Volume 4 • Issue 1 • April 2013

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS (IJAE)

ISSN : 2180-124X

Publication Frequency: 6 Issues / Year

CSC PUBLISHERS
http://www.cscjournals.org

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS (IJAE)

VOLUME 4, ISSUE 1, 2013

EDITED BY
DR. NABEEL TAHIR

ISSN (Online): 2180-124X

International Journal of Artificial Intelligence and Expert Systems (IJAE) is published both in traditional paper form and in Internet. This journal is published at the website http://www.cscjournals.org, maintained by Computer Science Journals (CSC Journals), Malaysia.

IJAE Journal is a part of CSC Publishers

Computer Science Journals

http://www.cscjournals.org

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS (IJAE)

Book: Volume 4, Issue 1, April 2013
Publishing Date: 30-04-2013
ISSN (Online): 2180-124X

IJAE Journal is a part of CSC Publishers
http://www.cscjournals.org

Published in Malaysia

Typesetting: Camera-ready by author, data conversation by CSC Publishing Services – CSC Journals, Malaysia

CSC Publishers, 2013

EDITORIAL PREFACE

The International Journal of Artificial Intelligence and Expert Systems (IJAE) is an effective medium for interchange of high quality theoretical and applied research in Artificial Intelligence and Expert Systems domain from theoretical research to application development. This is the *first* Issue of Volume *four* of IJAE. The Journal is published bi-monthly, with papers being peer reviewed to high international standards. IJAE emphasizes on efficient and effective Artificial Intelligence, and provides a central for a deeper understanding in the discipline by encouraging the quantitative comparison and performance evaluation of the emerging components of Expert Systems. IJAE comprehensively cover the system, processing and application aspects of Artificial Intelligence. Some of the important topics are AI for Service Engineering and Automated Reasoning, Evolutionary and Swarm Algorithms and Expert System Development Stages, Fuzzy Sets and logic and Knowledge-Based Systems, Problem solving Methods Self-Healing and Autonomous Systems etc.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Started with Volume 4, 2013, IJAE appears with more focused issues related to artificial intelligence and expert system research. Besides normal publications, IJAE intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

IJAE give an opportunity to scientists, researchers, and vendors from different disciplines of Artificial Intelligence to share the ideas, identify problems, investigate relevant issues, share common interests, explore new approaches, and initiate possible collaborative research and system development. This journal is helpful for the researchers and R&D engineers, scientists all those persons who are involve in Artificial Intelligence and Expert Systems in any shape.

Highly professional scholars give their efforts, valuable time, expertise and motivation to IJAE as Editorial board members. All submissions are evaluated by the International Editorial Board. The International Editorial Board ensures that significant developments in image processing from around the world are reflected in the IJAE publications.

IJAE editors understand that how much it is important for authors and researchers to have their work published with a minimum delay after submission of their papers. They also strongly believe that the direct communication between the editors and authors are important for the welfare, quality and wellbeing of the Journal and its readers. Therefore, all activities from paper submission to paper publication are controlled through electronic systems that include electronic submission, editorial panel and review system that ensures rapid decision with least delays in the publication processes.

To build its international reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJAE. We would like to remind you that the success of our journal depends directly on the number of quality articles submitted for review. Accordingly, we would like to request your participation by submitting quality manuscripts for review and encouraging your colleagues to submit quality manuscripts for review. One of the great benefits we can provide to our prospective authors is the mentoring nature of our review process. IJAE provides authors with high quality, helpful reviews that are shaped to assist authors in improving their manuscripts.

Editorial Board Members
International Journal of Artificial Intelligence and Expert Systems (IJAE)

EDITORIAL BOARD

Dr. Muhammad Ali Imran
University of Surrey
United Kingdom

TABLE OF CONTENTS

Volume 4, Issue 1, April 2013

Pages

Two Phase Algorithm for Solving VRPTW Problem

Bhawna Minocha *bminocha@amity.edu*
Amity School of Computer Sciences
Noida, India

Saswati Tripathi *saswati@iift.ac.in*
Indian Institute of Foreign Trade
Kolkata, India

Abstract

Vehicle Routing Problem with Time Windows (VRPTW) is a well known NP hard combinatorial scheduling optimization problem in which minimum number of routes have to be determined to serve all the customers within their specified time windows. Different analytic and heuristic approaches have been tried to solve such problems. In this paper we propose a two phase method which utilizes Genetic algorithms as well as random search incorporating simulated annealing concepts to solve VRPTW problem in various scenarios.

