A Multi-Objective Genetic Type-2 Fuzzy Logic Based System for Mobile Field Workforce Area Optimization

Andrew Starkey[1], Hani Hagras[1], Sid Shakya[2], Gilbert Owusu[2]

[1] The Computational Intelligence Centre, School of Computer Science and Electronic Engineering. University of Essex, Wivenhoe Park, Colchester CO4 3SQ, UK

[2] Business Modelling and Operational Transformation Practice. British Telecom, Adastral Park, Martlesham Heath, Ipswich, UK

Corrisponding Author

Andrew Starkey

University of Essex

School of Computer Science and Electronic Engineering,

Wivenhoe Park, Colchester CO4 3SQ, UK

Tel: 07581183308

Email: astark@essex.ac.uk

Abstract

In industries which employ large numbers of mobile field engineers (resources), there is a need to optimize the task allocation process. This particularly applies to utilities companies such as electricity, gas and water suppliers as well as telecommunications. The process of allocating tasks to engineers involves finding the optimum area for each engineer to operate within where the locations available to the engineers depends on the work area she/he is assigned to. This particular process is termed as Work Area Optimisation and it is a sub-domain of Workforce Optimization. The optimization of resource scheduling, specifically the work area in this instance, in large businesses can have a noticeable impact on business costs, revenues and customer satisfaction.

In previous attempts to tackle workforce optimization in real world scenarios, single objective optimization algorithms employing crisp logic were employed. The problem is that there are usually many objectives that need to be satisfied and hence multi-objective based optimization methods will be more suitable. Type-2 fuzzy logic systems could also be employed as they are able to handle the high level of uncertainties associated with the dynamic and changing real world workforce optimization and scheduling problems.

This paper presents a novel multi-objective genetic type-2 Fuzzy Logic based system for mobile field workforce area optimisation, which was employed in real world scheduling problems. This systems had to overcome challenges, like how working areas were constructed, how teams were generated for each new area and how to realistically evaluate the newly suggested working areas. These problems were overcome by a novel neighbourhood based clustering algorithm, sorting team members by skill, location and effect, and by creating an evaluation simulation that could accurately assess working areas by simulating one days worth of work, for each engineer in the working area, while taking into account uncertainties.

The results show strong improvements when the proposed system was applied to the work area optimization problem, compared to the heuristic or type-1 single objective optimization of the work area. Such optimization improvements of the working areas will result in better utilization of the mobile field workforce in utilities and telecommunications companies.

1. Introduction

For large companies with high numbers of mobile staff, efficiency can have a significant impact on many areas of the business, most importantly operation costs and revenue. This particularly applies to large utilities companies that provide services such as water, electricity or telecoms.

One area of efficiency that is key is the optimization of allocating engineers to available tasks. Assigning each engineer the right set of tasks can be crucial in increasing the amount of tasks completed satisfactorily across the organization. The increase in completed tasks can lead to the improvement of customer satisfaction, as customers have to wait less time for services to be delivered to them. This also has the potential to increase revenue as there is more capacity to take on new customers. Furthermore, the increased utilization of the engineers has the potential to lower costs, as this will mean using the existing set of engineers to execute the given tasks within their working hours, rather than paying more money for overtime expenses or hiring external workforces to complete the given tasks [17], [23].

A way in which the utilization of the engineers can be improved is by optimizing the area the engineers are assigned to. These areas, known as Working Areas (WAs) or Work Locations (WLs) [23] create the boundaries in which groups of engineers (teams) work within. These boundaries contain geographical areas and generate demand (tasks) for the engineers. However they also restrict the tasks that can be allocated to the engineers. If the WAs are not optimal, this will have a direct impact on the overall resource utilization.

In [17] the work revolves around a genetic fuzzy approach to assigning tasks to resources. However it does not look at the designs of the WAs the engineers work in. It does not generate new teams and it does not take into account factors such as travel or the imbalance of hours between the WAs. So this work greatly expands on the concept of workforce optimization but in a number of different ways, meaning the work noted in [17] could lead to sub-optimal solutions because it does not aim to optimize all the necessary factors that contribute to an engineer's utilization.

The overall structure and size of a WA can depend on the organizations management structure. As a number of WAs may be grouped together to form a region for the organization's higher level managers to oversee. This type of organization structure is a tree structure and is very common, as it is the same structure that is used in the military.

Figure 1 illustrates an example of how the United Kingdom (UK) may be split up into areas and the management structure in place to oversee the operations of these areas. In this example, the Director is responsible for 3 Regional Managers. The South UK Regional Manager has N number of Sub-Region Managers (indicated by the dashed line) they are responsible for. A Sub-Region Manager is responsible for N number of Branch Managers (indicated by the dashed line). Finally, the Branch Managers are responsible for a team of engineers which is divided into sub teams. These sub teams operate in their respective Working Areas, there can be between 1 and N number of WAs (indicated by the dashed line in Figure 1).

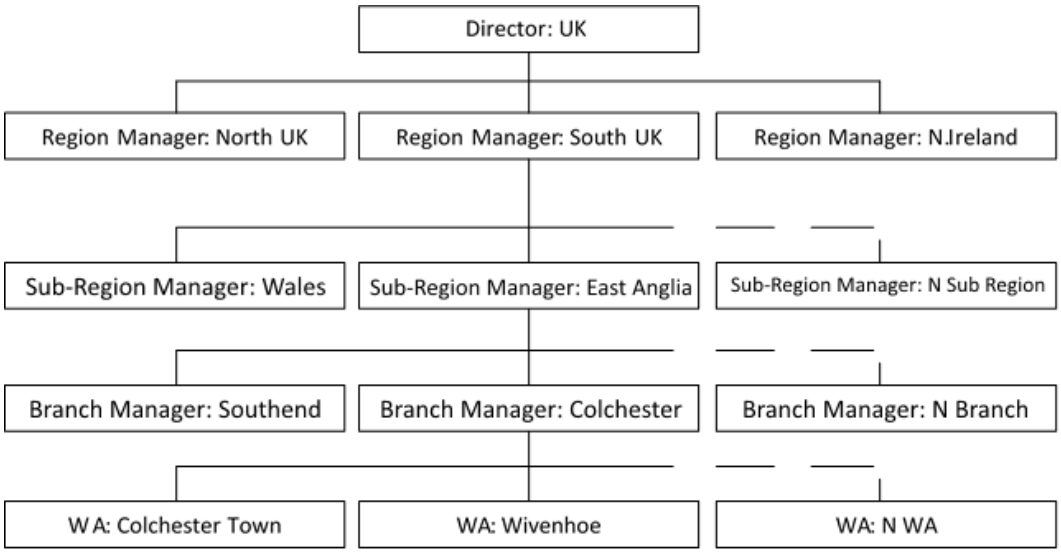


Figure 1. Management Tree Structure Example

The combination of the geographical working areas, the management structure and resource planning all contribute to the organization's efficiency and therefore needs to undergo an optimization process. This is to increase the efficiency with lower costs, as well as reducing travel costs or time dependent penalties and increasing the demand satisfaction. There are other secondary benefits involved with an optimized organizational structure, including the reduction of the organization's ecological impact (via less travel) and improved working conditions for engineers and managers.

Given the potential benefits of increasing an organization's efficiency, there have been a number of methodologies investigated to tackle the problem of optimizing workforce scheduling where heuristic techniques are widely applied. However a brute force or exhaustive search method will not find a good solution in good time. These problems are known as Combinatorial Optimization (CO) problems [27] and hence heuristic optimization techniques tend to lead to suboptimal solutions which consider only single objectives.

Algorithms designed to tackle CO problems usually aim for a metaheuristic approach. Metaheuristics are algorithms that attempt to find a solution that is good enough at solving the problem [4], [20]. This is most applicable in an organization with a large mobile field workforce.

A common metaheuristics approach to tackling these large scale and complex optimization problems is Genetic Algorithms (GAs) [3],[13],[25]. When using a GA there needs to be a way of testing how effective the created solution is at solving the problem. A good way to do this is to run the solution through a simulation. For WA optimization this would be a simulation of how effectively tasks would be completed given any setup of WAs. This would require calculating the paths engineers would take to complete tasks so their estimated travel distance and time can be calculated. This essentially links into the Travelling Salesman Problem (TSP).

The goal of the TSP is for a salesman to visit all cities in a given set only once and end up at the starting city. This has to be done in the shortest distance (minimum cost). However the number of potential paths increases exponentially with the increase in the number of cities the salesman has to visit [18]. Minimizing travel distance is not only a computationally complex problem, it is a real world constraint. The engineers in the real world will always take the shortest route where possible; this route is usually provided by their Global Positioning System (GPS) or familiarity with the WA. It is unlikely an engineer will visit a job location twice on two separate occasions. If one street has two jobs to be completed, the engineer will complete these jobs one after another. Instead of leaving the location after the first job and coming back to do the second.

Given the complexity and multiple objectives of these large scale optimization problems, traditional single objective genetic algorithms may not be appropriate. This is because they fail to take into account the conflicting nature some of the objectives may have. One way of solving this problem is to use Multi-Objective Genetic Algorithms (MOGAs).

In previous attempts to tackle workforce optimization in real world scenarios, single objective optimisation algorithms employing crisp logic were employed. So we can use these as a benchmark to compare against our proposed system. The advantage of this is that we can test both the multi-objective algorithm against the single objective algorithm. We can also test the impact type-1 and type-2 fuzzy logic has when compared against the crisp logic. We can then also compare how the multi-objective type-2 fuzzy logic system compares to the single objective crisp logic solution. As a result we can compare the impact of each proposed improvement over current methods.

One popular MOGA is the Non-Dominated Sort Genetic Algorithm II (NSGA-II) [2]. The benefit of using such an algorithm is that each objective specified is compared directly between solutions. Tackling a multi-objective problem using a single objective based algorithm

may hinder the exploration of the search to find the best solution [22] and thus lead to a sub-optimal solution.

In business processes there are many objectives to consider. Many of which are conflicting objectives, but multi-objective algorithms can help to find suitable solutions when applied business processes [26]. MOGAs have been shown to improve on the results of Single objective GAs in scheduling problems [28]. This is beneficial to WA optimization because scheduling forms a large part of the evaluation process. This means other single objective only type algorithms, such as Bacterial Foraging Algorithm (BFA) and Differential Evolution (DE) cannot be used, as for a fair assessment their needs to be a comparison between the single and multi-objective versions of the same algorithm. This effectively leaves us with a GA and MOGA like NSGA-II or Partial Swarm Optimization (PSO) and Multi-Objective Particle Swarm Optimization (MOPSO).

MOPSO can find a solution quicker and perform relatively well however NSGA-II can find more optimal solutions because it can maintain a good variety of solutions when compared to MOPSO [24]. The variety of solutions is especially important for the proposed problem as it is a higher-order multi-objective problem in a real world environment, so the diversity of solutions and reliability can explore more of the search space.

Higher-order multi-objective problems are not explored as much as bi-objective problems. For example in [22] a variant of the MOPSO is compared against NSGA-II using the same predefined problems (SCH, FON, ZDT1, ZDT2, ZDT3 and ZDT6) as [2] used to test the performance of NSGA-II against other algorithms. However due to the proposed problem being a higher-order multi-objective problem and is restricted by real-world constraints, we wouldn't be able to confidently say that our proposed problem would gain the same benefits over NSGA-II as described in [22] by using a MOPSO.

Another area that aims to improve the solutions generated for the WA optimization problem is fuzzy logic. The reason fuzzy logic should be applied to WA optimization is the potentially high level of randomness and uncertainty which face the problem of WA optimization in changing and dynamic environments, for example:

- Uncertainties in the data used for optimization, as the data used is collected or estimated from real world data captured from sensors which are subject to noise and impression.
- Uncertainties on the available skills per day due to engineers falling sick or going on holiday.
- Estimated travel times and distances. The travel time between jobs is estimated based on the time given by a route planner. The times and distances given may not reflect road works, traffic collisions, toll roads or rush-hour traffic.

- Estimated job completion times. The average time to complete a job of a particular type is used to estimate the job completion time. However each engineer has their own rate of efficiency that is not used.

