

A cloud computing based many objective type-2 fuzzy logic system for mobile field workforce area optimization

Andrew Starkey¹ · Hani Hagra¹ · Sid Shakya² · Gilbert Owusu² · Ahmed Mohamed² · Daniyal Alghazzawi³

Received: 22 April 2016 / Accepted: 28 July 2016 / Published online: 4 August 2016
© Springer-Verlag Berlin Heidelberg 2016

Abstract Large scale optimization problems in the real world are often very complex and require multiple objectives to be satisfied. This applies to industries that employ a large mobile field workforce. Sub-optimal allocation of tasks to engineers in this workforce can lead to poor customer service, higher travel costs and higher CO₂ emissions. One solution is to create optimal working areas, which are geographical regions containing many task locations, where the engineers can work. Finding the optimal design of these working areas as well as assigning the correct engineers to them is known as workforce optimization and is a very complex problem, especially when scaled up over large areas. As a result of the vast search space, given by this problem, meta heuristics like genetic algorithms and multi-objective genetic algorithms, are used to find solutions to the problem in reasonable time. However, the hardware these algorithms run on can play a big part in their day-to-day use. This is because the environment in which the engineers are working within is changing on a daily bases. This means that there are severe

time-restrictions on the optimization process if the working areas were to be optimized every day. One way to tackle this is to move the optimization system to the cloud where the computing resources available are often far greater than personal desktops or laptops. This paper presents our proposed cloud based many objective type-2 fuzzy logic system for mobile field workforce area optimization. The proposed system showed that utilizing cloud computing with multi-threading capabilities significantly reduce the optimization time allowing greater population sizes, which led to improved working area designs to satisfy the faced objectives.

Keywords Type-2 fuzzy logic · Genetic algorithms · Cloud computing · Multi-objective · Optimization

1 Introduction

In large companies that have mobile workforces such as field engineers, the efficient allocation of tasks is critical. If task allocation is sub optimal this can lead to increased costs, lower revenue, lower customer satisfaction and lower employee satisfaction. This is particularly true in large utilities companies, as utilities companies often cover large geographical areas due to the natural monopolistic nature of the infrastructure.

For the purpose of management, all engineers (where there could be thousands) cannot be sent to any work location (where there could be tens of thousands of locations). As a result, working areas need to be set up so that teams of engineers can service the tasks within their respective working areas. This leads to familiarity and a sense of ownership of the local area.

This produces one of the primary optimization problems; the design of these working areas. These working areas are

✉ Hani Hagra
hani@essex.ac.uk

Andrew Starkey
astark@essex.ac.uk

Sid Shakya
sid.shakya@bt.com

Daniyal Alghazzawi
dghazzawi@kau.edu.sa

¹ The Computational Intelligence Centre, School of Computer Science and Electronic Engineering, University of Essex, Colchester, United Kingdom

² Business Modelling and Operational Transformation Practice, British Telecom, London, United Kingdom

³ Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

geographical areas and form the boundaries in which the engineering teams should work within. Each of the working areas is made up of a collection of service delivery points (SDPs) where these SDPs connect to local properties (household and commercial). Each SDP contains tasks for the engineers to complete. The SDPs will contain different amounts and different types of work depending on where the SDP is located.

There are a number of potential benefits from optimizing the workforce of a large mobile workforce such as increasing the amount of work completed. This leads to all tasks being completed sooner, increasing customer satisfaction and increasing the capacity of the workforce.

More benefits include lower fuel costs because the workforce is travelling less, which then leads to less CO₂ emissions, reducing the organizations ecological impact. Lastly in times of unexpected demand (such as extreme weather conditions) the organization has the ability to utilize the workforce as best as possible, reducing the stress put onto the field workforce where repairing lost utilities is seen as urgent.

The large list of potential benefits of workforce optimization have led to a number of methodologies being investigated to solve the problem. Given the large search space created by this type of real world problem, meta-heuristics such as genetic algorithms (GAs) and multi-objective genetic algorithms (MOGAs) have been used [1].

Previous work has looked into the use of genetic algorithms in real world optimization problems and how multi-objective algorithms have been used to help produce solutions, due to the multi-objective nature of the problem [2,3]. Additionally previous work has looked into cloud based optimization [4].

One recent development is the exploration of the many-objective problem associated with multi-objective algorithms. This is applicable to our work given we have five objectives.

In addition to using meta-heuristics to search for good solutions we have developed fuzzy logic systems to help deal with uncertainty when building the individual solutions. Fuzzy systems have proven their capabilities when it comes to dealing with uncertainty and a method of dealing with uncertainty is needed when dealing with real-world problems. We extended this to type-2 fuzzy logic systems as type-1 fuzzy systems cannot fully handle all levels of uncertainty in the changing environment of this problem.

One area that has yet to be explored fully is the hardware capabilities and the impact this has on the frequent optimization process. This is especially applicable to real-world problems where time can be a limiting factor for the optimization algorithm. This differs from lab experiments where time is less of a factor and algorithms are given the ability to run as long as necessary to get the best results.

Taking the optimization software off the laptops and putting it in the cloud allows the software more brute-force power, but there are drawbacks with using the cloud such as security, but we will discuss these issues.

This paper aims to address the problem of workforce optimization where the paper presents our proposed cloud based many objective type-2 fuzzy logic system for mobile field workforce area optimization. The paper contributions could be summarized as follows:

- Presenting a novel cloud computing based many objective type-2 fuzzy logic system for mobile field workforce area optimization. The proposed system shows the benefits of genetically optimizing both the membership functions and footprints of uncertainty of type-2 fuzzy logic systems to satisfy the multiple objective in workforce area optimization.
- The proposed system showed that utilizing cloud computing with multi-threading capabilities significantly reduce the optimization time allowing greater population sizes, which led to improved working area designs to satisfy the faced objectives. These improvements allow the system to be effectively used on a daily bases.
- The proposed system incorporates a new distance metric to help overcoming selection pressure issues associated with many objective problems.

The rest of the paper is organized as follows: Sect. 2 presents an overview on the related work and on the work area optimization problem. In Sect. 3, we provide a brief overview of type-2 fuzzy systems, genetic algorithms, multi-objective genetic algorithms and many-objective problems. Section 4 presents an explanation of the security risks of using cloud based resources. Section 5 presents the proposed cloud based many objective type-2 fuzzy logic system for mobile field workforce area optimization. Section 6 reports on the experiments carried out and results obtained. Finally Sect. 7 presents the conclusions and future work.

2 Related work and problem overview

The cloud is now a well-established and important part of business information systems. Utilizing the cloud for resource optimization purposes has been explored before as in [3,4]. These works highlighted the need for parallel processing, and the benefits of extra computational power typically available in cloud based servers.

Genetic algorithms (GAs) are a well-used meta-heuristic in complex large scale optimization problems [2,3]. 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 that

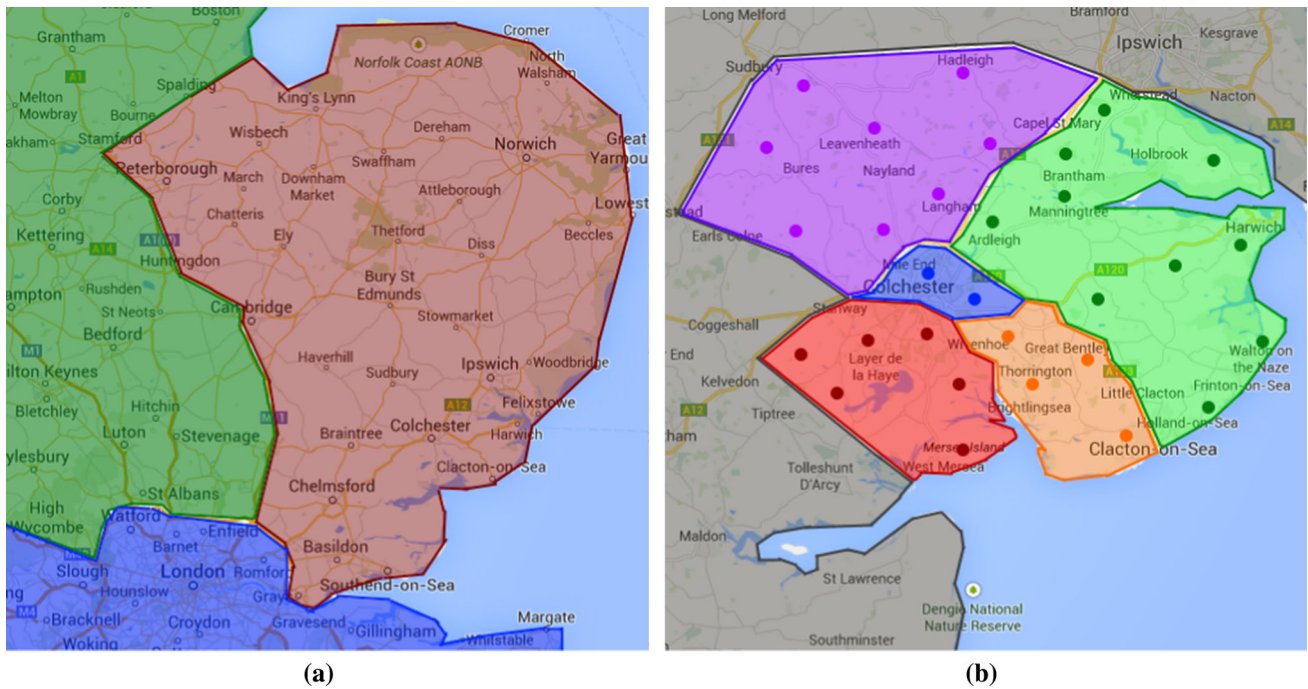


Fig. 1 a Regional areas. b WAs within a sub-region

is representative of the problem it is attempting to solve. For workforce optimization this would be a simulation of how effectively tasks would be completed given the solution generated by the optimization. This would require calculating the paths engineers would take to complete tasks so their estimated travel distance and time can be calculated. This is a type of vehicle routing problem and is a generalized version of the travelling salesman problem (TSP) [5].