Keywords: Vehicle Routing Problem with Time Windows, Genetic Algorithm, Random Search Algorithm, Simulated Annealing

1. INTRODUCTION

The Vehicle Routing Problem (VRP) is a well known NP-hard combinatorial optimization problem. It arises in distribution systems that usually involve scheduling in constrained environment. It lies at the heart of distribution management. It is faced each day by companies and organizations engaged in the delivery and collection of goods and people. Much progress has been accomplished since the publication of the first article on the “truck dispatching” problem [23]. Several variants of the basic problem have been proposed and different formulations developed. Exact solutions techniques as well as numerous heuristics have also been developed and tried for solving the vehicle routing problems.

In VRP a fleet of vehicles must visit a number of geographically scattered customers. All vehicles start and end at a common home base, called the depot. Each customer must be visited exactly once. The cost of traveling between each pair of customers and between the depot and each customer is given. Additionally each customer demands a certain quantity of goods delivered (known as customer demand). Each vehicle has an upper limit on the amount of goods that it can carry (known as the capacity). All vehicles are of the same type and hence have the same capacity. The problem is for a given scenario to plan routes for each vehicle, such that all routes start and end at the depot, each customer is served by exactly one vehicle, the total demand at locations enroute do not exceed the carrying capacity of the vehicle in a way that the overall cost of the routes are minimized. The Vehicle Routing Problem with Time Windows (VRPTW) is a generalization of VRP. VRPTW is one of its variant amongst Capacitated VRP, Multiple Depot VRP, VRP with Backhauls and VRP with Pick-up and Deliveries.

VRPTW is VRP with additional restriction that each customer has a time window. The vehicle can visit the customer in this specified time window only. It does not accept a vehicle after the latest time specified in the time window. However, if a vehicle arrives at such a destination prior to earliest specified time of the window, vehicle must wait and service will not start until the time window of the customer actually opens. Figure 1 shows a graphical model of VRPTW and its solution.

The objective of VRPTW is to service all the customers as per their requirement while minimizing the number of vehicles required as well as the total travel distance by all the vehicles used without violating capacity constraints of the vehicles and the customers' time window requirements such that each customer is visited only once by one of the vehicles. All the routes are to start and ultimately end at the originating depot.

VRPTW arise in a variety of real life situations. VRPTW arises in milk float, mail delivery, school bus routing, solid waste collection, heating oil distribution, newspaper distribution, transportation of persons, parcel pick-up and delivery, dial-a-ride systems, airline/railway fleet routing and several such other situations.

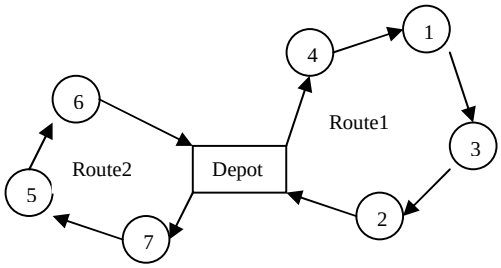

FIGURE 1: Typical output for VRPTW.

VRPTW is NP-hard. It has been extensively investigated in recent years using analytic optimization techniques, heuristics and meta-heuristics approaches. The term heuristics and meta-heuristics are usually associated with Random Search based techniques, Simulated Annealing based techniques, Tabu Search, Genetic Algorithms, Particle Swarm Optimization, Ant Colony Optimization, cross-entropy, stochastic approximation, multi-start clustering algorithms.

The available literature on VRPTW can be broadly divided into two categories: exact optimization and heuristic algorithms. Using exact optimization techniques, [10], [4], [11], [12] obtained significant improvements in Solomon's benchmark problem. Survey of the VRPTW literature by heuristics has been given by [3], [14]. Tabu search is used by [5], [25] to solve these problems whereas [7], [26], [27] considered ant colony optimization approach. Large neighborhood search (LNS) is applied by [18]. This was extended by [17] as Adaptive-LNS approach to solve VRPTW problems. Parallelization of a two-phase metaheuristic technique proposed by [9] for solving VRPTW. A complete survey of the VRPTW literature, which includes methods of both the categories, is given by [6].