There are a number of examples where real world problems use genetic algorithms to solve the issues presented [21],[31],[32]. There are also examples of multi-objective GAs being used to solve real world problems [1],[30] and there are also examples where multi-objective genetic algorithms have been combined with fuzzy logic to improve the results associated with a non real-world application in a normal TSP [29]. However none of the existing solutions which employed multi objective GA for real world WA optimization employed type-2 fuzzy logic.

The introduction of fuzzy logic can potentially improve the optimization because fuzzy logic can be used to handle the real world uncertainties. However type-1 Fuzzy Logic Systems (FLSs) cannot fully handle the high level of uncertainties associated with the real world dynamic and changing environments as the type-1 FLS employ the crisp and precise type-1 fuzzy sets. Type-2 FLSs can handle high uncertainty levels as they employ type-2 fuzzy sets which provide through their Footprint of Uncertainty (FOU) and third dimension additional degrees of freedom to enable handling higher uncertainty levels. There have been notable examples of type-2 FLSs outperforming the results of type-1 fuzzy logic system [6], [8], [14].

This paper presents a novel multi-objective genetic type-2 fuzzy logic based system for mobile field workforce area optimization, which was employed in real world scheduling problems. The reason for the creation of this type of system was that to accurately take into account all the business objectives of resource optimisation, a multi-objective approach is needed. This is because as a single objective approach will lead to suboptimal solutions as it cannot take into account each objective separately. As fuzzy logic performs well at handling real-world uncertainties, this was also integrated to improve the results. Given these constraints the strongest option was to build a multi-objective type-2 fuzzy logic system.

This systems had to overcome challenges, like how working areas were constructed, how teams were generated for each new area and how to realistically evaluate the newly suggested working areas. These problems were overcome by a novel neighbourhood based clustering algorithm, sorting team members by skill, location and effect, and by creating an evaluation simulation that could accurately assess working areas by simulating one days worth of work, for each engineer in the working area, while taking into account uncertainties.

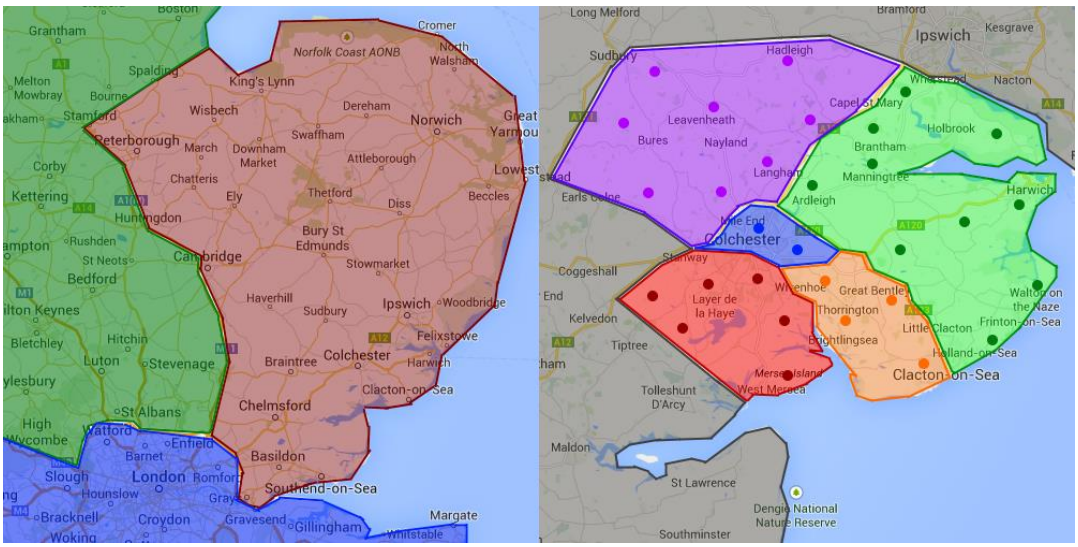
The results show significant improvements when the proposed system was applied to the work area optimisation problem as compared to the heuristic or type-1 single objective optimisation of the work area. Such optimisation improvements of the working areas will result in improving the utilization of the workforce.

Section 2 will present more details on the problem description of WA optimisation. Section 3 will provide high level overview on type-2 FLSs. Section 4 will present a high level overview on the employed MOGA NSGAII. Section 5 will present the proposed multi-objective genetic type-2 fuzzy logic based system for mobile field workforce area optimization. Section 6 will present the experiments and results while Section 7 will present the conclusions and future work.

2. Overview of the Work Area Optimization problem

2.1 Overview on Work Areas

The current problem concerns the optimization of the working areas in which the engineers are constrained. Large organizations usually split up the territory they work within into regions and sub regions for the purpose of management. At the lowest level, the areas are known as Working Areas (WAs) which are made up of a collection of Service Delivery Points (SDPs). These SDPs serve domestic and commercial properties by connecting these properties to services such as electricity, gas, water or telecoms, depending on the service the organization provides. These SDPs generate demand for services and create tasks for engineers. Figure 2a illustrates how the UK might be subdivided into regions.



(a) (b)

Figure 2. a) Regional Areas. b) WAs within a Sub-Region

Figure 2b shows a sub region, which might be operated by a branch manager. This area is divided into 5 WAs with the groups of SDPs shown for each. It's the grouping of these SDPs that needs to change when finding the most optimal WAs. Note that high density areas are smaller than rural areas. This is because the WAs need to be balanced in terms of work where high density urban

areas provide more work per SDP than low density rural areas. The goal is to have the engineers service as many tasks as possible at the lowest cost.

2.2 Objectives and Constraints

The WA optimization process has a number of objectives which need to be satisfied as follows:

- Maximize Coverage: Coverage is the amount of tasks that are estimated to be completed. This is measured in hours. In Equation (1) this is represented at the sum total of all engineers completed work (E_{cw}).

$$\sum E_{cw} \tag{1}$$

- Minimize Travel: Minimizing traveling distance increases the amount of available time for each engineer and also decreases costs. However minimizing travel directly conflicts with maximizing coverage. This is because an engineer (in the majority of cases) will be required to travel to each task. As coverage increases, travel also increases. In Equation (2) this is represented as the sum total of all engineers travel distance (E_{td}) divided by the sum total of engineers (E) as this is represented as average travel per engineer.

$$\sum E_{td} / \sum E \tag{2}$$

- Maximize Utilization: Utilization is the percentage of time an engineer is completing tasks. Unutilized time is when the engineer is idle or travelling. In Equation (3) this is the sum total of engineer completed work (E_{cw}) divided by engineer available time (E_{at}), this is then divided by the sum total number of engineers (E) to get the average utilization.

$$(\sum E_{cw} / E_{at}) / \sum E \tag{3}$$

- Maximize Area Balancing: WAs should be evenly balanced with the amount of work they contain. This means there will be smaller WAs for urban areas and larger WAs for rural areas. Balancing is measured in hours and is represented in Equation (4). It is the difference between the largest (WA_L) and smallest (WA_S) WAs in terms of hours of work.

$$WA_L - WA_S \tag{4}$$

There are a number of constraints that need to be looked at and included in the optimization. For example, all of the engineers won't all be working at all times (as some of them might fall sick, have holidays or day offs), so there is a degree of workforce shrinkage that needs to be taken

into account. Of the engineers that remain, they can only be assigned tasks that they are qualified to complete. Of these tasks, each engineer has preferred tasks that they work on. Taking this into account can help improve the average time taken to complete the tasks.

Another constraint is that each engineer is limited by the amount of work they can do each day (travel time has to be included in this). In addition, each team has to be equal in size and WAs should not cross large rivers or other geographical obstacles.

3. Overview on Type-2 Fuzzy Logic Systems

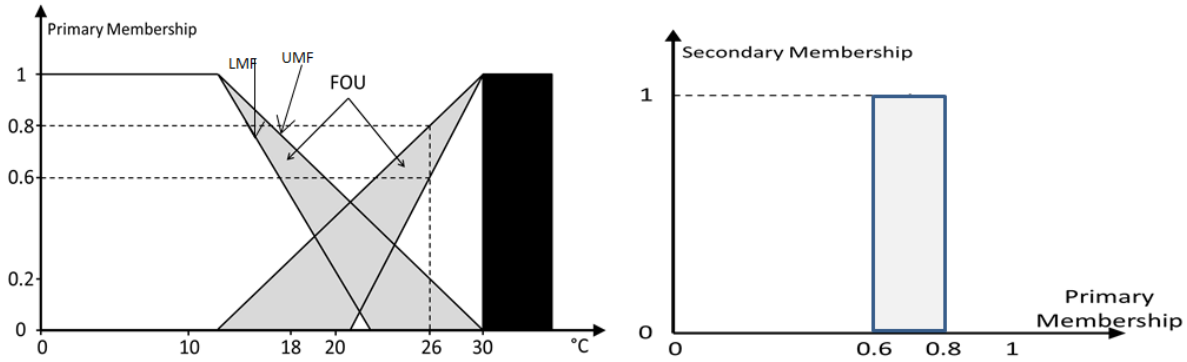
Fuzzy Logic Systems (FLSs) have been credited with providing white box transparent models which can handle the uncertainty and imprecision. However, the vast majority of the FLSs were based on type-1 fuzzy logic systems which cannot fully handle or accommodate the uncertainties associated with changing and dynamic environments. Type-1 fuzzy sets handles the uncertainties associated with the FLS inputs and outputs by using precise and crisp membership functions [12]. Once the type-1 membership functions have been chosen, all the uncertainty disappears, because type-1 membership functions are totally precise [6], [12].

The uncertainties associated with real world environments cause problems in determining the exact and precise antecedents and consequents membership functions during the FLS design. Moreover, the designed type-1 fuzzy sets can be sub-optimal for given environment conditions. However due to the change in the individual engineer circumstances and the uncertainties present in the surrounding environments, the chosen type-1 fuzzy sets might not be appropriate anymore. This can cause degradation in the FLS performance and we might end up wasting time in frequently redesigning or tuning the type-1 FLS so that it can deal with the various uncertainties faced. Type-2 FLSs which employ type-2 fuzzy sets can handle such high levels of uncertainties to give very good performances.

A type-2 fuzzy set is characterized by a fuzzy membership function, i.e. the membership value (or membership grade) for each element of this set is a fuzzy set in [0,1], unlike a type-1 fuzzy set where the membership grade is a crisp number in [0,1] [12]. The membership functions of type-2 fuzzy sets are three dimensional and include a Footprint of Uncertainty (FOU), it is the new third-dimension of type-2 fuzzy sets and the footprint of uncertainty that provide additional degrees of freedom that make it possible to directly model and handle uncertainties [6], [12]. As shown in Figure 3a, the Interval Type-2 (IT2) fuzzy set $\tilde{A}$ can be represented in terms of the Upper Membership Function (UMF) (denoted by $\overline{\mu}_{\tilde{A}}(x)$, $\forall\, x \in X$) and the Lower Membership Function (LMF) (denoted by $\underline{\mu}_{\tilde{A}}(x)$, $\forall\, x \in X$) as follows:

$$\tilde{A} = \int_{x \in X} \left[\int_{u \in \left[\underline{\mu}_{\tilde{A}}(x), \overline{\mu}_{\tilde{A}}(x) \right]} 1/u \right] / x \qquad (5)$$

The UMF and LMF are bounds for the *FOU*($\tilde{A}$) of an IT2 fuzzy set $\tilde{A}$. As shown in Figure 3b, in an IT2 fuzzy set the secondary membership function is equal to 1 for all the points in the primary membership for $\forall\, x \in X$.



(a) (b)

Figure 3. a) A interval type-2 fuzzy set- primary membership function. b) An interval type-2 fuzzy set secondary MF at a specific point *x'*.