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). More information on both GAs and MOGAs can be found in Sects. 3.1 and 3.2.

2.1 Work area optimization

The current problem concerns the optimization of the working areas in which the engineers are assigned to. 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, which are made up of a collection of service delivery points (SDPs). These SDPs serve domestic and commercial properties by connecting them to services such as electricity, gas, water or telecoms, depending on the service the organization provides. These SDPs contain tasks for the engineers to

complete. Figure 1a illustrates how part of the UK might be subdivided into regions.

Figure 1b shows a sub region which is divided into five working areas (WAs). The SDPs are shown as dots in each of the working areas. It’s the grouping of these SDPs that can change when finding the most optimal working areas. The primary 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 Eq. (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 Eq. (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.

$$\sum E_{id} / \sum E \quad (2)$$

- **Maximize utilization** Utilization is the percentage of time an engineer is completing tasks. Unutilized time is when the engineer is idle or travelling. In Eq. (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.

$$\left(\sum E_{cw} / E_{at} \right) / \sum E \quad (3)$$

- **Minimize area imbalance** working areas should be evenly balanced with the amount of work they contain. This means there will be smaller working areas for urban locations and larger ones for rural locations. Balancing is measured in hours and is represented in Eq. (4). It is the difference between the largest (WA_L) and smallest (WA_S) working areas in terms of hours of work.

$$WA_L - WA_S \quad (4)$$

- **Minimize team imbalance** The teams assigned to each working area should also be as balanced as possible. Team balancing is measured in engineers and is represented in Eq. (5). It is the difference between the largest (T_L) and smallest (T_S) teams in number of engineers.

$$T_L - T_S \quad (5)$$

There are a number of constraints that need to be factored in when evaluating the working area designs. 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.

3 A brief overview on genetic algorithms and type-2 fuzzy logic systems

This section will present a brief overview of genetic algorithms, from single objective to multi-objective and will outline the issues with many-objective problems. We will also give an overview on fuzzy systems from type-1 and interval type-2 fuzzy systems.

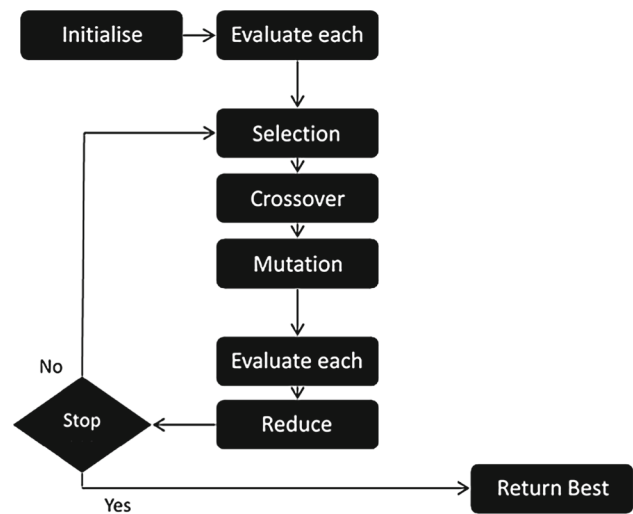


Fig. 2 GA flow

3.1 Genetic algorithms

Genetic algorithms (GAs) are a population based meta-heuristic. The individuals in the population are potential solutions to the problem and each solution is evaluated to assess which are the stronger solutions. The population will then go through a process of evolution so that the solutions that have characteristics most suited to the problem are more likely to pass these characteristics on to the next generation of solutions.

When the next generation of the species is created, genes from two parent individuals will be combined to determine the characteristics of the child [6]. Figure 2 shows the process of a standard GA. The first step is to initialize the population. This is done by creating N solutions and randomly assigning the genes of each solution.

These genes are then evaluated and 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. 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). Crossover will generate two 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.

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

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. If the criteria is met the best (most fit) solution will be returned, else the GA will loop back round for another generation.

If there is more than one objective to be optimized, then the objective values need to be combined using a function to give a single fitness value. 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 (CO₂) 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 for this is shown in Eq. (6).

$$Fitness = \frac{A \cdot B}{C} = 2.50 \tag{6}$$

Table 1 shows some possible solutions that could come from the fitness function given in Eq. (6). 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 also 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

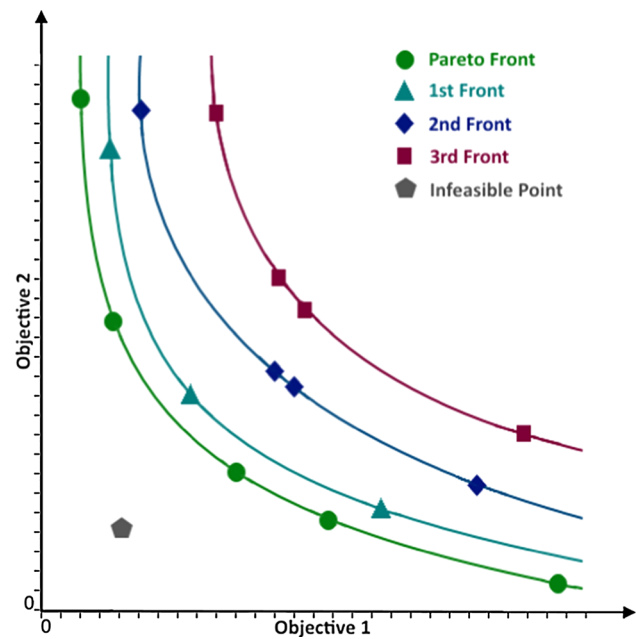


Fig. 3 Fronts in multi-objective results

example illustrates why problems with multiple objectives should use a multi-objective GA.

3.2 Multi-objective genetic algorithms

Multi-objective genetic algorithms (MOGAs) compare the results of each objective independently. One way of doing this is by ‘Domination’ as is done in the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) [7]. 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. Figure 3 illustrates what the fronts may look like in a MOGA with two minimization objectives. Figure 3

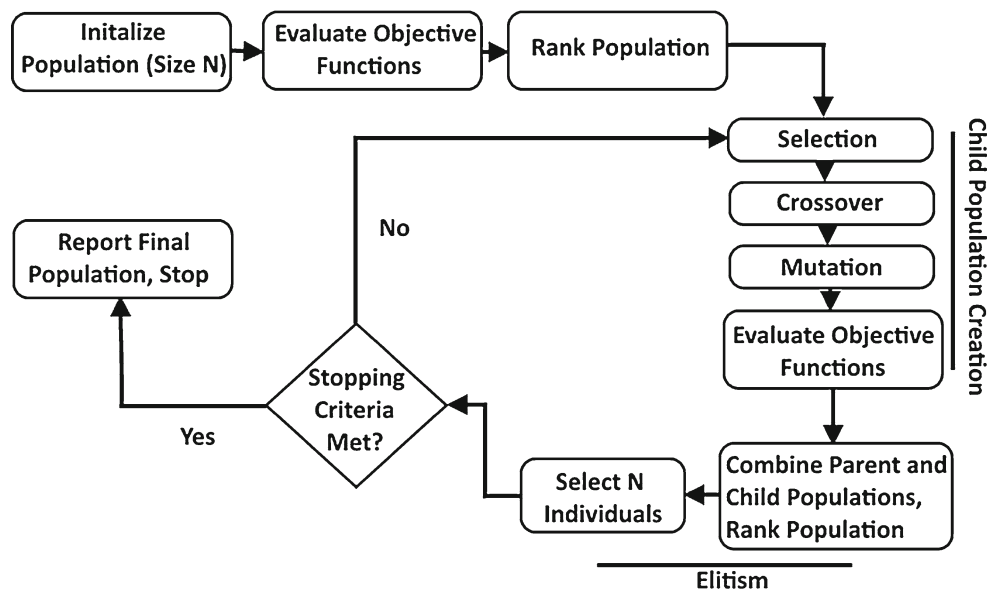


Fig. 4 Multi-objective flow diagram

also shows an infeasible point which is a point that is not possible given the constraints of the objectives.¹

The process of a multi-objective genetic algorithm, specifically NSGA-II, is laid out in Fig. 4. 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.