The Genetic Algorithm (GA) approach was proposed by [8] in 1975. It is an adaptive heuristic search method that mimics evolution through natural selection. It works by combining selection, crossover and mutation operations of genes. Genetic Algorithm and random search heuristics have been frequently used for solving combinatorial optimization problems which are nondeterministic polynomial hard or nondeterministic polynomial complete. Whereas genetic algorithms search for the optimal solution in a large region of search space making use of genetic operators, random search based techniques usually achieve this in conjunction with annealing approach.

GA approach was first used by Blanton et al. [2] to solve VRPTW. They hybridized GA approach with a greedy heuristic. A cluster-first, route-second method using genetic and local search optimization was used by [22] and GENEROUS by [15]. A multi-objective representation of VRPTW using pareto-ranking was used by [13]. [1], [21] and many more also used GA for solving VRPTW problems.

Genetic Algorithms have been frequently used for solving VRPTW problem which is NP-hard. However it has been noticed that the techniques do not guarantee a near optimal solution in each case. In some cases the yielded solution is far from the global optimal. Keeping this in view an attempt has been made in the present study to first search for the optimal solution of the VRPTW problem using conventional genetic algorithm approach and once the solution is achieved to perturb it randomly in annealing type manner to see if a still better solution can be achieved.

The present study is organized as under: In section 2, we describe the proposed two phase optimization technique for solving VRPTW. Section 3 describes the details of genetic algorithm approach adopted by us for solving VRPTW. Use of random search approach in annealing type manner for further improvement of the obtained results is next explained in section 4. Application of the technique on selected 30 problems taken from [19] benchmark set of problems is considered in section 5. Conclusions based on present study are finally drawn in section 6.

2. PROPOSED ALGORITHM FOR VRPTW

The proposed algorithm for solving vehicle routing problem with time windows works in two phases. In the first phase it uses a genetic search based algorithm described in section 3 to generate an optimal solution. In the next phase it perturbs iteratively the optimal solution obtained in Phase I in a random search manner incorporating annealing concept to see if a still better solution is possible or not. The proposed algorithm works as under:

Phase I: Genetic Algorithm phase

i) Build an initial population and evaluate fitness value of each individual in the population.

ii) Find the individual which has best fitness value.

iii) Generate new individuals by applying genetic operators to suitably selected members from the current population.

iv) Evaluate fitness of new individuals and place them in new population.
v) Repeat step (ii) to (iv) till specified stopping criterion satisfied.

vi) Store the individual with the best fitness value.

Phase II: Random Search Phase

i) Use the best solution obtained in phase I as current solution.

ii) Randomly perturb the current solution to obtain a new solution and evaluate the fitness.

iii) Is new solution better than the current one? If yes, replace current solution by the new one else replace current solution by the new one in annealing type manner.

iv) Repeat step (ii) and (iii) iteratively a specified number of times.

The final solution achieved in phase II is the desired optimal solution. A special feature of the proposed two phase algorithm is that the Phase II does not start random neighbourhood search from an arbitrarily chosen solution but with the best solution achieved in Phase I using genetic algorithm approach. So there are great chances of the obtained solution being better than the one provided by genetic algorithm approach in case such a solution exists.

3. GENETIC ALGORITHM PHASE

In this phase an initial population is first generated. After building the initial population, all individuals are evaluated according to the fitness criteria. The evolution continues with

tournament selection in which good individuals are selected for reproduction. In each generation two best individuals are preserved for the next generation without being subjected to genetic operations. Crossover and mutation operations are then applied to modify the suitably selected individuals to form new feasible individuals for the population. The detailed description of genetic algorithm used to solve vehicle routing problem with time windows is as under.

3.1 Chromosome & Individual Representation

The chromosomes in genetic algorithms are often represented as a fixed-structure bit string, for which the bit positions are assumed to be independent and context insensitive. However, such a representation is not suitable for VRPTW, which is an order-oriented NP-hard optimization problem where specific sequences among customers are essential.

For solving VRPTW, the representation that we have used is same as used by [24]. The representation is as follows: Each customer is assigned a unique integer identifier *i*, where $i \varepsilon N$. An individual, which is a collection of chromosomes, represents a complete routing solution. Depending on how the customers are routed and distributed, every individual can have different number of routes for the same data set.