Figure 4 shows an overview on the type-2 FLS where the crisp inputs are fuzzified to input type-2 fuzzy sets which are fed to the inference engine which maps the input Type-2 Fuzzy sets to output Type-2 fuzzy sets using the rule base. The output set is then processed by the type-reducer in the type reduction section which generates a type-1 output set. In this paper we use the Centre of Sets type-reduction, shown in Equation (6), as it has a reasonable computational complexity that lies between the computationally expensive centroid type-reduction and the simple height and modified height type-reductions which have problems when only one rule fires [12].

$$Y_{cos}(x)_k = [ylk, yrk] = \int_{y_k^1 \in [y_{lk}^1, y_{rk}^1]} \cdots \int_{y_k^M \in [y_{lk}^M, y_{rk}^M]} \int_{f^1 \in \left[\underline{f}^1, \overline{f}^1\right]} \cdots \int_{f^M \in \left[\underline{f}^M, \overline{f}^M\right]} 1 \Big/ \frac{\sum_{i=1}^{M} f^i y_k^i}{\sum_{i=1}^{M} f^i}$$

(6)

After the type-reduction process, the type-reduced sets are defuzzified (by taking the average of the type-reduced set) so as to obtain crisp outputs. This is shown in figure 4. More information regarding the interval type-2 FLS and its applications can be found in [7],[9], [10], [15], [16].

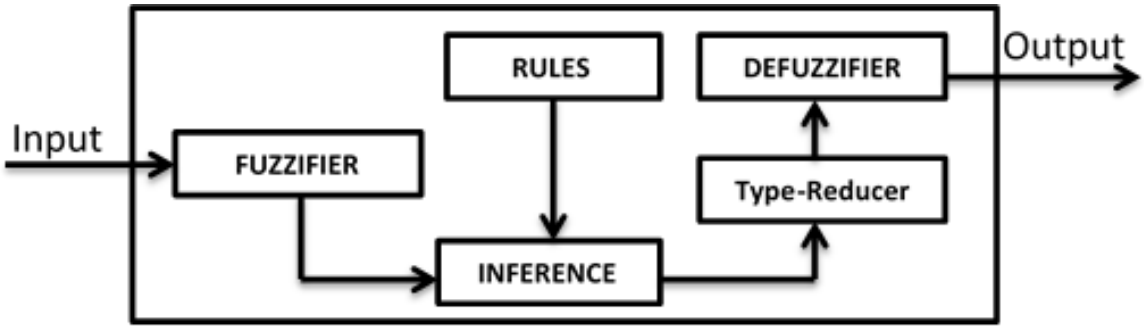


Figure 4. Type 2 FLS [16].

4. Overview on Single and Multi Objective Genetic Algorithms

4.1 Single Objective GAs

Genetic Algorithms (GAs) are based on Charles Darwin's theory of natural selection and evolution. In GAs, over time a population, or species, will adapt to its environment. This adaptation takes place through the idea of survival of the fittest. The individuals in the population that have characteristics most suited to its environment are the individuals most likely to survive. Thus these characteristics are passed on to the next generation of the species. The individuals less suited to the environment don't survive so these characteristics are lost in the next generation.

In genetics each individual has a chromosome (or multiple chromosomes) where a chromosome is a collection of genes. These genes are what determine the characteristics of the individual. When the next generation of the species is created, genes from two parent individuals will be combined to determine the characteristics of the child [11]. Figure 5 shows the process of a standard GA. The first step is to initialise the population. This is done by creating N solutions and randomly assigning the genes of each solution.

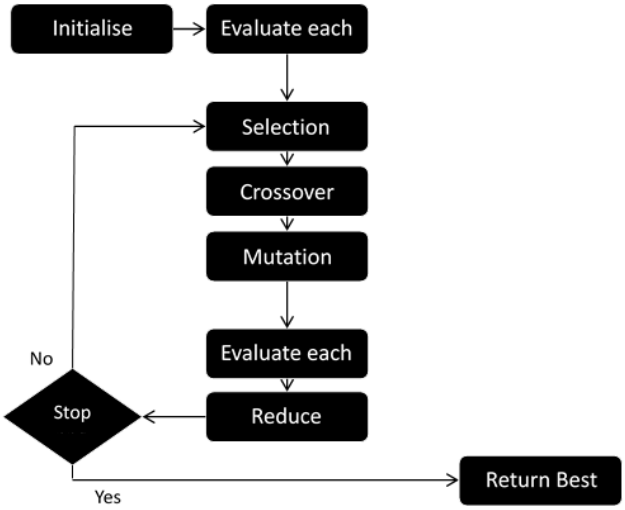


Figure 5. GA flow

These genes are then evaluated to see how suited to the environment they are. Each solution (individual) is then given a score to represent the solution fitness. Once each solution has been evaluated. The evolution process can begin. The first step in this is the selection of solutions to crossover. There are a number of selection operators out there such as tournament or roulette [19]. In tournament selection a subset of solutions from the population are chosen. Then the solution with the highest fitness will be chosen as the first parent. The process is repeated to find the second parent.

Once two parents have been chosen they will crossover their genes using a crossover operator (1 point, 2 point, uniform) [19]. Crossover will generate 2 child solutions that will be added to the new population set. Every so often one of the genes in a child solution will randomly change, this is known as mutation. Once enough children have been generated and the new population set is the same size as the old population set, the old population will die off and the new population will go back to the fitness evaluation stage.

The stopping criteria decides when the GA process should stop. This can be done by setting the maximum number of generations or waiting until convergence happens. Convergence is when all solutions in the population are the same and have 'converged' on the same point in the search space. If the criteria is met the best (most fit) solution will be returned, else the GA will loop back round for another generation.

Whenever one of the generated solutions is being evaluated by the genetic algorithm it uses a fitness function. If there is one objective to be optimized in the GA, then the fitness function will reflect the objective to be optimized. However if there is more than one objective, then the objective values need to be combined using a function to give a single fitness value. If the objectives are complementary or do not have any correlation then the objective function will be sufficient in some cases. In cases where the objectives conflict then the use of a fitness function starts to show its weakness. Some examples of conflicting objectives include:

- Minimizing Cost while Maximizing production
- Minimizing Carbon Dioxide (CO^2) emissions while Maximizing Transport Capacity
- Maximizing Customer Satisfaction while Minimizing Staff

The problem with these conflicting objectives is that neither can be 100% satisfied without causing significant damage to the other objective. For example we can easily minimize costs to 0, however production would also be 0. This situation is not acceptable, especially in real world problems.

The following is an example of how using a single objective GA to solve multiple objectives can be ineffective at tackling all objective. Imagine a problem which has three

objectives whose current values (original solution) are A:5, B:5, C:10. B and C are conflicting objectives where more B gives more C with a linear relationship. A and B are maximization objectives and C is a minimization objective. The fitness function would be written as follows:

$$Fitness = \frac{A.B}{C} = 2.50 \quad (6)$$

Here are some possible solutions that could come from the single objective GA with this fitness function:

Table 1 – Possible Solutions

Possible Solution	*A Value*	*B Value*	*C Value*	*Fitness*
1	6.00	8.00	12.00	4.00
2	4.00	4.00	6.00	2.67
3	4.00	2.00	3.00	2.67
4	9.00	10.00	15.00	6.00

In Table 1, Solutions 1 and 4 have worse C than the original solution (as it is higher and this is a minimization objective) but the solutions are considered better in terms of their fitness value. Solutions 2 and 3 have worse A and B than the original but the solutions are considered better. All of the solutions presented above are deemed better than the original solution fitness value with the fitness function, however none of the solutions optimize in all of the objectives when compared with the original. This example illustrates why problems with multiple objectives should use a multi-objective GA.

4.2 Multi-Objective Genetic Algorithms

Multi-Objective Genetic Algorithms (MOGAs) compare the results of each objective between solutions. This means that the value for the first objective for one solution can be compared to the value for the first objective of another solution explicitly. In this way it's clear which solution is stronger in this objective, rather than having all values amalgamated into one value and not really knowing how the objective values compare.

Given that each objective is compared between solutions there needs to be a method of deciding if one solution is better than another. One way of doing this is by 'Domination' as is done in the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) [2]. Domination determines if one solution dominates another by setting out conditions. These conditions are as follows (to determine if solution A dominates solution B):

1. Solution A has no objective value that is worse than the respective objective value in B.
2. Solution A has at least one objective value that is better than the respective objective value in B.

If both of these conditions are met it would be determined that A dominates B, meaning solution A is the better solution. If each solution is compared with every other solution in the population in the same way, the domination count can be calculated. The domination count is the number of solutions that dominate the current solution.

Once the domination count has been calculated a simple sorting algorithm can be used to order the solutions from best to worst. The solutions with a domination count of 0 (no solutions are deemed better) are grouped together to form the Pareto front. This is the set of solutions that are all deemed to be the best and selecting any of these solutions will be the most suitable for the problem. All other fronts (sets) are made up of the other solutions and are grouped based on their domination count.

Figure 6 illustrates what the fronts may look like in a MOGA with two minimization objectives. Figure 6 also shows an infeasible point which is a point that is not possible given the constraints of the objectives. A benefit of having a Pareto front set of solutions is that it gives the user a choice of solutions. Given the nature of multi-objective problems different solutions with different strengths may be suitable at different times. The user can then select the solution they find most appropriate for the situation, knowing there is no better solution available and the other objectives (although they may not be as good as other solutions on the Pareto front) are the best they can be given the current priority and constraints.

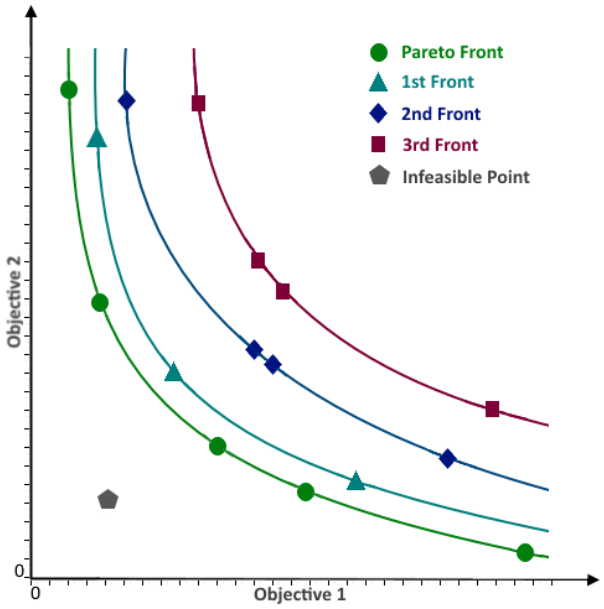


Figure 6 – Fronts in Multi-Objective Results [5].

The process of a multi-objective genetic algorithm, specifically NSGA-II, is laid out in Figure 7. It starts off with generating a population of size N. This is done in the same way as explained in the single objective algorithm. Chromosomes are created for each new member of the population where each of these chromosomes have a randomly generated set of genes.

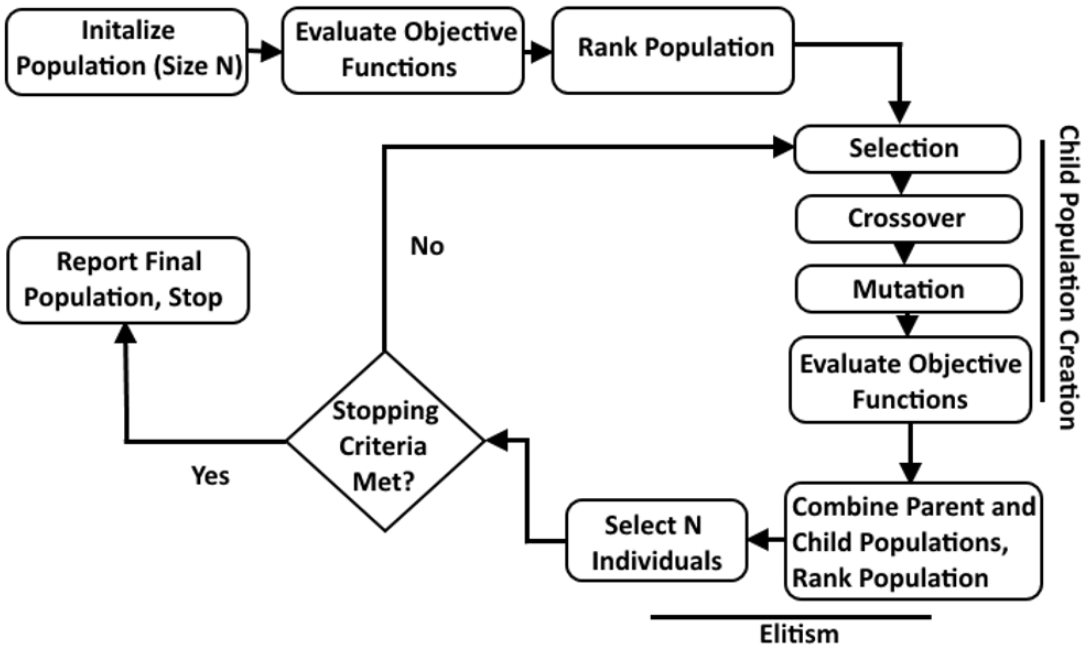


Figure 7 – Multi-Objective Flow Diagram [5]

Once there are N members in the new population, each one will be evaluated in each objective. The results for each objective will depend on the characteristics of the solution, which are determined by its genes. However unlike the single objective GA each objective value is stored separately and is not combined within a fitness function. The population is then ranked using the NSGA-II domination count where members of the population with a count of 0 will belong to the Pareto front (Illustrated previously in Figure 6).