Once there are N members in the new population, each one will be evaluated in each objective, unlike the single objective GA where each objective value is 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 Fig. 3).

Once the new population has been ranked the process then moves on to the evolution stage. This consists of three main steps; Selection, Crossover and Mutation. 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, the comparison is down to the rank (or domination count) of the individuals. 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 we can randomly select between the 2, use crowding distance or

some other metric to determine which solution should win the tournament.

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 5 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.

The population of size $2N$ cannot be taken through to the next generation, else the population would grow exponentially. 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 .

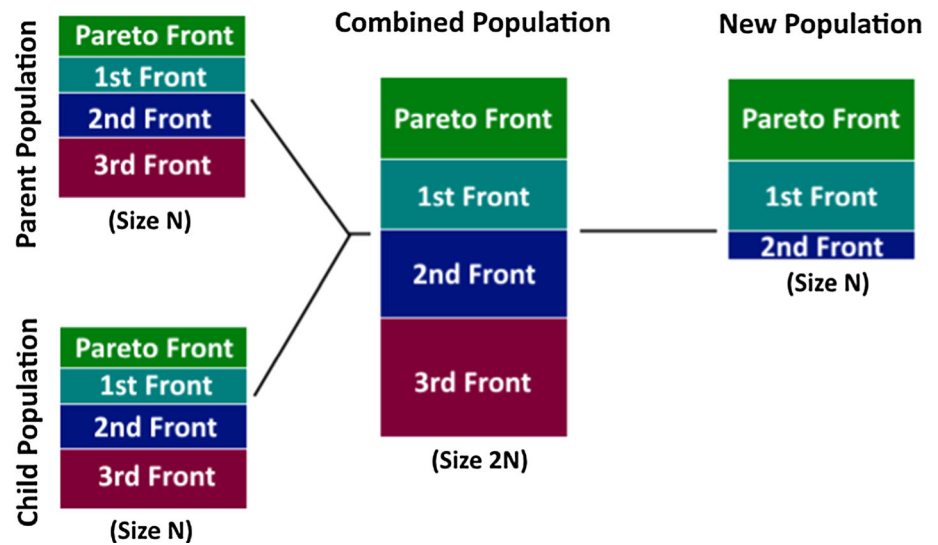
3.3 Many-objective problems

As the number of objectives in a problem grow the usefulness of the Pareto front decreases. Particularly in two ways. The first is how it chooses solutions as parents (selection pressure) and secondly, the way it presents its final set of optimal solutions. The more objectives there are the more difficult it is to visualize too [8], but this cannot be avoided [9].

Problems that contain more than three objectives are known as many-objective problems (MaOP). To handle

¹ http://www2.mae.ufl.edu/hafitka/stropt/Lectures/multi_objective_GA.pdf. [Last Accessed: 1st April 2016].

Fig. 5 Creating a new population from the combined population



MaOPs more successfully, a MOGA needs to be modified into a many-objective genetic algorithm (MaOGA). This applies to NSGA-II where it has been stated that NSGA-II is not so good at handling MaOPs, because of its Pareto based nature and there have been many example of work trying to solve this important problem [9–11]. One way in which we attempt to resolve some of these issues associated with the Pareto front is a distance metric and is described in Sect. 5.2. The method presented in [12] is similar to what we will present for a new distance metric, comparing new solutions to a current solution to compare the differences between all new solutions. In [12] a point in the search space is chosen to compare each solutions values it, which may not be optimal. Also, [12] still requires the user to put in weights to find a solution that they think might be a good set up. This leads to the additional optimization problem of finding the best weights to select for each situation. Other examples of placing a point on the search space to compare the generated solutions against include [9, 13].

3.4 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. Once the type-1 membership functions have been chosen, all the uncertainty disappears, because type-1 membership functions are totally precise [14, 15].

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]. 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 [14, 15]. As shown in Fig. 6a, the interval type-2 (IT2) fuzzy set \tilde{A} can be represented in terms of the upper membership function (UMF) (denoted by $\bar{\mu}_{\tilde{A}}(x), \forall x \in X$) and the lower membership function (LMF) (denoted by $\underline{\mu}_{\tilde{A}}(x), \forall x \in X$) as follows:

$$\tilde{A} = \int_{x \in X} \left[\int_{u \in [\underline{\mu}_{\tilde{A}}(x), \bar{\mu}_{\tilde{A}}(x)]} 1/u \right] / x \tag{7}$$

The UMF and LMF are bounds for the Footprint of Uncertainty ($FOU(\tilde{A})$) of an IT2 fuzzy set \tilde{A} . As shown in Fig. 6b, in an IT2 fuzzy set the secondary membership function

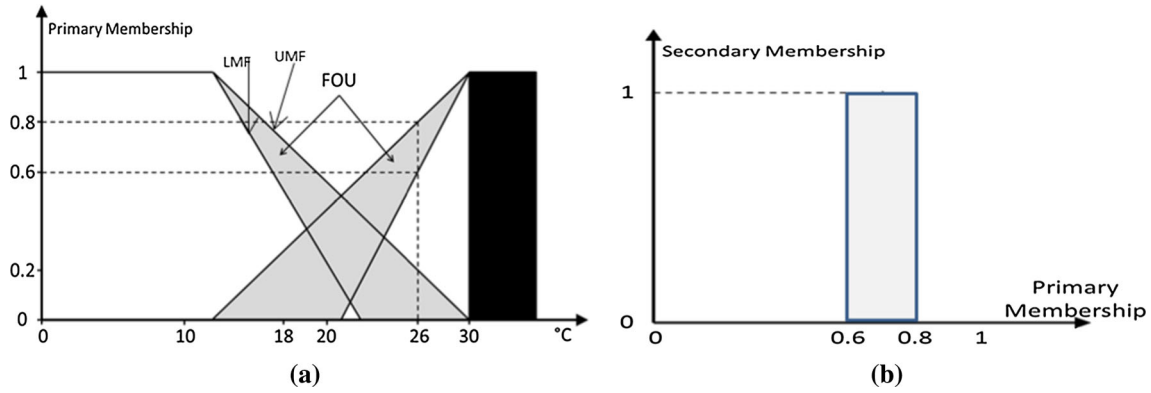
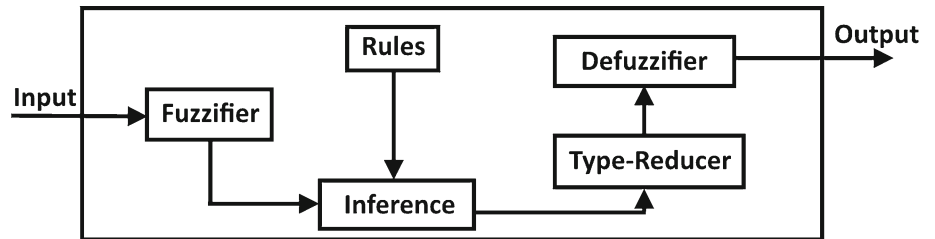


Fig. 6 a A interval type-2 fuzzy set- primary membership function. b An interval type-2 fuzzy set secondary MF at a specific point x'

Fig. 7 Type 2 FLS [18]



is equal to 1 for all the points in the primary membership for $\forall x \in X$.

Figure 7 shows an overview of 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 Eq. (8), 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 [14].

$$Y_{cos}(x)_k = [y_lk, y_rk] = \int_{y_k^1 \in [y_{lk}^1, y_{rk}^1]} \dots \int_{y_k^M \in [y_{lk}^M, y_{rk}^M]} \int_{f^1 \in [f^1, f^{-1}]} \dots \int_{f^M \in [f^M, \bar{f}^M]} \frac{1 \cdot \sum_{i=1}^M f^i y_k^i}{\sum_{i=1}^M f^i} \quad (8)$$

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. More information regarding the interval type-2 FLS and its applications can be found in [16–19].

4 Cloud computing security issues

Cloud computing is generally understood as computing in grids and as a utility where the storage and computing are provided as a service through the cloud via virtualization, which means that the client remotely uses a service delivered by a provider commonly termed to be “in the cloud”.