Each chromosome represents a route, which is variable in length. It contains a sequence of customers in the order in which they are visited by the vehicle. A different vehicle is needed for every chromosome of the individual. Every individual and every route ensure to be feasible, in terms of capacity and time window constraints. The central depot is not considered in this representation, because all routes necessarily start and end at the depot. However time from the depot to first customer of the route and time from last customer of the route to depot is taken into account. Figure 2 represents an illustrative complete routing solution for a problem instance with 25 customers, consisting of 3 routes which are served by 3 vehicles; genetically we call each route a chromosomes and complete solution an individual.

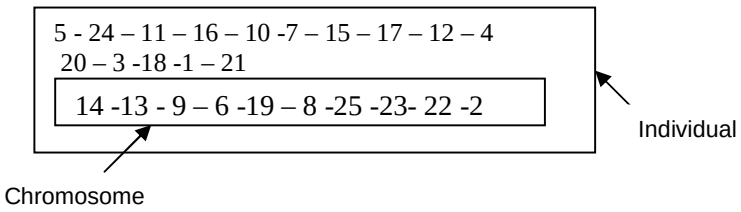

FIGURE 2: An individual with 3 chromosomes (vehicles / routes). It represents a solution for a problem instance with 25 customers.

3.2 Initial Population

An initial population is built such that each individual (which represents a complete routing solution in our case) is a feasible solution. In other words every individual and every chromosome/route in the selected population satisfies time window and capacity constraints. The first feasible solution is generated using Push Forward Insertion Heuristics (PFIH) introduced by [19]. This method has been frequently used in literature. Details of this method are available in [22]. Rest of the solutions of initial population are generated by selecting the customers in a random manner and inserting them in an existing route, if such a possibility exists, otherwise a new route is created. Any customer that violates any constraint is deleted and a new route is added to serve such a customer. This process is repeated until all the customers get served and a feasible initial population has been generated.

3.3 Fitness

The fitness function measures the quality of the represented solutions. As soon as all the individuals have been created, they are ranked as per their fitness. It is commonly taken as the

objective function of the optimization problem but it may not necessarily coincide with it. For solving VRPTW using the proposed algorithm following fitness criteria's have been considered:

1. *Inverted Distance (INVD):* Inverse of the total traveled distance is used to calculate the fitness of the individuals.

2. *Distance Traveled and Number of Routes (DR):* In this case we minimize the total distance traveled keeping at the same time the number of vehicles as low as possible as each route requires one vehicle to operate it.

3. *Number of Routes and Distance traveled (RD):* Same as (2) however with the priorities interchanged. First priority is to reduce the number of routes and second priority is to reduce total distance traveled.

4. *Weighted Sum Method (WM):* In this case effort has been made to minimize weighted sum of two objectives. For VRPTW, the weighted sum objectives which has been minimized is:

$$F(x) = \alpha \cdot V + \beta \cdot TD$$

where α and β are suitable weight coefficients associated with total number of vehicles, V and total distance traveled, TD by vehicles. The weight values of the coefficients used have to be established empirically (we have used α = 100 and β = 0.001) for this study.

3.4 Selection and Elitism

In selection, parents are selected for crossover. There are many methods proposed in the literature for this. In this study, an x-way tournament selection procedure has been used. Here x individuals are randomly selected and then the individual with highest fitness is declared the winner. This process is repeated until the number of selected individuals equals the number necessary for crossover. In this study, tournament size, i.e. x has been taken to be 3.

In the elitism process the good individuals are retained for reproduction. This ensures that the best solution obtained from the present population is copied unaltered in the next population. We replace the two worst individuals in the new population with the best two individuals of the parent population.

3.5 Crossover and Mutation

The classical single/double point crossover operators are relevant to string entries that are order less. They put two integer/binary strings side by side and make a cut point (or two cut points) on both of them. A crossover is then completed by swapping the portions after the cut point (or between the cut points) in both the strings. However it is not appropriate for scheduling problems like TSP or VRP where sequence or order among the integers is very important because duplication and omission of vertices can produce infeasible sequences in the offspring. Therefore in present study Route-Exchange crossover is used. Earlier this crossover has been applied by [21], [13]. However they choose the best route according to the objectives for crossover. In the present study, once a pair of individuals is selected for crossover, efforts are made to exchange a route that has minimum number of nodes in each of the two individuals. To ensure that all individuals are feasible routing solutions after crossover any duplication is deleted.