Once the new population has been ranked the process then moves on to the evolution stage. This consists of 3 main steps; Selection, Crossover and Mutation. The aim of evolution is to create a child population, with the hope that the child population set holds better solutions than the parents. Selection is the process of selecting the parents to carry forward to the crossover stage. We will use Tournament selection here to illustrate the difference between single objective GAs and multi-objective GAs. Tournament selection randomly picks a small subset of the population to compete as described previously in section 4.1. However there is no fitness function, so there are no fitness values to compare. The comparison is down to the rank (or domination count) of the individual. This is because if the first solution is on the Pareto front and the second

is on the 2nd front we know that the first solution dominates the second and wins the tournament. If there is a case of the two solutions having the same rank, then they are evenly matched. So in this case randomly select between the two. The two winning solutions from the tournament selection are carried through to the crossover stage.

Crossover and mutation processes are the same as in the single objective GA. Once all the new children have been generated, they all need to go through the same evaluation process as the parents. Once the child population has been created there are now two populations of size N (the parent population and the child population). These populations need to be combined and ranked. Figure 8 illustrates this process where both the child and parent populations are combined together to create a population of size 2N. Then this Combined Population needs to go through the ranking process. This process of ranking is a natural way of maintaining elitism.

When the Combined Population is ranked it no longer matters which solutions were parents and which were children. It only matters which solutions dominate. The population of size 2N cannot be taken through to the next generation. So the Combined Population is cut down into the New Population. This is done by going through the Combined Population in rank order, adding solutions to the New Population and then stopping once the New Population is of size N.

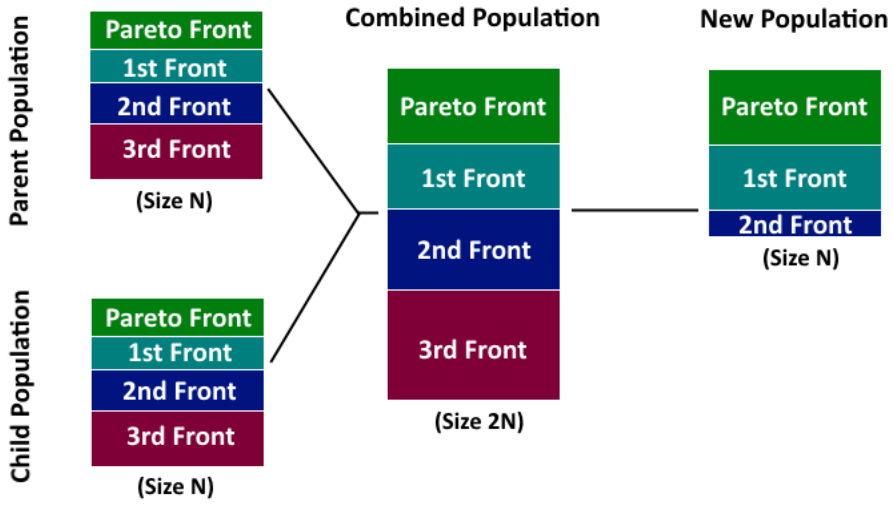


Figure 8 – Creating a New Population form the Combined Population [5].

Multi-Objective GAs, like NSGA-II, have stopping criteria just like single objective GAs. Convergence and max generation are perfectly acceptable criteria for this. If the criteria has not been met the GA will go back to the evolution stage and create a new child population set. If the stopping criteria has been met then the GA stops and the current set of solutions in the Pareto front are reported. If there are many solutions in the Pareto front, it will be up to the user or a subsequent algorithm to select which solution is the most appropriate for the current environment.

5. The Proposed Multi-Objective Genetic Type-2 Fuzzy Logic Based System for Mobile Field Workforce Area Optimization

5.1 The Proposed System Overview

Figure 9 provides an overview on the stages of the proposed multi-objective genetic type-2 fuzzy logic based system for mobile field workforce area optimization. The first step in this system is to collect the list of engineers and the list of SDPs to optimize. The engineers and SDPs will already be grouped together into teams and patches (WAs) from their current set-up, so the system organizes the entities into the groupings from the data presented.

The system now has the current setup of patches with their respective teams. This configuration is then put through the one-day simulation to assess how the current setup is performing. The one-day simulation cycles through each engineer and assigns them tasks based on their skills and the patch they are in. The simulation will attempt to assign the closest tasks to the engineer. Once a task has been assigned it will be removed from the task list.

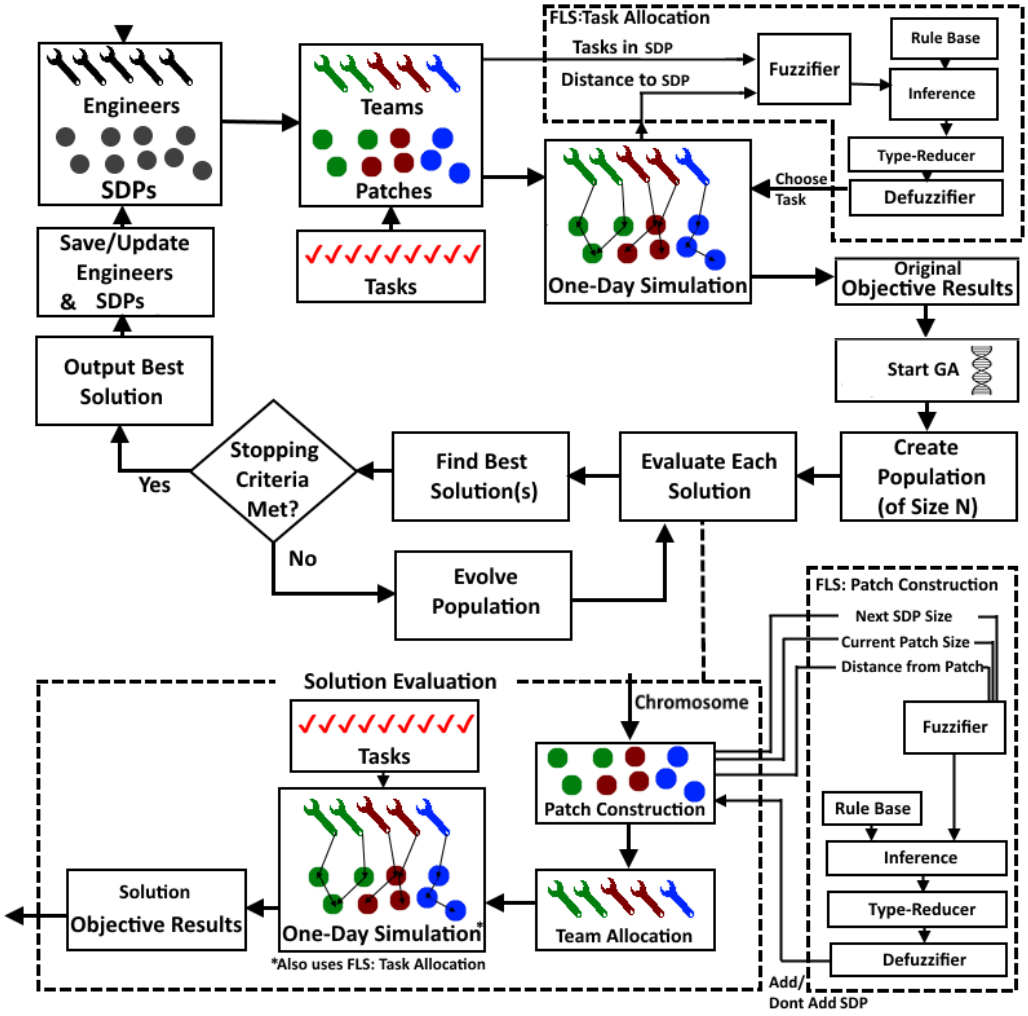


Figure 9 – The Proposed multi-objective Genetic Type-2 Fuzzy Logic Based System for Mobile Field Workforce Area Optimization.

Each engineer will be assigned tasks until their time has been filled or there are no more tasks available. Each engineer is allocated 7 hours and each task has an estimated completion time attached to it. When an engineer is assigned a task this time will be added to their utilized time, while the time it takes to travel to the task will be added to the engineers travel time (part of the engineers unutilized time). The distance traveled is also stored per engineer. The simulation will stop assigning tasks once the utilized time combined with the travel time is over 7 hours. The simulation will also stop assigning tasks if there are no more available tasks for that engineer to complete. Any remaining time an engineer has will be idle time, which is part of the engineer's unutilized time.

The one-day simulation step is where the Task Allocation Fuzzy Logic System can be applied. When choosing which task to assign to an engineer the distance and time to the task is fuzzified. The number of tasks at the SDP is calculated and fuzzified. This helps the simulation take into account the uncertainty of the travel time and to direct the engineer to SDPs with more tasks. More on this can be found is section 5.2.

Once each engineer has been cycled through, the system will calculate the objective results. The first value to calculate is the coverage. This is the total amount of hours of completed work. This is calculated be summing all the utilized time of the engineers. The second value is the total travel distance. This is calculated by summing all the total travel distances of the engineers. The third value is utilization, this is calculated by dividing the utilized time of an engineer by the max time (7 hours). This value is then expressed as an average across all engineers. The final value is balancing which adds up all the task time per patch (WA) and finds the number of hours different between the largest patch and smallest patch. Ideally this difference value should be 0, meaning perfect balancing.

Given that the current setup has been evaluated, these values can be used as a simple benchmark for the optimization process to improve upon. The system gives the user the option to adjust any of the GA's parameters before the optimization process is started.

When the GA is started it will create a new population of solutions. Each member of the population has P genes, where P is the number of patches to optimize for. Each gene is the centre location of a patch and the rest of the patch will be constructed from these points.

Each of the solutions need to be evaluated. The first step to this is building the patch setup from the centre points. There are certain restrictions that apply to the patch construction. SDPs in the same patch cannot be separated by rivers or by other patches. The patch construction works in the following way: Each center point works out who its neighbouring SDPs are, this is provided

by a precomputed neighbourhood index. This neighbourhood index was generated by pairing together SDPs that share the same boarders.

Then out of these neighbours, works out which is the closest. If no other patch has deemed that SDP to be the closest it will be added to the patch. The next patch will do the same. Each time a SDP is removed from the list and added to a patch, each patch has to recalculate who its available neighbours are. This is done by retrieving the neighbours of the current WA's SDPs from the matrix, then removing SDPs that have been added to any WA. The removal of SDPs already added means that SDPs cannot be mistakenly re-added.

The patch construction is where the Patch Construction Fuzzy Logic System can be applied. When it is being decided if an SDP should be added to a patch, the list of all neighbouring SDPs will be passed through the FLS whose inputs are the size of the SDP (in hours), the size of the patch (in hours) and the distance to the SDP from the centre point. More on this can be found in section 5.3.

Once the patches have been constructed from the centre points, the teams for each patch need to be assigned. This first step in this process is to assign each engineer to the patch they live in (or are closest to, if they do not live in any patch). This will usually mean the teams are extremely unbalanced as city/town patches will have over populated teams and rural patches will have underpopulated teams.

So the next step is to balance out the teams. This is done by a bidding process. The system will cycle through each overpopulated patch and 'sell off' its engineers to the highest bidders. Each underpopulated patch will cycle through the current overpopulated patch's engineers and give each a bid value. If there are no other bids for this engineer they will move over to the underpopulated patch, if there are other bids the highest bid wins. The bid value is made up of the distance the engineer is from the underpopulated patch, how much their skills are needed and the level of under-population the patch is at. Once the bidding process is complete the engineers should be spread as best as possible between the patches.