There are various security issues facing cloud computing applications where the critics of cloud computing contend that it is not a secure mechanism as the data is no longer present within the safe confines of the organization local area network (LAN). However, statistics show that 1/3 of security breaches result from stolen or lost laptops and other devices and from employees accidentally exposing data on the Internet, with nearly 16 % of security breaches due to insider theft.² Hence, cloud computing can help to address some of these issues where Google apply patches more quickly than most enterprises to plug holes in software. In addition, the Google Apps Premier edition offers the ability to protect data in transit by encrypting it in the pipeline between Google and the user’s desktop, as well as offer control over who can access the data.

In cloud computing based applications, the security of the data is dependent on the level of commitment the vendor has in enforcing security policies and verification done by third party. The verification is being offered to the clients

² <http://www.cnet.com/uk/news/cloud-computing-security-forecast-clear-skies/> [Last Accessed: 1st April 2016].

as an on-demand service by vendors such as Google, Salesforce and Amazon etc. Cloud service providers are obliged to offer evidence of security certifications, and are subject to inspections by auditors thus placing them under much higher scrutiny than typical in-house security teams.

The recent years have witnessed an increasing number of small companies moving towards cloud computing solutions due to their cost and performance gains and to have suitable protection against cyber-attacks due where Loopholes in the programming make the firewall ports vulnerable. However, there is a need to provide effective security policies to sustain the cloud computing based solutions.

4.1 Cloud computing security challenges

The term security in cloud computing encompasses various factors like confidentiality, integrity & availability.³ Confidentiality means the ability to decide who has the responsibility of storing of encryption keys used to encrypt the data of the user (the encrypted data stored with the vendor must be kept secured from the employees of the vendor). Integrity means allowing a common policy for approved data exchange. Availability deals with the availability of the service to eliminate any “denial of service”. One more thing to consider is that contract policy exists between the client and the vendor implying that the data would belong only to the client and third party may not get involved at any manner at any point of time.

There are various security challenges which need to be addressed for cloud computing based solutions as follows:

- User access privileges—transmission on the internet is prone to be attacked and is susceptible to risk owing to the issue of ownership of data. Thus companies should have ample knowledge of the policies of the provider.
- Compliance to regulations—the security of the solution and its accountability finally rests with the client as they have the option to choose between vendors who agree to be audited by 3rd parties for security and those who do not.
- Location of data—some of the companies do not even have the knowledge of the location of their data, based on the contract type.
- Segregation of data—mechanism for segregation of data need to be in place where in some cases encrypted data for different clients is stored on the same storage.
- Recovery—there should be a disaster recovery mechanism deployed by the vendor for protecting the user’s data.

³ <http://cloudsecurity.org/> [Last Accessed: 1st April 2016].

4.2 Various tools to handle security challenges in cloud computing

New developments are continuously being added into the cloud computing technology with the aim of enhancing security. One such product is QualysGuard.⁴ It is meant for discovering the loopholes in a network. It has acquired considerable reputation after being used by almost 200 of the Forbes Global 2000 companies. The product functions by placing an apparatus behind the Firewalls where it monitors the network for any breaches or threats. All data is encrypted by the device and it has no access to the client’s data, with just access to only specified IPs and the administrator for modification of script & credentials. The product provides a new mode of security where any possible threats or attacks are monitored and dealt with by a 3rd party. In case of an unauthorized access attempt or an attack on any of the services rendered by the cloud, the solution cuts off access to the service from the source of the threat and prevents it from affecting the service availability.

Hashes and checksum are also provided for each file stored on the cloud by the vendors to evade the requirement of encryption of the files on the client side. Another feature available is the Password Assurance Testing which utilizes the processing capabilities available in the cloud to track and monitor the attempts made to break into the system by using password breaking/guessing.

The open security architecture (OSA) provides a free framework capable of being integrated easily into any application.⁵ The framework has its basis on plans that depict the flow of information for specific implementations, along with the policy being implemented in every phase for security reasons. When access to certain resource is required by end user in the cloud, he must be conscious of the access agreement like the acceptable method of use/conflict of interests. Here, the commitment to the policy may be confirmed through the signature of the end user. All code and protocol on the client side devices such as the Server, Firewall or Mobile device, susceptible to attack must be detected and patch uploaded to secure the local machine immediately upon detection. This ensures similar security for the cloud and the client. However, in case of the cloud there is a need of additional security from a user who tries to access the cloud with ulterior motives. Thus the denial of service (DOS) protection should be included in the cloud. Once the security is

⁴ <https://www.qualys.com/enterprises/qualysguard/> [Last Accessed: 1st April 2016].

⁵ <http://www.opensecurityarchitecture.org/cms/> [Last Accessed: 1st April 2016].

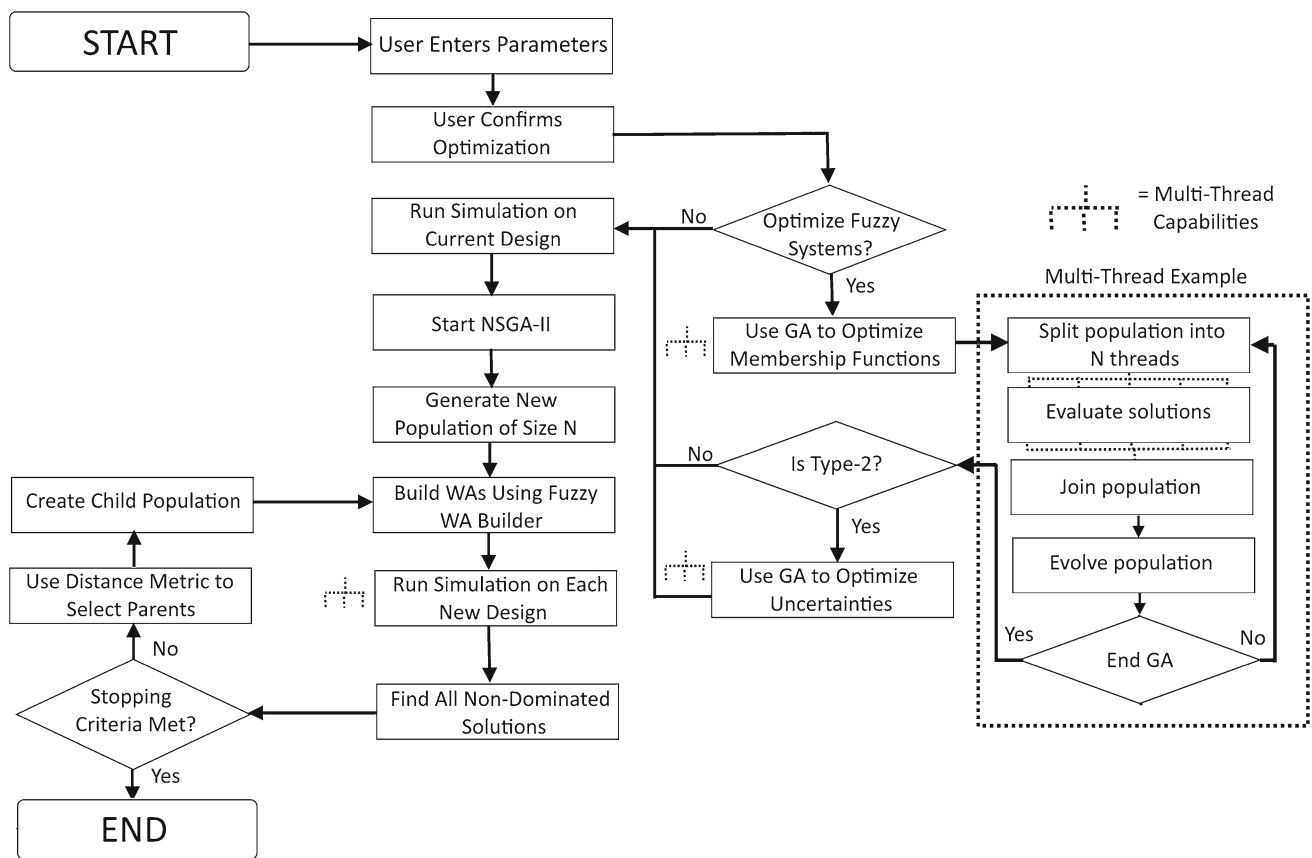


Fig. 8 Proposed cloud based many-objective type-2 fuzzy logic based mobile field workforce area optimization system

catered to at the cloud level then we come across the issue of integrity of transmission between the cloud and the client. SSL/TSL (secure socket layer/transport layer security) protocols can be utilized as one of the measures to ensure integrity of the transmission and preventing any attacks in the middle. IPsec (secure internet protocol) can also alternatively be used to secure the network at the lower level. There is a need also to ensure that there is no eavesdropping on the session.

The security level of the cloud implementation need to be audited by a 3rd party auditor and this practice is supported both by the client companies as well as the Cloud vendors. Accreditation obtained by the vendor provides them with the competitive edge and proves their level of commitment towards security.