Mutation is necessary for inserting new characteristics that are not present in the current individuals. Without mutation the search gets limited to a very small area in the feasible region. In present study effort has been made to achieve this by transferring customers from a route that has minimum number of customers to other routes if feasible so that such a route gets deleted (if possible) and the number of routes reduced.

4. RANDOM SEARCH PHASE

The application of random search concept incorporating simulated annealing in Phase II of the algorithm to solve the VRPTW is as follows: Initially the best solution, say *S* of the Phase I of the genetic algorithm is taken as the starting current solution. From this a new solution *S′* is obtained by randomly moving a customer of the current solution from one route to another route where it is feasible. The new solution if it decreases the objective function value (or leaves it unchanged) is accepted to replace the current solution; else it is accepted to replace the current solution in annealing type manner. More precisely, the new solution *S′* is accepted as the new current solution if $\Delta E \leq 0$, (where ΔE = Fitness (*S′*) - Fitness(*S)*) else to ensure the search to escape a local optimum, solutions that increases the objective function value are accepted if

$$exp\ (-\Delta E / T_k) > \theta \qquad (4.1)$$

where θ is a randomly selected number between [0, 1], if $\Delta E > 0$, where T_k is a parameter called the "temperature". (The value of T_k is gradually decreased from a relatively large value to a small value close to zero. These values are controlled by a cooling schedule which specifies the initial and temperature values at each stage of the algorithm. Eqn. (4.1) implies that in a minimization problem large increases in objective function, so called uphill moves, are more likely to be accepted when T_k is high. As T_k approaches zero most uphill moves are rejected.

In order to achieve good optimization results the simulated annealing metaheuristic has to be adjusted specific to the problem. With respect to the VRPTW following parameters have been taken into account:

a) *Initial temperature of annealing* T_0: If T_0 is too high then almost all new solutions are accepted and the search produces a series of random solutions. When T_0 is too low, very few movements are allowed which reduce the scope of the search. After a series of experiments we decided to set constant initial temperature T_0 =100.

b) *Cooling schedule:* One of the most popular temperature reduction functions is by [28]. It is based on geometric reduction $T_{k+1} = T_k * \gamma$. In the present work we have set parameter γ as 0.96.

c) *Number of annealing steps executed in each temperature:* It is usually related to the size of a solution neighbourhood. In our computational experiments the numbers of annealing steps are set as n^2, where n is the number of customers in the problem considered.

d) *Termination condition:* We set the termination criteria as T_0 =0.

Contrary to the classical approach in which a solution to the problem is taken as the last solution obtained in the annealing process, we memorize the best solution found during the whole annealing process and record it as the best solution found. The algorithm has been coded in C++ and run on an Intel(R) Core(TM) 2 Duo 2.0 GHz for solving chosen test problems.

5. NUMERICAL RESULTS

In this section we present our computational experience of using the proposed algorithm in solving a set of benchmark test problems selected from Solomon's [19] set of problems.

In the present study 30 problems have been chosen from Solomon's set of problems. Of these 15 problems are of 25 customers, 10 problems of 50 customers and 5 problems of 100 customers. Each problem was solved ten times using developed algorithm. The best and the worst results obtained in each problem in ten trials are listed in the tables. We first applied the two phase algorithm considering the vehicle routing problem as a single objective problem, thus applied the fitness criteria INVD. Results obtained in phase I and their refinement after the phase II is shown in TABLE 1 for 25 customer problems, TABLE 2 for 50 customer problems and TABLE 3 for 100

customer problems. Out of ten solutions obtained in each case the Best solution and Worst solution of each problem after completion of phase I and phase II are listed in these tables. Stopping criterion used for phase I is a maximum of 1000 generations or when no improvement is observed in the objective function value of the best individual in consecutive 100 generations. In the Phase I, the population size was taken as 100; crossover rate was taken to be 0.80 and mutation rate 0.20.