The newly constructed patches and teams will then go through the same one-day simulation process as the original setup (also using the Task Allocation FLS if required) if the generated solution is valid. There are certain criteria that if not met the solution will be rejected or altered before the one day simulation is run on it. This includes the number of patches constructed. As the user specifies the number of patches and each gene represents a patch center, any solution cannot have two genes that represent the same center point. Also, all SDPs have to be added to the patch design, so the list of unassigned SDPs has to be empty before the simulation can be run. If there are any SDPs on the list they will be assigned the same patch as their closest neighbour.

Once the solution has passed the checks and is deemed valid, the objective values for this solution will be calculated. The GA will carry out the 'Solution Evaluation' for every solution it generates. More about how the single objective and multi-objective algorithms affect the optimization can be found in section 5.4.

With each solution in the population evaluated, regular GA process are resumed. The stopping criteria that is currently being utilized in the system is the number of generation. Once the GA has stopped the results are reported and output files can be generated. The output files list each engineer and their newly assigned patch and the structure of these new patches.

5.2 Fuzzy Task Allocation

Figures 10, 11 and 12 show the interval type-2 fuzzy sets used to decide which tasks to pick up. The average distance to a task (AD in Figure 10) is calculated for the area being optimized. This is done before the initial one-day simulation when the teams and SDPs are first loaded. The average amount of work in an SDP for the area (AW in Figure 11) is also calculated at this point. Figure 12 shows the output of the interval type-2 FLS which represent the probability of picking a task. This interval type-2 FLS uses the Centre of Sets type-reduction as it has a reasonable computational complexity.

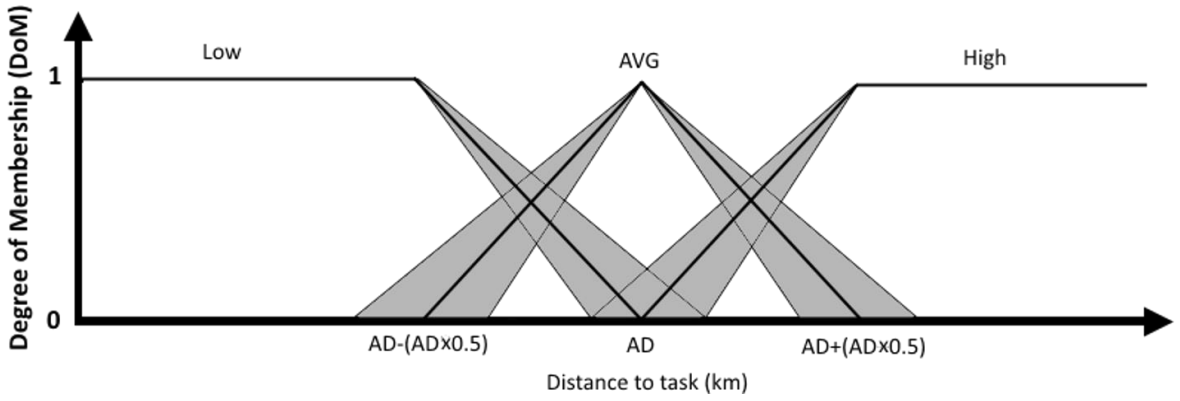


Figure 10. Distance to Task Type-2 Fuzzy Sets

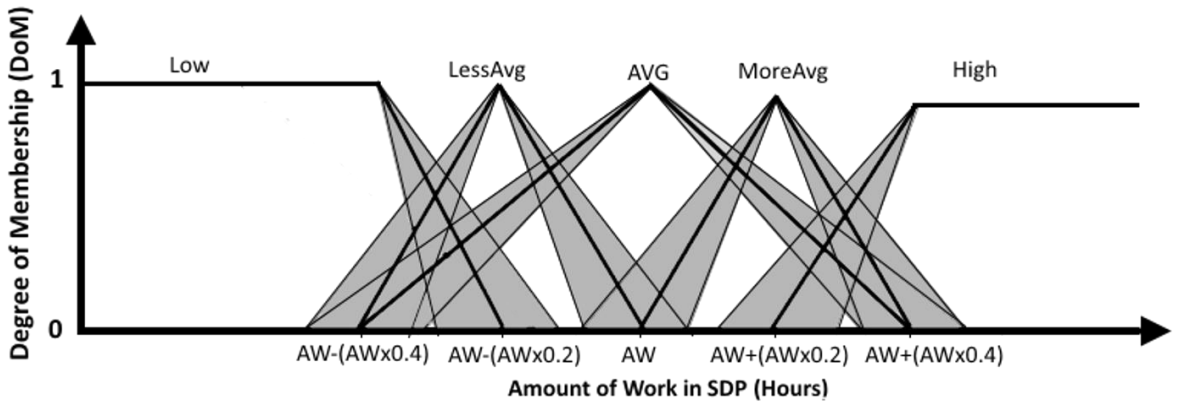


Figure 11. Jobs in SDP Type-2 Fuzzy Sets

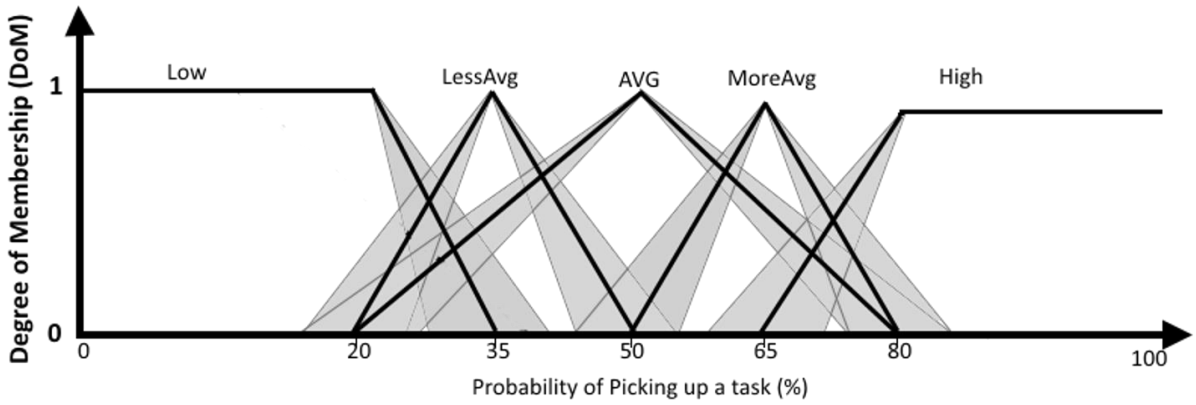


Figure 12: Probability of Picking Task Type-2 Fuzzy Sets

The footprint of uncertainty, shown in Figures 10, 11 and 12 as the grey areas, is variable. The uncertainty value is given to the system as an input and the footprint extends each side of the base point by the required percentage. The percentage of uncertainty is a variable in the experiments and results for this can be found in Section 6. The base points of the membership functions were tuned by running experiments to find the most suitable setup.

The values for the average distance (AD) and average work (AW) had to be calculated so that their values were relative to the area that was being optimized. For example an average distance per job in London might be 100m but in the Scottish Highlands this value might be 5km or more. Having the base points relative to the area is important, else input values will be wrongly categorized relative to the local area.

The reason for the triangular and trapezoid membership functions is that the alternative would be to have bell-shaped membership functions, created by using standard deviation. However due to the need to generate the membership functions dynamically, it is faster to use the triangular and trapezoid membership functions generated from calculated base points and scale them accordingly. Table 2 shows the list of rules used in this FLS.

Table 2 – Task Allocation Rule Base

Distance To Task	**Tasks at SDP**	**Probability of Choosing SDP**
Low	Low	Average
Low	LessAvg	Average
Low	Average	MoreAvg
Low	MoreAvg	High

Low	High	High
Average	Low	LessAvg
Average	LessAvg	LessAvg
Average	Average	Average
Average	MoreAvg	MoreAvg
Average	High	MoreAvg
High	Low	Low
High	LessAvg	Low
High	Average	Low
High	MoreAvg	LesAvg
High	High	Average

The following is an example of how this fuzzy system would work:

The system wants to find the next best SDP to send an engineer to. So the system finds out that the average amount of work in all SDPs in the WA. This is 5 hours. The average distance to a task is calculated to be 2 kilometers. The current engineer has 3 SDPs to choose to go to next. The first is 3 kilometers away with 5 hours worth of work. The second is 1 kilometer away with 6 hours worth of work and the third is 2 kilometers away with 8 hours worth of work.

Given these options the fuzzy system would classify the first option as High distance and Average amount of work giving a low probability of choosing that SDP. The second option would be classified as Low distance and More than Average amount of work giving a high probability of choosing the SDP. The third option would be classified as Average Distance and High amount of work giving a more than average probability of being chosen. With these 3 results their output defuzzified values are compared which would give option 2 the highest value and this SDP would then be assigned to the current engineer.

5.3 Fuzzy Patch Construction

Figures 13, 14 and 15 show the type-2 fuzzy sets that are used in the fuzzy patch construction. When the area to be optimized is initially loaded up, the average patch size in hours of work, Patch Average (PA), is calculated along with the average SDP size (SDPA). This is because these values can vary a lot between urban and rural areas. Hence, for London the average SDP may carry 500 hours' worth of work, but in the Scottish Highlands there may only be an average of 50 hours' worth of work, or even less.

The base points of the membership functions were tested to see if reasonable categorization of SDPs and patch sizes were given. As before, the reason for the triangular and trapezoid membership functions is the alternative would be to have bell-shaped membership functions, created by using standard deviation. Due to the need to generate the membership functions dynamically it is faster to use the triangular and trapezoid membership functions. This interval type-2 FLS also uses the Centre of Sets type-reduction, again because it has a reasonable computational complexity.