Another lever of security in cloud based applications can be supplied via demilitarized zone (DMZ) approach of boundary protection in data centers to ensure secure interaction among the various constituent entities like firewall, router gateways, storage & proxy server. Information that is most critical is stored on the other side of the demilitarized zone.

5 The proposed cloud computing based many-objective type-2 fuzzy logic based mobile field workforce area optimization system

The proposed system used to solve the work area optimization problem is given in Fig. 8. It has multiple genetic algorithms utilized within it. The first is used to optimize the membership functions of the fuzzy sets. The second is used to optimize the footprint of uncertainty (FOU) for each type-2 membership function. Then finally, the third is the main multi-objective genetic algorithm, NSGA-II, which is used to optimize the working areas.

Currently the system requires the user to enter parameters, such as the number of working areas to optimize for and the various GA parameters (number of generations, population size etc.). Once the user then confirms the settings and starts the optimization process the system will check if it should optimize the fuzzy systems that will be used. If yes, the system will use a GA to optimize the membership functions of the fuzzy systems that will be used in the main optimization. If the system has been selected to use type-2 fuzzy systems, it will then proceed to optimize the FOU of

each membership function. More details about this can be found in Sect. 5.1).

As the optimization of the fuzzy systems use genetic algorithms then the system will try to utilize the CPU resources it has available based on the current parameters. For each of the GAs used in the proposed system, multiple threads will be created at the point each solution in the population is about to be evaluated. In this way the solutions can be evaluated in parallel and this will have the potential to decrease the optimization time [20]. This is where multiple threads are best placed because the evaluation of each solution takes the most time, and does not require all the other solutions to be available, such as in selection and crossover.

Once the fuzzy systems have been optimized then system will run the simulation on the current design to get the objective values of the design currently being utilized by the mobile workforce. The simulation uses a fuzzy system to help decide which engineers should travel to each job. This simulation will give the current coverage, travel and utilization values. The balance of the working areas and teams is a trivial calculation based on the difference between the largest and smallest working areas and the largest and smallest teams.

Once the current design has been evaluated the NSGA-II will start the optimization process. It will create a population of solutions and evaluate each one, giving each solution a value based on the proposed distance metric (see Sect. 5.2). Again at this point multiple threads will be created and the population will fork into these threads, splitting the population evenly between the threads. Once all solutions have been evaluated the population will join back up again allowing the NSGA-II to operate as normal and start calculating the dominance of each solution and creating the fronts. Because of the issues we have outlined with all solutions ending up on the Pareto front for many objective problems, we use the distance metric to help with parent selection.

If the stopping criteria for the algorithm is met then the Pareto front of solutions will be presented to the user with the solution that has the highest distance value being highlighted as the best result. For more in-depth information on how the fuzzy systems operate see our previous work [1]. This includes the diagrams for the fuzzy sets and the rule base it operates with.

5.1 Genetically optimized fuzzy systems

Fuzzy logic systems (FLSs) have been shown to handle imprecisions and uncertainties within an environment. The majority of FLSs are type-1 based and therefore cannot fully handle the imprecisions and uncertainties presented by dynamic environments where as type-2 systems have demonstrated they can outperform type-1 systems in these environments [15, 16, 18].



Fig. 9 Real-value chromosome for the parameters of two type-1 fuzzy sets membership functions

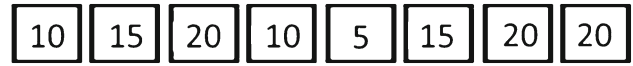


Fig. 10 Real-value chromosome for percentage uncertainty associated with the type-2 fuzzy sets

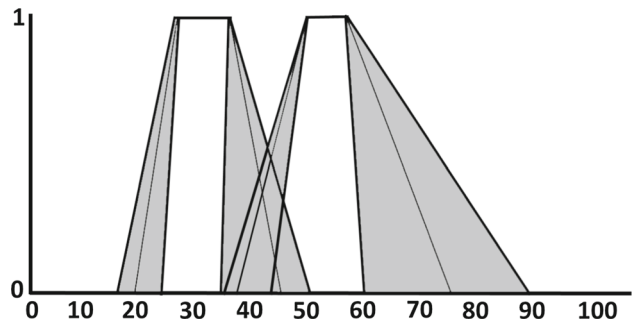


Fig. 11 Resulting type-2 membership functions from chromosomes

Additionally when some fuzzy systems are created their membership functions are generated by a human expert. These membership functions could then be sub-optimal and therefore need to be tuned to perform well in a changing environment. When a type-2 systems is used the uncertainty also needs to be calibrated to suit the environment the FLS will be used on.

As the proposed system will be used in multiple problem environments its membership functions cannot be tuned offline because it is unknown which set of working areas the user will be optimizing. Therefore in our proposed system the membership functions and uncertainties will be tuned using a Real Valued GA at the start of each optimization process. The genes of each solution will represent the points each membership function has along the x-axis.

Figure 9 shows an example of a chromosome for the parameters of the membership functions of two type-1 fuzzy sets. Each membership function will have four points associated with it giving a total of eight genes. The first four values are for the first membership function parameters and the last four values are for the second membership function parameters.

Figure 10 shows an example of a chromosome for the uncertainty associated with type-1 fuzzy sets. Each gene represents the uncertainty percentage associated with the base values of the type-1 fuzzy sets to result in the upper and lower membership functions of the type-2 fuzzy sets.

Figure 11 shows the resulting type-2 fuzzy set, given from the genes in Figs. 9 and 10. This GA will evaluate the fuzzy systems on their primary purpose. So for the working area builder FLS it will evaluate how much the proposed mem-

bership function improves on the balancing objectives. For the task allocation FLS, the system will evaluate how much improvement there is to the coverage to travel ratio. For the type-2 fuzzy systems the uncertainty tuning happens after the membership function tuning has taken place.

5.2 Many-objective distance metric

The proposed distance metric is used to help the parent selection process and suggest the best result to the user. The distance metric for our given objectives is shown by Eq. (9).

$$DIST = \left(\frac{C_s - C_o}{C_o} - \frac{T_s - T_o}{T_o} + \frac{U_s - U_o}{U_o} - \frac{AB_s - AB_o}{AB_o} - \frac{TB_s - TB_o}{TB_o} \right) \quad (9)$$

$$DIST = \frac{New - Original}{Original} \quad (10)$$

In Eq. (9) we have the coverage given by the new solution (C_s) and the coverage given by the original (C_o). The travel value given by the new solution (T_s) and the travel value given by the original (T_o). The utilization given by the new solution (U_s) and the utilization given by the original (U_o). The area balance given by the new solution (AB_s) and the area balance given by the original (AB_o). Finally there is the team balance given by the new solution (TB_s) and the team balance given by the original (TB_o).

Each objective in Eq. (9) calculates the distance using Eq. (10). The new value for that objective is subtracted by the original value for that objective, divided by the original value. Giving the distance as a percentage.

Coverage and utilization are both maximization objectives and add to the distance value. The remaining objectives are minimization objectives so they subtract from the distance value. This is, for example, if the travel value in the new solution is lower than the original, it will give a negative distance for that objective, this is what we want, and so subtracting this negative value increases our overall distance value, giving us indication that this solution is stronger.

It is worth noting that for this distance metric to work there needs to be original values. This is because we want something to be optimized, if we don't have a base to compare it to we don't know if we have improved over the currently implemented solution.

This metric could be used if there is no original results, for example creating new working areas where there were not any before. However this would require the new working areas to be designed by an expert or by a GA process that does not use the distance metric. Such as in our previous study [21]. Once these "original" results have been created then this proposed system could be used to improve on the results.

Table 2 Hardware comparison

Hardware comparison	Laptop	Cloud
CPU clock speed (GHz)	2×1.9	8×2.7
CPU threads	4	16
RAM (GB)	4	32

Table 3 Current area objective values

Coverage	Travel	Utilization	Patch balance	Team balance
74.60	7.00	74.03	428.74	71.00

6 Experiments and results

Our first set of experiments are the comparison of the un-optimized and the GA optimized fuzzy systems. These results include a comparison of both genetically optimized type-1 fuzzy systems and type-2 fuzzy systems. We will then present our qualitative results from these experiments showing the logical differences when comparing the designed working areas. We will then look at the benefits brought to this work by multi-threaded cloud computing and the first experiments for this will consist of evaluating the time difference of running the system on the current hardware compare to running the system in the cloud. These tests include splitting the population into multiple threads as well as comparing just the single thread option.

Once the time factor had been evaluated we looked at comparing the difference that increasing the overall population size would have on the optimization process, both in how it affected the overall results and how long the process would take. Thus we could then analyze how much benefit we can get from moving the optimization system into the cloud. The model of CPU in the laptop is an Intel Core i5-4300U. Whereas the model of CPU in the Cloud is stated to be an Intel Xeon E5-2680. A comparison of the specification of the laptop and the cloud is given in Table 2. Where we can clearly see the cloud has much more resources available. The use of the cloud helps solve the problem of resource scarcity with personal devices such as laptops [22,23].

