pages 697–702, Lisbon, Portugal, July 2009.

[16] N. Karnik and J. Mendel. Centroid of a type-2 fuzzy set. *Information Sciences*, 132:195–220, 2001.

[17] S. Coupland and R. I. John. Geometric type-1 and type-2 fuzzy logic systems. *IEEE Transactions on Fuzzy Systems*, 15(1):3 – 15, February 2007.