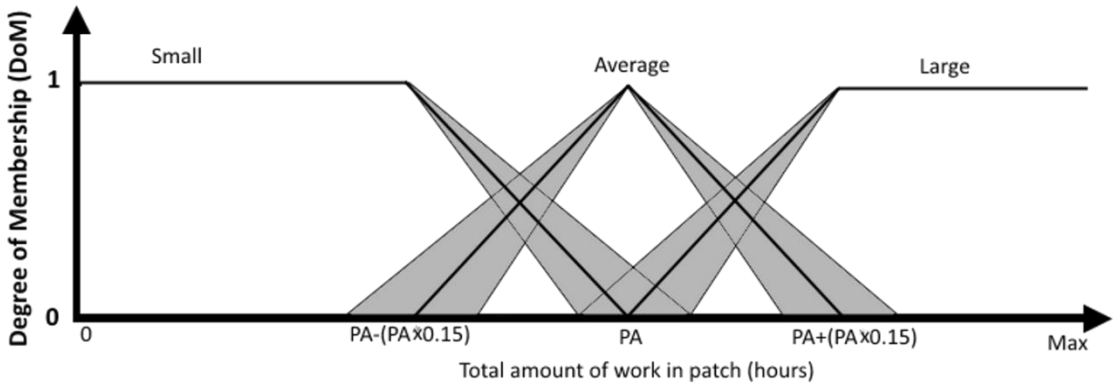


Figure 13. Patch Size Average Type-2 Fuzzy Set

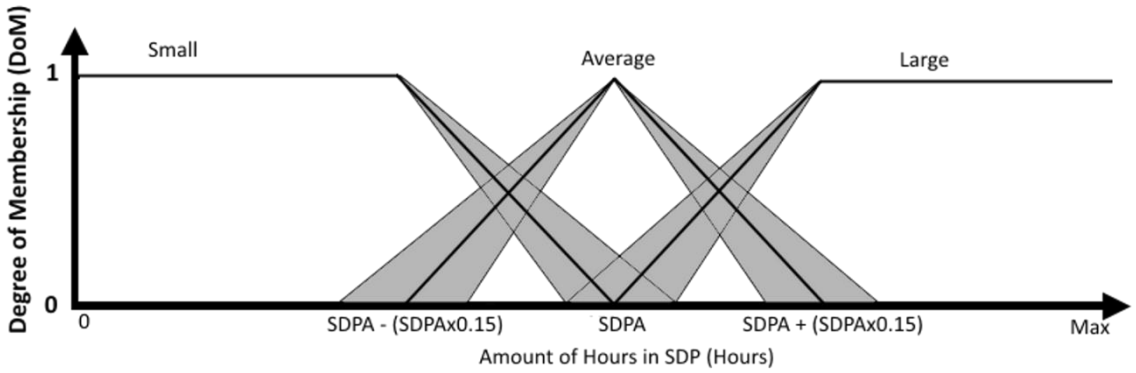


Figure 14. SDP Size Average Type-2 Fuzzy Set

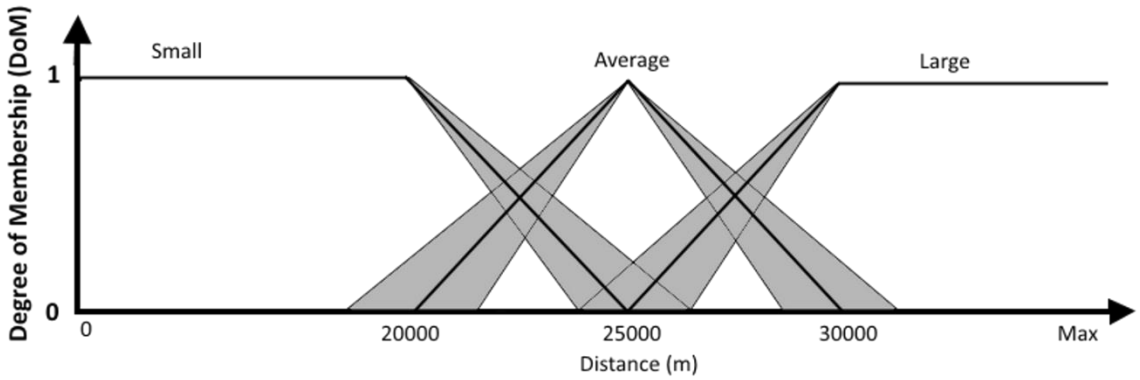


Figure 15. Average Distance Type-2 Fuzzy Set

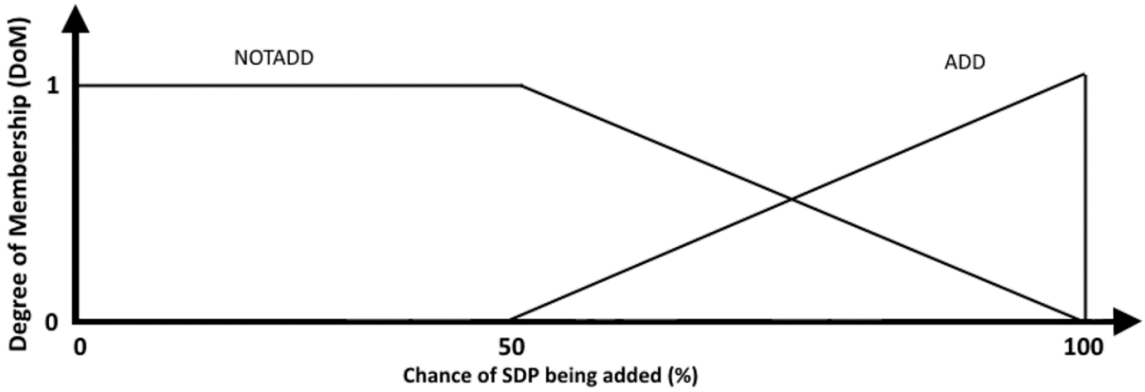


Figure 16. Add/Not Add Type-2 Fuzzy Set.

The task of this fuzzy logic system is to more sensibly add SDPs to patches (WAs). The center points of the patches are provided to the fuzzy system (these center points are the initial SDPs allocated to each patch). The size of the patch is re-calculated each time an SDP is added to it. Figure 16 shows the type-1 fuzzy sets representing the output of the type-1 FLS which is the chance of an SDP being added.

This is a more sensible way of adding SDPs to patches because the alternative way is to allocate an SDP to the patch that is deemed closer based on travel distance. This does not take into account the size of the SDP or the size of the patch it is being added to. The Add/Not Add membership functions were designed in such a way that a rule with a not add consequence would have more of an impact on the final outcome than an add consequence. The output values are compared between the patches, with the SDP being added to the patch with the highest output value. Table 3 shows the list of rules used in the fuzzy patch constructor.

Table 3 – Patch Construction Rule Base

WA Size	Distance to SDP	SDP Size	Consequence
Small	Small	Small	Add
Small	Small	Average	Add

Small	Small	Large	Add
Small	Average	Small	Add
Small	Average	Average	Add
Small	Average	Large	Add
Small	Large	Small	Add
Small	Large	Average	Add
Small	Large	Large	DontAdd
Average	Small	Small	Add
Average	Small	Average	Add
Average	Small	Large	DontAdd
Average	Average	Small	Add
Average	Average	Average	Add
Average	Average	Large	DontAdd
Average	Large	Small	Add
Average	Large	Average	DontAdd
Average	Large	Large	DontAdd
Large	Small	Small	Add
Large	Small	Average	DontAdd
Large	Small	Large	DontAdd
Large	Average	Small	DontAdd
Large	Average	Average	DontAdd
Large	Average	Large	DontAdd
Large	Large	Small	DontAdd
Large	Large	Average	DontAdd
Large	Large	Large	DontAdd

The following is an example of how this fuzzy system would work:

The system wants to find the next best SDP to add to the current WA. So the system finds out that the average amount of work in all SDPs in the area to be designed. This is 5 hours. The current WA is deemed to be an Average sized WA based on its current total amount of work. The current WA has 3 SDPs to choose from to add to itself. The first is 3 kilometers away with 5 hours worth

of work. The second is 2 kilometer away with 6 hours worth of work and the third is 2.5 kilometers away with 2 hours worth of work.

Given these options the fuzzy system would classify the first option as Large distance and Average amount of work giving a consequence of suggesting not to add this SDP to the current WA. The second option would be classified as Low distance and Large amount of work giving a consequence of suggesting not to add this SDP to the current WA. The third option would be classified as Average Distance and Small amount of work giving a consequence of suggesting too add this SDP to the current WA. With these 3 results their output defuzzified values are compared which would give option 3 the highest value and this SDP would then be added to the WA.

After one SDP has been added the system will move onto the next WA. The WA will only get a chance to add another available SDP to it once all the other WAs have had a chance. It is worth noting that it does not matter how low the score is from this fuzzy system, the highest value always wins. This is to ensure that all exchanges are added to a WA, even if that means adding a large SDP to a Large WA. Ultimately this will just mean this solution will perform badly in the patch balancing objective, yet it would still be a valid solution as all SDPs would have been added to the design.

5.4 Genetic Algorithms

Both single objective and multi objective genetic algorithms can be used with the system and the different results given by each can be found in section 6.1. If a single objective GA is being used then the following fitness function (equation 7) will be used to assess the solutions.

$$Fitness = \frac{(Coverage \times W_1) \times (Utilization \times W_2)}{(Travel \times W_3) \times (Balancing \times W_4)} \quad (7)$$

W is the weighting of each objective, w_1 is the weighting of the coverage objective, w_2 is the weighting of the utilization objective, w_3 is the weighting of the travel objective and w_4 is the weighting of the balancing objective. Changing these values pushes the optimization to find solutions that satisfy the objectives with the higher weightings. Any weighting could be set to 0 to remove that objective from the fitness function. If this is done, the objective value combined with the weighting defaults to a value of 1.

If a multi-objective GA is being used there will be no fitness values, only each individual objective value. The output will also be a set of solutions (provided there is more than one solution on the Pareto front). This allows managers to pick a set up that is best suited bases on local knowledge that could not be taken into account by the proposed system. This adds an extra layer of validation.

6. Experiments and Results

The aim of the experiments is to take an existing structure of WAs in a telecommunications domain with its current patch set up and teams of engineers, then run it through the optimization process to see how well the working areas get optimized. These experiments are then repeated with potential improvements added to the optimization to see the impact these potential improvements will make. The experiments involved altering the optimization process by gradually increasing the use of more advanced optimization methods.

The process started by comparing the use of single and multi-objective GAs and then progressed to evaluate the effect of employing type-1 and type-2 FLSs. The real world tool (which is a leading tool for mobile workforce allocation) created for this process is shown in Figure 17. There are no other tools that attempt to handle the real world uncertainties or multi-objective nature of real constraints for this type of problem. Therefor current systems lead to sub optimal performance because of this, the tool shown in figure 17 has been created. The tool also allows the visualization of the WAs and SDPs.

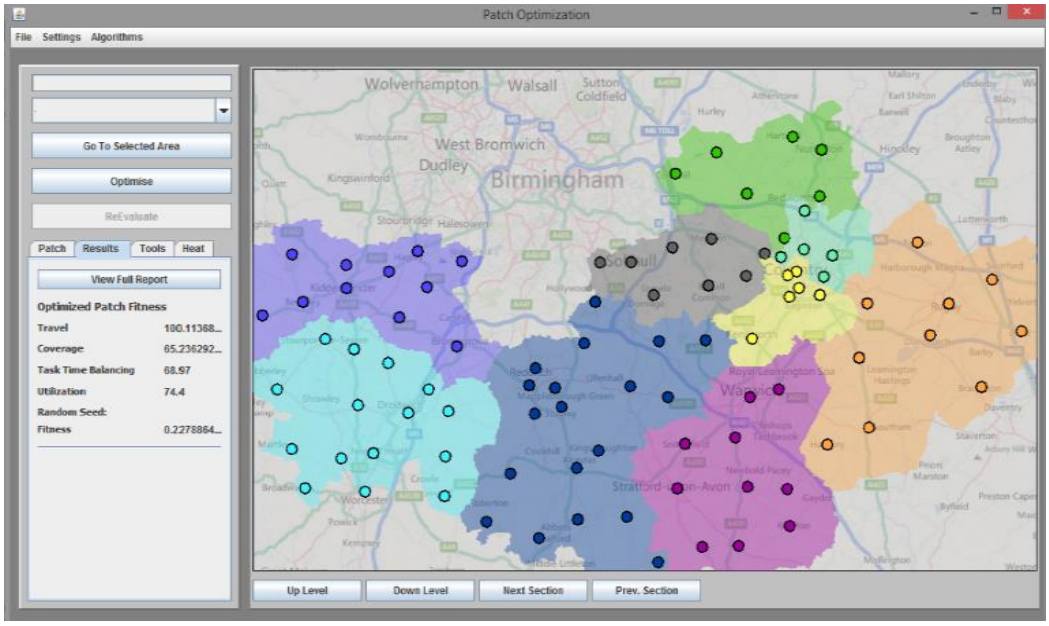


Figure 17. The Mobile Field Workforce Area Optimization Visualization & Optimization Tool

For all of the experiments the both the GA and MOGA were set to carry out 20 generations and have a population size of 40. Due to the complexity in generating designs of WAs and simulating one-day, the time it took to complete one generation was significant enough to prevent more generations from being carried out. Both the GA and the MOGA ran with a crossover rate of 0.4 and a mutation rate of 0.05. These settings were already good in the current real-world tool that is used on a daily bases. So these settings were kept the same for the following

experiments for a fair comparison on how fuzzy logic and an MOGA would affect those daily results.

6.1 Single Vs Multi-Objective GAs

The goal in this first experiment was to see if the multi-objective genetic algorithm (MOGA), NSGA-II, optimized more objectives than the standard Single Objective Genetic Algorithm (SOGA). Where Travel is measured in kilometers (km) and balancing and coverage are measured in hours (hrs.).