6.1 Comparison of genetically optimized fuzzy systems

We selected an area in the utilities company that needed optimizing. The values for each of the objectives of the current design for this area can be found in Table 3. The current designs for these active working areas were created by experts who have local knowledge about the area. Much of the design was done by hand, using maps and sheets of data. As a result, although these working areas were being created by an expert, the designs took weeks to months to final-

ize. Our proposed system can create working area designs in minutes.

For the selected area we ran the system with the genetic optimization of the fuzzy systems turned off and selected type-1 systems. We repeated the optimization of this area with these settings 10 times. For each run we took the best result, based on the distance metric, and placed that result in Table 4, with the average (Avg.) value for each objective listed in the second to last row and the standard deviation (SD) for each of the objectives found in the last row.

We can see that the distance value, used on the average of the 10 runs, gives the value 0.38 for the system with un-tuned type-1 systems. With a standard deviation of the distance of 0.07.

We then repeated the 10 runs again except this time we switch on the genetic tuning of the membership functions. The best solution, given by the distance metric, from each of these 10 runs can be seen in Table 5. The average distance value given by these 10 solutions is 0.41. An improvement on the un-tuned type 1 system of 7.89%.

We can also compare the standard deviation (SD) of the un-tuned and tuned systems. With the tuned system’s SD for the distance value improving by 0.02, or 28.57%.

We then switch from the type-1 system to the type-2 system and repeat the evaluation process. Again we run 10 times and take the best result based on the distance value for each run. We have put the results for the un-tuned type-2 systems in Table 6.

The un-tuned type-2 systems have the same membership functions as the un-tuned type-1 systems. However they also have 1% uncertainty applied to them. This is based from the previous study where about 1% uncertainly performed the best for this problem [1].

Table 4 Results from using Type-1 untuned fuzzy systems

	Coverage	Travel	Utilization	Balance	Team balance	Dist. val
	87.86	2.75	87.19	117.48	19.00	0.48
	87.20	3.33	86.53	302.15	30.00	0.35
	90.50	3.19	89.81	321.14	49.00	0.31
	94.46	2.92	93.74	253.84	38.00	0.40
	86.14	4.12	85.48	258.32	14.00	0.38
	91.10	4.55	90.41	278.15	26.00	0.36
	93.41	3.08	92.70	150.86	24.00	0.47
	90.19	4.04	89.50	221.18	20.00	0.41
	89.41	3.56	88.73	202.13	40.00	0.37
	84.72	3.16	84.07	362.07	53.00	0.25
Avg.	89.50	3.47	88.82	246.73	31.30	0.38
SD	3.09	0.59	3.07	75.73	13.19	0.07

Table 5 Results from using type-1 tuned fuzzy system

	Coverage	Travel	Utilization	Balance	Team balance	Dist. val
	89.93	3.44	89.24	203.74	26.00	0.42
	90.15	2.71	89.46	153.65	20.00	0.48
	86.46	3.25	85.80	247.37	18.00	0.40
	92.70	4.54	91.99	105.49	14.00	0.48
	85.28	3.55	84.63	267.11	20.00	0.37
	85.68	4.31	85.02	230.90	12.00	0.39
	85.10	3.57	84.44	233.91	10.00	0.42
	93.37	4.20	92.66	189.72	33.00	0.40
	88.31	3.11	87.64	237.09	14.00	0.43
	84.07	3.83	83.43	360.32	28.00	0.29
Avg.	88.11	3.65	87.43	222.93	19.50	0.41
SD	3.30	0.57	3.28	68.21	7.50	0.05

Table 6 Results from using type-2 untuned fuzzy systems

	Coverage	Travel	Utilization	Balance	Team balance	Dist. val
	90.74	2.45	90.04	147.40	28.00	0.47
	92.01	3.72	91.30	106.31	11.00	0.51
	84.31	4.83	83.66	223.52	11.00	0.38
	89.12	3.91	88.44	180.32	17.00	0.43
	83.42	2.92	82.78	261.61	16.00	0.40
	83.57	3.97	82.93	322.10	31.00	0.30
	91.24	3.73	90.54	230.78	38.00	0.37
	89.83	3.46	89.14	285.34	42.00	0.33
	91.12	2.30	90.42	247.30	46.00	0.38
	85.60	3.19	84.95	224.93	18.00	0.41
Avg.	88.10	3.45	87.42	222.96	25.80	0.40
SD	3.47	0.76	3.44	64.12	13.01	0.06

We can see from Table 6 that distance value based on the average of the ten solutions is better than the un-tuned type-1 system, 0.38 compared with 0.40. The type-2 un-tuned gives a 5.26% improvement over the type-1 un-tuned. This strengthens the case for type-2 systems being applied. However we can also see that the un-tuned type-2 systems performed slightly worse than the tuned type-1 system. This suggests that tuning a type-1 system takes some of the uncertainty out of the membership functions.

Finally we ran the tuned type-2 system 10 times. The tuning applied to the membership functions, just like the tuned type-1 systems. However, there was additional tuning of the footprint of uncertainty of each membership function. The results are presented in Table 7.

We can see the tuned type-2 performed better than the un-tuned type-2 by 10.00% and performed better than the tuned type-1 by 7.32%.

Table 7 Results from using type-2 tuned fuzzy systems

	Coverage	Travel	Utilization	Balance	Team balance	Dist. val
	90.72	3.25	90.03	155.48	24.00	0.45
	90.45	4.29	89.76	275.61	38.00	0.33
	92.91	3.59	92.20	131.11	8.00	0.51
	88.43	2.44	87.75	174.89	36.00	0.42
	91.05	2.68	90.35	190.63	30.00	0.44
	90.75	3.67	90.05	140.34	7.00	0.50
	87.44	3.14	86.77	206.36	31.00	0.40
	87.23	3.62	86.56	186.20	12.00	0.44
	92.19	4.50	91.48	128.64	15.00	0.46
	87.58	3.91	86.91	223.40	19.00	0.40
Avg.	89.88	3.51	89.19	181.26	22.00	0.44
SD	2.06	0.65	2.04	46.05	11.45	0.05

Additionally we can see that the tuned type-2 systems gave results with a smaller average standard deviation than the un-tuned type-2. A difference of 0.01 or 16.67 %.

From these results we can say that the best results, based on the distance value are those given by the genetically optimized type-2 system. All of the results, given by the proposed many-objective type-2 fuzzy logic based system, improve in all objectives when compared to the original. So any of these versions of the system could be used to satisfy all the objectives.

6.2 Qualitative analysis of designed working areas

We can compare the results given by the proposed system visually from a subjective view. Figure 12 shows the original design. Area 1 in Fig. 12 (labeled ‘1’) is a large urban area. As this large urban area is all in one working area it results in the large imbalance of the working areas shown in Table 3.

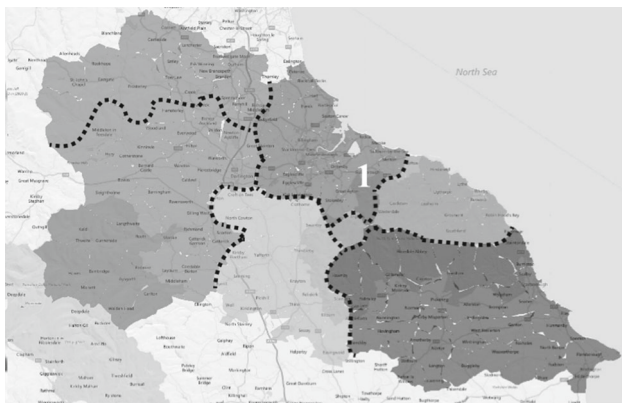


Fig. 12 Original WA design

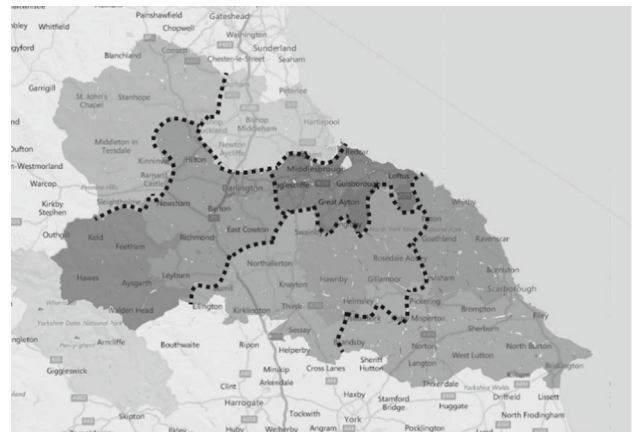


Fig. 13 A type-1 un-tuned solution



Fig. 14 A type-1 tuned solution

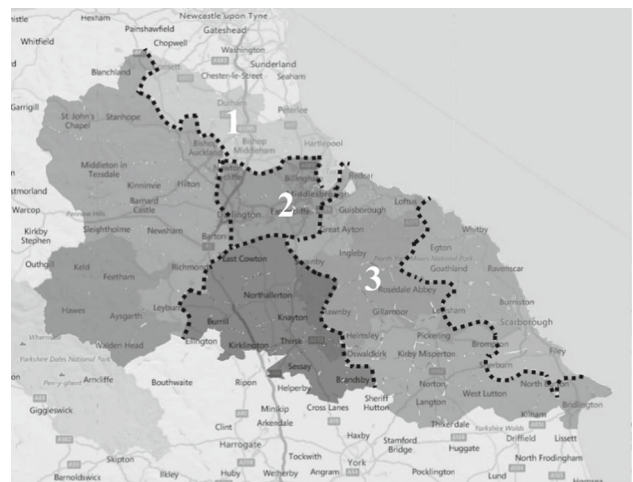


Fig. 15 A type-2 un-tuned solution

Figures 13–16 show a ‘best’ result from each of the tested fuzzy systems, optimized and un-optimized. Figure 13 shows the un-tuned type-1 system has split this large urban area into two working areas. Which is a good start as this much

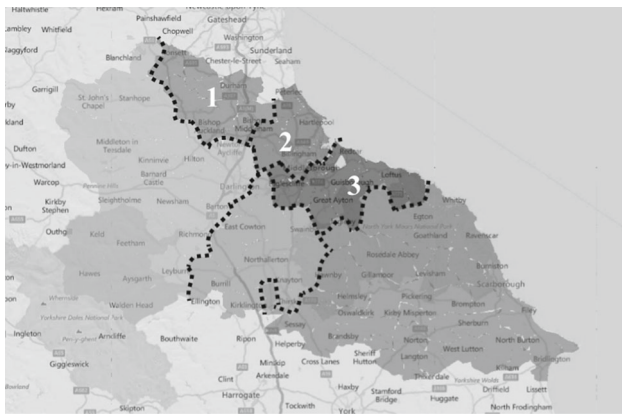


Fig. 16 A type-2 tuned solution

improves the area balance over the original. However it is likely three working areas are needed.

This is supported by the fact the Fig. 14 shows the tuned-type 1 and it splits the area into three working areas, improving the balancing. However because area 1 in Fig. 14 is so small and 2 is so large, in terms of geography, it impacts on the travel, and therefore coverage.

Figure 15 shows us an un-tuned type-2 result. It splits up the urban area into three working areas, which is good but area 3 extends far away from the urban area. Similar to Fig. 14. The similarities of Figs. 14 and 15 are backed up by the similar results of the type-1 tuned results and the type-2 un-tuned results in Tables 5 and 6.

Visually it is clear from Fig. 16 that the tuned type-2 result is more logical. The urban area is split into three equally sized working areas (1–3) with the rural WAs outside and much larger.

6.3 The speed of optimization results

Now that we have established that we want to use the type-2 tuned fuzzy logic version of the optimization system we can look at the potential benefits of utilizing cloud resources. Figure 17 shows the comparison of how long a GA (and MOGA) would take. This is important as if we want to use the type-2 tuned fuzzy systems we add two additional GAs into the optimization process. Figure 17 give us an indication on the level of improvement we would expect, before moving onto GAs with larger populations.

In Fig. 17, we can see that on the laptop for a population size of 100 and the old single threaded model it takes approximately 12 min to complete the optimization. However if we move the system into the cloud and run the same optimization with a population of 100 we can dramatically reduce the time taken to optimize, reducing the overall time by approximately 66.66 % to about 4 min. This is just due to the extra CPU resources available in the cloud. We can then increase

the population and measure the increase in time in the cloud. We run the optimization with a population of 200, given an average optimization time of 8 min. Then we double the population size again to 400 and the optimization takes 14 min. This tells us that we can quadruple the population size in the optimization on the cloud and it only takes 16.67 % longer.

However this is just the single threaded model. If we try to utilize as much CPU power as possible we can continue to decrease the optimization time. Just by increasing the number of threads to 2 we can reduce the time taken to optimize on the laptop by about 33.33 % to 8 min. But the reduction in optimization time is greater in the cloud as the optimization time is reduced to approximately 2 min 23 s. As a result we can say that by adding multi-threading capabilities to the system and moving the system into the cloud we can reduce the optimization time from approximately 12–2 minutes 23 s, give a reduction in time of about 9 min and 37 s, or about 80.14 %.

We can continue to further reduce the time by increasing the threads. Although there is evidence of diminishing returns having significant effect. Increasing the threads to 4 reduces the average time to 2 minutes 17 s. Increasing the number of threads to 8 reduces the time to approximately 2 min for a population size of 100. Giving a total reduction in time of 10 min, or 83.33 %

Due to this significant time reduction in the multi-threaded model we can then increase the population size as we did with the single threaded model. If we increase the population size to 200 we get times of 3 min 12 s for 2 threads, 4 min 21 s for 4 threads and 3 min 42 s for 8 threads. The minor fluctuation in time, can be attributed to a few causes. It could be that there were different number of processes taking place in the cloud at the time of optimization, thus affecting the time to optimize. This is one of the minor drawbacks, as you may not have total control over the available resources in the cloud. Additionally it could be that there needs to be a minimum number of solutions per thread to have practical benefit. For example if a population of 200 is split into 8 threads then that's only 25 solutions per thread.

If we increase the population to 400, we get times of 8 min for 2 threads, 7 min 27 s for 4 threads and 7 min for 8 threads. The continued reduction in time seems to support the theory of a minimum number of solutions per thread to have maximum time benefit.

Overall for the time experiments we can conclude that moving the system to the cloud and adding multi-threading capabilities significantly improve the time. However some tuning may be required to optimize the number of threads to be used, to gain the most time benefit. With this reduced time to optimize we can then increase the population size in the optimization to 400. This gives us a similar time to optimize in the cloud when compared to the time to optimize on the laptop with a population of 100.

Table 8 Results from increasing population to 200

	Coverage	Travel	Utilization	Balance	Team balance	Dist. val
	90.60	2.76	90.10	153.48	20.00	0.48
	89.66	3.38	89.16	93.84	4.00	0.53
	90.96	3.13	90.45	92.66	17.00	0.51
	92.77	4.59	92.26	84.71	8.00	0.50
	90.91	3.58	90.40	105.00	16.00	0.49
Avg.	90.98	3.49	90.48	105.94	13.00	0.50
SD	1.13	0.69	1.12	27.54	6.71	0.02

Table 9 Results from increasing population to 400

	Coverage	Travel	Utilization	Balance	Team balance	Dist. val
	92.10	3.55	91.59	102.89	14.00	0.51
	87.26	2.87	86.78	137.65	8.00	0.50
	92.45	4.17	91.94	100.40	9.00	0.50
	92.74	2.72	92.23	102.55	10.00	0.54
	92.49	2.98	91.98	115.19	8.00	0.54
Avg.	91.41	3.26	90.90	111.73	9.80	0.52
SD	2.33	0.60	2.32	15.61	2.49	0.02

6.4 The increased population results

Now that we have gained a significant time benefit, because the system now runs in the cloud, we can increase the population size, thus covering more of the search space. However our aim here is to see if increasing the population size actually gives us improved results. As if there is minimal benefit in the results of the optimization, then it may be that the most benefit from moving the system to the cloud is just time and the population should stay at 100 to gain the most time benefit.

We ran the optimization with the type-2 genetically optimized fuzzy systems selected. We increased the population to 200 and the results for this experiment are given in Table 8.

We ran the optimization five times, smaller than the 10 for the other experiments, however the standard deviation (SD) is significantly reduced due to the increase in population size. It has been reduced from 0.05 to 0.02 or 60%. In addition to the more consistent results the results give improved objective values and result in an increased distance value by 0.06 or 13.64%. Perhaps more significantly this increased population size has helped the NSGA-II and the many-objective problem, as all five objectives are improved over the average results in Table 7.

We then ran the optimization again with a population of 400. These results can be found in Table 9. The standard deviation is the same as a population of 200. However the average distance value has increased to 0.52, and increase of 4%. As with the population of 200 all objectives have been improved over the average results given in Table 7. Additionally these results improve in four out of five objectives when compared to the population of 200 results.

6.5 Results summary

A summary of the average results can be found in Table 10. Where T1 means type-1 fuzzy systems and T2 means type-2 fuzzy systems. T2_POP200 and T2_POP400 are the tuned type 2 systems with populations of 200 and 400 respectively. Here we can see the increased improvement of the results. All the results improve over the original, in all objectives. A result of using the fuzzy systems with a multi-objective genetic algorithm. We have also shown that tuning any fuzzy system that is to be used will improve the results and showing that the tuned type-2 systems improve the results the most.