Table 4 – Original Vs Single Vs Multi-Objective GA

ORIGINAL SCORE			**SINGLE OBJECTIVE**			**MULTI OBJECTIVE**		
Travel (km)	**Balancing (hrs.)**	**Coverage (hrs.)**	**Travel (km)**	**Balancing (hrs.)**	**Coverage (hrs.)**	**Travel (km)**	**Balancing (hrs.)**	**Coverage (hrs.)**
80.00	17.00	455.00	**73.86**	22.28	453.25	**73.20**	44.70	**460.59**
99.00	68.00	476.00	102.17	**38.21**	**485.64**	**97.39**	**64.38**	**492.11**
50.00	102.00	212.00	52.55	**12.13**	**214.73**	**45.86**	**48.40**	**214.74**

Table 4 shows a sample of the results collected when comparing single and multi-objective GAs. The first row of results from Table 4 shows that the SOGA optimized in travel only, when compared to the original setup of the patch. Whereas the MOGA optimized in both travel and coverage when compared to the original. Although the SOGA did a better job of optimizing in the balancing objective than the MOGA, neither beat the original at balancing in this case.

In the second and third rows of results the SOGA optimizes in balancing and coverage but not travel. However the MOGA optimizes in all objectives when compared to the original patch set up. In the SOGA results the balancing objective is better than the MOGA result, however this is due to the fact that the SOGA has sacrificed the travel objective to reach this level of balancing. The goal is to optimize in all objectives, the SOGA fails to do this because of a good result in one of the objectives that overrides the poor result in another.

In the results presented in Table 4, the MOGA optimizes in more objectives than the SOGA when compared to the original patch set up. This suggests that MOGAs are better at dealing with problems with multiple conflicting objectives.

6.2 Single Vs Multi-Objective GAs with Type-1 Fuzzy Logic

The next set of experiments aim to assess the impact of the inclusion of type-1 fuzzy logic in the patch construction and one-day simulation processes. In the results shown in Table 5, there are two different areas (A1 and A2) that are optimized. Rows 1 to 3 show that in area 1 (A1) when a SOGA is used and the FLSs are used, we increase the coverage by 24.72% reduce the imbalance between the WAs by 46.10% and increase the utilization by 24.72%. Coverage and utilization are linked very closely together so the rate of change of these values are almost the same, this pattern continues through all of the results in Table 5. However as a result of these significant improvements we do get an increase in the level of travel by 8.75%.

Table 5 – Addition of Type-1 FLS to Patch construction and Job Allocation

Type	Travel (km)	Coverage (hrs.)	Balancing (hrs.)	Utilization (%)
A1 SOGA Without Fuzzy	122.63	763.74	369.16	57.03
A1 SOGA With Fuzzy	133.37	952.53	170.18	71.13
A1 SOGA Effect with fuzzy	**8.75%**	**24.72%**	**-46.10%**	**24.72%**
A1 MOGA Without Fuzzy	135.70	1014.15	70.02	75.72
A1 MOGA With Fuzzy	48.14	1021.36	82.19	76.27
A1 MOGA Effect with fuzzy	**-64.53%**	**0.71%**	**17.38%**	**0.73%**
A2 SOGA Without Fuzzy	123.48	624.38	310.20	61.50
A2 SOGA With Fuzzy	145.87	739.75	174.01	72.97
A2 SOGA Effect with fuzzy	**18.13%**	**18.47%**	**-43.90%**	**18.65%**
A2 MOGA Without Fuzzy	165.44	799.16	74.80	78.72
A2 MOGA With Fuzzy	44.90	779.90	16.19	76.82
A2 MOGA Effect with fuzzy	**-72.86%**	**-2.41%**	**-71.06%**	**-2.41%**

In rows 4 to 6, we see the results of the MOGA on area 1 with and without the FLSs. In this instance, we see that we get a 64.53% reduction in travel, with a slight increase in coverage and utilization when the FLSs are used. This very small increase may be due to the coverage being topped out by the MOGA (Very little work left in SDPs). As the MOGA improves over the SOGA results, the coverage value may have already hit the upper limits, so the potential

improvements that could be made by the FLSs on coverage are very small. Hence the much improved travel objective, as the FLSs cannot improve on coverage, they can improve on the rate of travel per hour of task. In this example it is the balancing objective that has suffered to the largest degree. However when comparing this value to the SOGA with FLSs value we still get a 51.71% reduction in the imbalance of the WAs.

When the same experiments were run on area 2 (A2) we get similar results for the SOGA, rows 7 to 9 where we achieved an 18.47% increase in coverage and 18.65% increase in utilization and a 43.90% reduction in the imbalance of the WAs.

When we look at the MOGA results for area 2, rows 10 to 12, we get a 72.86% reduction in travel and a 71.06% reduction in the imbalance of the WAs. As a result of these very large improvements we suffer a small decrease in coverage and utilization at a rate of 2.41% each. Again this may be because the coverage values have hit the upper limit in the MOGA without the FLSs so there is little to improve upon.

If we take area 2 as an example and compare the SOGA without the FLSs and the MOGA with the FLSs, we see 63.64% reduction in travel, a 24.91% increase in coverage and utilization and a 94.78% reduction in the imbalance of the WAs which is regarded as significant improvement in all areas and most notably in travel and patch balancing which are the 2 primary areas where the FLSs are applied.

The results shown in Table 5 suggest that including the FLSs in Task Allocation and Patch Construction have a significant improvement on the results generated by the proposed system.

6.3 Type 1 FLSs Vs Type 2 Fuzzy FLSs

The third experiment aims to test the impact type-2 FLSs have on the results. These following results include the type-1 FLS results and the type-2 FLS results with different uncertainty values. If the uncertainty value is 1% this means that the footprint of uncertainty extends 1% (of the average value) either side of the base point.

For this experiment seeding was used in the GA to allow a more accurate comparison of the results. This is possible because the GA for each run is given the same starting population and conditions, giving a more accurate reflection of how the final outcome is affected by the different types of FLS and uncertainty values. A single objective GA was used in this experiment so that the fitness values can be directly compared between results and there is no ambiguity as to which result is better.

Table 6 gives a sample of the results collected for the comparison of the type-1(T1) and type-2 (T2) FLSs.

Table 6 – Type 1 FLS vs Type-2 FLS in Work Area Optimization System

Type (U)	Travel (km)	Coverage (hrs.)	Balancing (hrs.)	Utilization (%)	Fitness
T1	180.30	819.33	133.28	62.93	**1.83**
T2 (1%)	165.22	833.72	111.47	64.03	**4.60**
T2 (3%)	157.25	794.94	161.30	61.06	**2.82**
T2 (5%)	180.30	819.33	133.28	62.93	**1.83**

In Table 6, the type-1 FLSs gave an overall fitness value of 1.83. This is now compared with the results from the type-2 FLSs where three uncertainty values were tested. A 5% uncertainty actually gave the same result as the type-1 FLSs, this is possible because of the seeding and the same optimization conditions. A 3% uncertainty value significantly improved on the fitness by 54%. Finally an uncertainty value of 1% gave a fitness value of 4.60 a 151% increase over the type-1 FLSs.

The results shown in Table 6 suggest that upgrading from a type-1 FLS to a type-2 FLS can have significant improvements to the final results. However the uncertainty values have to be tuned correctly for these results to be realized.

6.4 Progressive Results

One final set of results aims to test the suggestions given by the previous experiments in one sequential real time test. These results are not an average, not seeded, use the same patch, and run as if they would be in the real world. Coverage here is expressed as a percentage of the total amount of work available.

Table 7 – Progressive Real World Run Results

	Travel (km)	Coverage (%)	Balancing (hrs.)	Utilization (%)
Original	**172.00**	**71.34%**	**68.96**	**63.88%**
Single	**187.16**	**68.86%**	**110.16**	**61.67%**
Multi	**173.26**	**68.46%**	**54.21**	**61.30%**
Multi-Fuzzy T1	**67.01**	**69.68%**	**62.09**	**62.40%**

Multi-Fuzzy T2 (Tuned)	**68.15**	**71.25%**	**30.08**	**63.81%**

Table 7 shows the results from the progressive tests. The original patch values are given in row 1. The first step is to optimize this patch with the SOGA. Row 2 shows us that on this occasion the SOGA failed to optimize in any objective. This means that the optimization would have to be run again and the GA setting would need to be tuned for this specific area to get a better result. This would cause frustration to the user and cost time.

Row 3 shows us the most suitable solution from the MOGA. On this occasion the MOGA has optimized in balancing, travel is less than 1% worse so that can be seen as the same. However the MOGA has failed to optimize on coverage and utilization. If the user was looking to only improve on balancing and was happy to suffer the reduction in the other two objectives then this may be acceptable, else the optimization would need to be run again.

Row 4 shows the most suitable solution from the MOGA using type-1 FLSs in the optimization. Here we can see that the MOGA has now optimized in two objectives, with travel being significantly improved, now only 38.96% of the original travel value. However coverage and utilization still suffer. But they suffer less than if the MOGA did not use the type-1 FLSs. As there is a 1.8% increase in both coverage and utilization over the MOGA that does not use any FLSs.

Finally row 5 shows the most suitable MOGA result with type-2 FLSs (that has been tuned to 1% uncertainty). On this occasion two objectives have been optimized and the remaining two do not suffer any noticeable fall, less than 0.13% for coverage and less than 0.11% for utilization. This gives the user a solid result and can confidently say that this new patches are better than the old patches. This is on one run of the optimization and with no specific tuning of the GA required, which is great from a user's point of view.

As a result we can say that these results support a multi-objective genetic type-2 fuzzy logic based system for mobile field workforce area optimization.

6.5 Subjective Evaluations

Figures 18, 19, 20 and 21 show the visualization of how the results change with each incremental improvement of the proposed system. Figure 18 shows the SOGA try and divide the selected area, into 9 WAs. The selected area includes both rural and urban areas, including the densely populated city area and surrounding suburbs. The single objective optimization has split the city area (circled in Figure 18) up into 3 WAs, this is not good as engineers will have to keep travelling in and out of the city. The other WAs are either too large or too small.

Figure 19 shows one of the solutions on the Pareto front from the MOGA with no FLSs. This solution is slightly better as it has sectioned off the center of the city. But this WA is now too small as the outside of the city forms part of another WA to the north. This has left one suburb in a very oversized WA and another in a small WA. However the remaining WAs are of reasonable size.

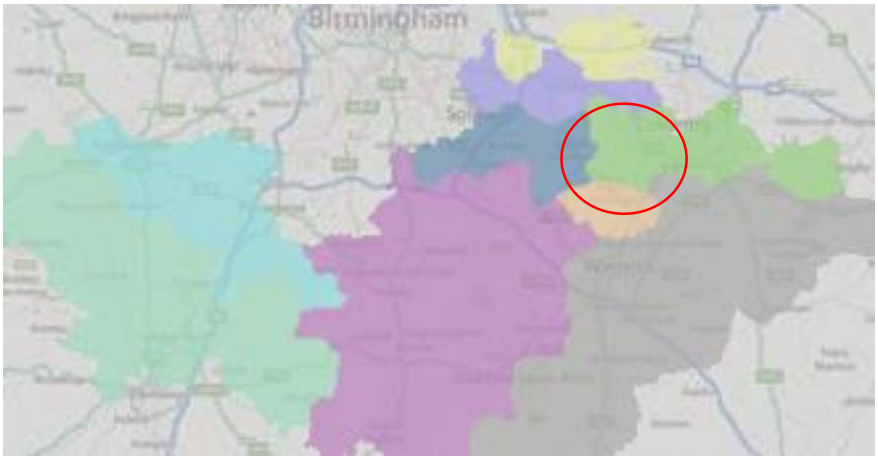


Figure 18 – SOGA Optimization Design (main city area is circled)

Figure 19 – Multi-Objective Optimization

Figure 20 shows a solution that used the MOGA with type-1 FLSs in the optimization process. This has done a slightly better job of sectioning off the city but there are a few SDPs that were not included in that WA. There is also a WA in the West that is too small and there is a suburb still in an oversized WA.