Due to the availability of cloud resources and the modification of the software to support multi-threaded genetic algorithms, we can improve the optimization process in two

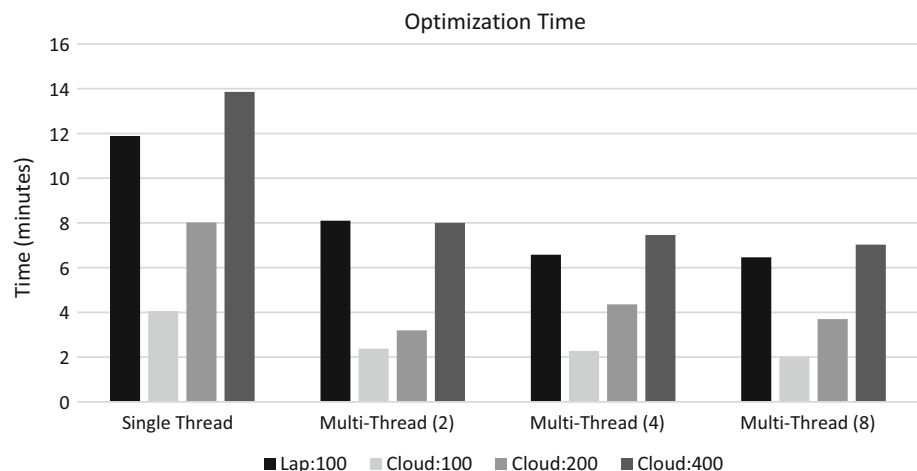
Fig. 17 Time for optimization

Table 10 Results summary

	Coverage	Travel	Utilization	Balance	Team balance	Dist. val	SD
Original	74.60	7.00	74.03	428.74	71.00	–	–
T1	89.50	3.47	88.82	246.73	31.30	0.38	0.07
T2	88.11	3.65	87.43	222.93	19.50	0.41	0.05
T1_Tuned	88.10	3.45	87.42	222.96	25.80	0.40	0.06
T2_Tuned	89.88	3.51	89.19	181.26	22.00	0.44	0.05
T2_POP200	90.98	3.49	90.48	105.94	13.00	0.50	0.02
T2_POP400	91.41	3.26	90.90	111.73	9.80	0.52	0.02

ways. We can either run the optimization in a greatly reduced time, or we can run it for the same time with a greatly increased population size. The increase in population has produced even stronger and more consistent results. Improving by as much as 18.18% if the population is increased to 400.

7 Conclusions and future work

In this paper we have presented a cloud based many-objective type-2 fuzzy logic based mobile field workforce area optimization system. We have demonstrated the need to optimize any fuzzy logic system used in the optimization process. The optimization of our type-2 fuzzy logic system improved the results by 10.00%. Additionally we have explained the potential practical benefits and given a detailed analysis of the security risks associated with cloud computing and network security.

We have also evaluated the potential improvements in results that can be gained from moving the system from personal hardware to the cloud. This allow the optimization process to run much faster, by as much as 83.33%, allowing the population size of the genetic algorithm to be increased by 300%. This increase in population resulted in better results. These results improved, on average, in all objectives when compared to the smaller population tests by as much as 18.18%.

These improvements allow the system to be effectively used on a daily bases. The users of the system will still run it for the same amount of time, but are presented with better results. In addition to this, their CPU's are freed up and they can run the optimization as long as they want without having to worry about shutting off their laptops for travel purposes. One of the key benefits of cloud computing.

In our future work we now need to tests its adaptiveness to real-time events. With this system we have the capabilities to optimize the working areas for the upcoming week, or even for the next day. However it has yet to be tested in a rapidly changing environment, where the optimization may need to

run every hour throughout the workday or on emergency or high priority incidents. Real time optimization may require resources being updated remotely, which could be done from the cloud. This would then lead to further data collection as the engineers' mobile devices would then contribute data to the optimization process [24].

Compliance with ethical standards

Conflict of interest All the authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Starkey A et al (2016) A multi-objective genetic type-2 fuzzy logic based system for mobile field workforce area optimization. *Inf Sci* 329:390–411
2. Domberger R, Frey L, Hanne T (2008) Single and Multiobjective Optimization of the train staff planning problem using genetic algorithms. In: *IEEE congress on evolutionary computation*, pp 970–977
3. Zhu K, Song H, Liu L, Gao J, Cheng G (2011) Hybrid genetic algorithm for cloud computing applications. In: *2011 IEEE Asia-Pacific services computing conference*, pp 182–187
4. Lei Z, Xiang J, Zhou Z, Duan F, Lei Y (2012) A multi-objective based scheduling strategy based on MOGA in a cloud computing environment. In: *Proceedings of IEEE CCIS2012*, pp 386–391
5. Mudaliar DN, Modi NK (2013) Unraveling travelling salesman problem by genetic algorithm using M-Crossover operator. In: *2013 international conference on signal processing image processing & pattern recognition*, pp 127–130
6. Hossain K, El-Saleh A, Ismail M (2011) A comparison between binary and continuous genetic algorithm for collaborative spectrum optimization in cognitive radio network. In: *IEEE student conference on research and development*, pp 259–264
7. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput* 6(2):182–197
8. Ishubuchi H, Tsukamoto N, Nojima Y (2008) Evolutionary many-objective optimization: a short review. In: *IEEE congress on evolutionary computation*, pp 2419–2426
9. Deb K, Jain H (2014) An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting

- approach, part I: solving problems with box constraints. *IEEE Trans Evol Comput* 18(4):577–601
10. Tanigaki Y, Narukawa K, Nojima Y, Ishibuchi H (2014) Preference-based NSGA-II for many-objective knapsack problems. In: 7th joint international conference on soft computing and intelligent systems Japan, pp 637–642
 11. Cheng J, Yen G, Zhang G (2015) A many-objective evolutionary algorithm with enhanced mating and environmental selections. *IEEE Trans Evol Comput* 19(4):592–604
 12. Wickramasinghe UK, Li X, (2009) A distance metric for evolutionary many-objective optimization algorithms using user-preferences. In: Nicholson A, Li X (eds) *AI (2009) advances in artificial intelligence*. Springer, Berlin
 13. Deb K, Jain H (2012) Handling many-objective problems using an improved NSGA-II procedure. In: *IEEE world congress on computational intelligence*. Brisbane, Australia, pp 1–8
 14. Karnik NN, Mendel JM (2001) Centroid of a Type-2 fuzzy set. *Inf Sci* 132:195–220
 15. Hagrass H (2004) A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. *IEEE Trans Fuzzy Syst* 12(4):524–539
 16. Hagrass H, Wagner C (2012) Towards the widespread use of type-2 fuzzy logic systems in real world applications. *IEEE Comput Intell Mag* 7(3):14–24
 17. Hagrass H, Alghazzawi D, Aldabbagh G (2015) Employing type-2 fuzzy logic systems in the efforts to realize ambient intelligent environments. *IEEE Comput Intell Mag* 10(1):44–51
 18. Lynch C, Hagrass H, Callaghan V (2006) Embedded interval type-2 neuro-fuzzy speed controller for marine diesel engines. In: *Proceedings of the 2006 international conference on information processing and management of uncertainty in knowledge-based systems (IPMU 2006)*. France, Paris, pp 1340–1347
 19. Mendel JM, John RI, Liu F (2006) Interval type-2 fuzzy logic systems made simple. *IEEE Trans Fuzzy Syst* 14(6):800–807
 20. Wan J et al (2014) VCMIA: a novel architecture for integrating vehicular cyber-physical systems and mobile cloud computing. *ACM/Springer Mob Netw Appl* 19(2):153–160
 21. Starkey A, Hagrass H, Shakya S, Owusu G (2015) A genetic type-2 fuzzy logic based approach for the optimal allocation of mobile field engineers to their working areas. In: *IEEE international conference on fuzzy systems*. Istanbul, Turkey, pp 1–8
 22. Reza Rahimi M, Ren J, Liu C, Vasilakos A, Venkatasubramanian N (2014) Mobile cloud computing: a survey, state of art and future directions. *Mob Netw Appl* 19(2):133–143
 23. Reza Rahimi M, Venkatasubramanian N, Vasilakos A (2013) MUSIC: mobility-aware optimal service allocation in mobile cloud computing. In: *6th international conference on cloud computing*, pp 75–82
 24. Feng Z, Zhu Y, Zhang Q, Ni L, Vasilakos A (2014) TRAC: truthful auction for location-aware collaborative sensing in mobile crowdsourcing. In: *IEEE international conference on computer communications*, pp 1231–1239