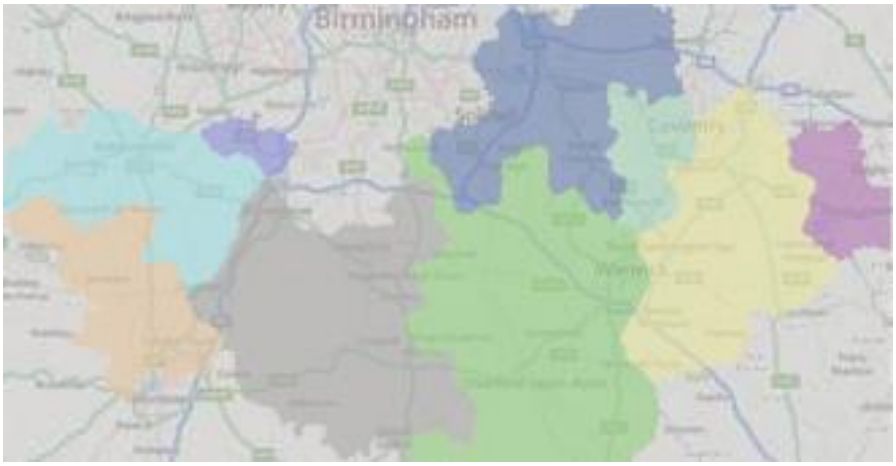


Figure 20 – MOGA with Type-1 Fuzzy.

Figure 21 – MOGA- with Type-2 Fuzzy.

Figure 21 shows a solution that has replaced the type-1 fuzzy with type-2 fuzzy logic in the MOGA. This solution has done a good job of sectioning off the city. Each WA is more balanced in size and even the town to the west is its own WA. There also seems to be reasonable utilization of the road networks in the area. The MOGA with Type-2 Fuzzy has produced the most sensible WA (patch) designs visually (this is important to the engineers and managers who have to accept these designs) as well as the best results from the simulation.

7. Conclusions & Future Work

In this paper, we have presented a multi-objective genetic type-2 fuzzy logic based system for mobile field workforce area optimization. The benefits of having such a system in real world utilities companies include increased utilization of the workforce leading to reduced costs. The data used was based on a large mobile field workforce and collected over 3 months. The system has been tested in a real world telecommunication service industry.

The proposed system uses GA based optimization, however it was explained that traditional single objective GAs cannot fully handle optimization processes with multiple

objectives, especially when those objectives are conflicting. This is an important challenge to overcome, our solution to this was to introduce multi-objective genetic algorithms to the system, specifically NSGA-II. This gave the optimization process the ability to compare the results of the individual objectives between possible solutions and rank them accordingly.

There were additional challenges to overcome to build the system. This included a suitable way to cluster together SDPs that satisfied the constraints. These constraints included avoiding geographical obstacles like rivers and balancing out the work evenly between the WA clusters. This was solved using a neighbourhood matrix and monitoring the size of each WA as they had more SDPs added to them.

Another problem was how teams were built for the new WA structures, as the old teams would become obsolete when the WAs change. This was solved using a bidding system for each WA that took into account the engineers location and skill and the current WA team size.

As the proposed system is designed to tackle a real world problem with real world data, there are many uncertainties. Because of these uncertainties we introduced why fuzzy logic is adapted to handle such uncertainties and we also outlined the differences between type-1 fuzzy logic systems and type-2 fuzzy logic systems. The experiments that we ran showed that the system performed much better with the inclusion of fuzzy logic. The results were further improved when type-2 was used instead of type-1.

To fully evaluate each aspect of the proposed systems we ran through several experiments, each designed to assess the impact of the different methodologies. The results from these experiments showed that a multi-objective system was able to optimize in more objectives than a single objective system. The results also showed that including type-1 fuzzy logic systems on the task allocation and the patch construction parts of the optimization improved the results the system generated. With one example showing that we could have better performance in all objectives when compared to the SOGA system that employed crisp logic. With some minimization objectives being reduced by up to 94.78%.

Finally the results showed that upgrading the type-1 fuzzy logic systems to type-2 further improved on the results, giving up to 151% improvement over type-1 fuzzy in some instances. The system presented has significantly improved the solutions generated by the current system. We have seen from the progressive results the multi-objective genetic type-2 fuzzy logic based system can produce results that are significantly better than the current system, reducing the travel by 63.59%, increasing the coverage by 2.39%, reducing the imbalance of the WAs by 72.69% and increasing the utilization by 2.14%. The result produced is then significantly improved over the original to be implemented in the real-world environment. Whereas the current systems result has to be rejected.

As this is a real world problem being tackled there are many aspects that could be improved upon to have a system that generates even stronger results. One area of improvement is where the parameters of the type-2 systems could be optimized.

There are also limitations with the work presented, which will be addressed in our current and future work. The system is limited to planning for medium to long term WA designs. The system could therefore be improved by optimizing all objectives in real-time and attempt to converge rapidly on a solution to handle any changes in the environment. This way the patch designs could be up kept up to date and help reduce the potential risks to utilization that could not be planned for. These risks could include under or over estimating the number of engineers that will leave in the medium or long term. This could also include vehicles being damaged and therefore not available to engineers to travel to SDPs.

Work area optimization is just one aspect to the overall vision of workforce optimization. Another domain that should be tackled in our future work is the optimization of the engineers. Each engineer has a set of skills, however they can be trained to gain more skills or they can stop being assigned tasks they underperform in. Another area we aim to improve upon in our future work is the optimization of the parameters of the type-2 fuzzy systems employed in this work. Currently they were tuned manually through tests, however using an optimization algorithm may improve the performance.

8. References

[1] X. Chen-Guan, Z. Mai-huan, L. Yue-Peng, C. Nan-Xiang, The Water Resource Optimal Allocation Based on Multi-Objective Genetic Algorithms, in: 2010 International Conference on Computational and Information Sciences, 2010, pp. 877-880.

[2] K. Deb, A. Pratap, S Agarwal, T. Meyarivan, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, IEEE Trans. on Evol. Comput. 6 (2) (2002) 182-197.

[3] R. Domberger L. Frey, T.Hanne, Single and Multiobjective Optimization of the train staff planning problem using genetic algorithms, in: IEEE Congress on Evolutionary Computation, 2008, pp. 970-977.

[4] W. Fanm, Z, Gurmu, E. Haile, A Bi-Level Metaheuristic Approach to designing Optimal Bus Transit Route Network, in: 3rd Annual International Conference on Cyber Technology in Automation, Control and Intelligent Systems, 2013, pp. 308-313.

[5] T. Goel, Elitist Non-dominated Sorting Genetic Algorithm: NSGA-II, http://www2.mae.ufl.edu/haftka/stropt/Lectures/multi_objective_GA.pdf [Last Accessed: 20th July 2015].

[6] H. Hagras, A Hierarchical Type-2 Fuzzy Logic Control Architecture for Autonomous Mobile Robots, IEEE Trans. on Fuzzy Syst. 12 (4) (2004) 524-539.

[7] H. Hagras, Type-2 FLCs: A new Generation of Fuzzy Controllers, IEEE Comput. Intell. Magazine. 2 (1) (2007) 30-43.

[8] H. Hagras, Type-2 fuzzy logic controllers: a way forward for fuzzy systems in real world environments, Computational Intelligence: Research Frontiers, Springer Berlin Heidelberg, 2008.

[9] H. Hagras, C. Wagner, Towards the Widespread Use of Type-2 Fuzzy Logic Systems in Real World Applications, IEEE Comput. Intell. Magazine. 7 (3) (2012) 14-24.

[10] H. Hagras, D. Alghazzawi, G. Aldabbagh, Employing type-2 fuzzy logic systems in the efforts to realize ambient intelligent environments, IEEE Comput. Intell. Magazine. 10 (1) (2015) 44-51.

[11] K. Hossain, A. El-Saleh, M. Ismail, A Comparison between binary and continuous Genetic Algorithm for Collaborative Spectrum Optimization in Cognitive Radio Network, in: IEEE Student Conference on Research and Development, 2011, pp. 259-264.

[12] N. N. Karnik, J. M. Mendel, Centroid of a Type-2 Fuzzy Set, Inform. Sci. 132 (2001) 195-220

[13] Y. Liu, S. Zhao, X. Du, S. Li, Optimization of Resource Allocation in Construction Using Genetic Algorithms, in: Proceedings of the 2005 International Conference on Machine Learning, 2005, pp. 18-21.

[14] C. Lynch, H. Hagras, V. Callaghan, Embedded Type-2 FLC for Real-Time Speed Control of Marine & Traction Diesel Engines, in: 14th IEEE International Conference on Fuzzy Systems, 2005, pp. 346-352.

[15] C. Lynch, H. Hagras, V. Callaghan, Embedded interval type-2 neuro-fuzzy speed controller for marine diesel engines, in: Proceedings of the IPMU, 2006, pp. 1340-1347.

[16] J. M. Mendel, R. I. John, F. Liu, Interval Type-2 Fuzzy Logic Systems Made Simple, IEEE Trans. on Fuzzy Syst. 14 (6) (2006) 800-807.

[17] A Mohamed, H. Hagras, S. Shakya, G. Owusu, A Fuzzy-Genetic Tactical Resource Planner for Workforce Allocation, in: Conference on Evolving and Adaptive Intelligent Systems, 2013, pp. 98-105.

[18] D.N. Mudaliar, N.K. Modi, Unraveling Travelling Salesman Problem by genetic algorithm using M-Crossover Operator, in: 2013 International Conference on Signal Processing Image Processing & Pattern Recognition, 2013 pp. 127-130.

[19] T. Murata, H. Ishibuchi, Positive and Negative Combination Effects if Crossover and Mutation Operators in Sequencing Problems, in: Proc. of the IEEE Int. Congress on Evolutionary Computation, 1996, pp. 170-175.

[20] N. Nikolaev, L. M. de Menezes, H. Iba, Overfititng Avoidance in Genetic Programming of Polynomials, in: Proceedings of the 2002 Congress on Evolutionary Computation, 2002, pp. 1209-1214.

[21] T. Oda, A. Barolli, E. Spaho, L. Barolli, F. Xhafa, J. Iwashige, Genetic Algorithms for Node Placement in WMNs: Effect of Changes in Population Size and Number of Generations, in: 7th International Conference on Broadband, Wireless Computing, Communication and Applications, 2012, pp. 356-361.

[22] R. Taormina, K. Chau, Neural Network River Forecasting with Multi-Objective Fully Informed Particle Swarm optimization, Journal of Hydroinformatics 17 (1) (2015) 99-113

[23] N. Safaei, D. Banjevic, A.K.S Jardine, Workforce Planning for Power Restoration: An Integrated Simulation-Optimization Approach, IEEE Trans. on Power Syst. 27 (1) (2012) 442-449.

[24] R. Saravani, R.K. Moghaddam, Optimal Control and Design of PMBLDC Motor Using NSGA-II Multi-objective Algorithms, Int. Journal of Sci.: Basic and Applied Res. 14 (2) (2014) 220-234

[25] J. Tanomaru, Staff Scheduling by a Genetic Algorithm with Heuristic Operators, in: International Conference on Evolutionary Computation, 1995, pp. 456-461.

[26] A. Tiwari, K Vergidis, B. Majeed, Evolutionary Multi-Objective Optimization of Business Processes, IEEE Congress on Evolutionary Computation, 2006, pp. 3091-3097.

[27] O.Turchyn, Comparative Analysis of Metaheuristics Solving Combinatorial Optimization Problems, in: 9th Int. Conf. on the Experience of Designing and Appl. of CAD Syst. in Microelectronics, 2007, pp. 276-277.

[28] M. Ulbricht, Single-Objective vs. Multi-Objective scheduling Algorithms for Scheduling Jobs in Grid Environments, in: 10th Int. Symposium on Applied Machine Intelligence and Informatics, 2012, pp. 411-414.

[29] J. Yoon, S.Cho An Efficient Genetic Algorithm with Fuzzy C-Means Clustering for Travelling Salesman Problem, IEEE Congress on Evolutionary Computation, 2011, pp. 1452-1456.

[30] J. Zhang, W. Wang. X, Xu, J. Jie, A Multi-objective Particle Swarm Optimization for Dual-Resource Constrained Shop Scheduling with Resource Flexibility, IEEE Symposium on Computational Intelligence for Engineering Solutions, 2013 pp. 29-34.

[31] Y. Zhang, Research on Human Resource Allocation Optimization Based on Genetic Algorithm from the Perspective of Two-way Choice Model, Int. Conf. on Educational and Inform. Technol., 2010, pp. 380-383.

[32] H. Zhaodong, C. Wenbing, Xiao Yiyong, L Rui, Optimizing Human Resources Allocation on Aircraft Maintenance with Predefined Sequence, in: 2010 International Conference on Logistics Systems and Intelligent Management, 2010, pp. 1018-1022.