However VRPTW is in reality a multi objective problem in which we went to minimize the number of routes used as well as the total distance traveled. Thus we have used the same two phase algorithm to solve VRPTW considering it as a multi-objective problem in order to check if performance in both objectives can be simultaneously improved. The performance of proposed algorithm is tested on the same set of 30 benchmark problems. TABLE 4, 5 and 6 shows the results obtained after phase II for the three multiobjective criteria's DR, RD and WM for 25 customer, 50 customer and 100 customer problems respectively.

Tables also present a summary of the results obtained and their comparison with the best-known solutions available to us from literature. Bold numbers in tables indicate that the obtained solution are same as the best-known or has yielded an improvement on the currently best known solution in the literature.

5.1 Discussion on the Results

Results presented in the TABLE 1, 2 and 3 shows the results after completion of genetic algorithm Phase I and random search based Phase II. For 25 customer problems there are only four cases out of fifteen in which no further refinement in the solutions is seen after Phase I. For 50 customer problems there are two such cases out of ten and for 100 customer problems there is one such case out of five. In all out of thirty problems, in twenty three cases Phase II refines solution obtained after Phase I.

For 25 customer problem the proposed algorithm has produced new improved results in six problems out of fifteen problems investigated as shown in TABLE 1. The obtained results require less number of vehicles, however at the expense of additional distance to be traveled. In TABLE 2 for 50 customer problems; five problems from set R2 show improvement while one problem from each C1 and C2 set have identical solution with the best-known solutions. In R201 problem obtained results reflect significant decrease in the requirement of number of vehicles, (maximum reduction is two and minimum is one) than the number of vehicles used in best-known solutions. In case of 100 customer problems proposed algorithms provide improved results in one problem out of five considered as shown in TABLE 3 while another problem provide solution identical with the best-known solution available in the literature. In this case the improvement is in terms of less total distance traveled than the best-known solution. However, it needs one extra vehicle than the number of vehicles used in the best-known. In the remaining problems solutions obtained are either identical to the best known solutions or within 10% of the best-known solution. Figure 3 shows the geographical representation of solution obtained for problem C101 for 100 customers.

TABLE 4, 5 and 6 shows the results obtained after phase II for the three multiobjective criteria's DR, RD and WM for 25 customer, 50 customer and 100 customer problems respectively. In TABLE 4 for 25 customer problems the proposed algorithm has yielded either improved results in terms of lesser number of vehicles required than the best known solutions or identical solutions with the best-known solutions in ten problems out of selected fifteen problems. In some cases such as RC203 and RC206 the improvement is quite significant as it decreases the need of number of vehicles by one to two vehicles compared to the number of vehicles used in the best-known solutions. However this improvement in terms of lesser number of vehicles required is at cost of more distance to be traveled. In case of remaining five problems obtained solutions are within 10% of the best-known solutions. A comparison of the results obtained with three objectives shows that objectives DR and WM provide better results than objective RD.

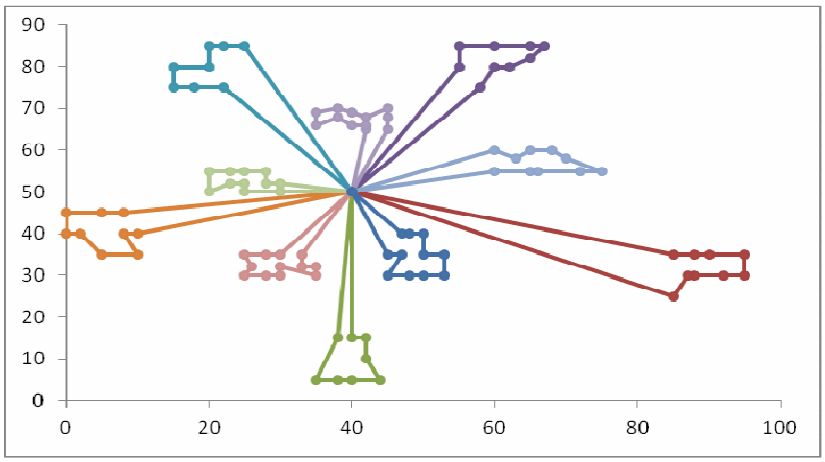

FIGURE 3: Geographical representation of solution obtained for problem C101 for 100 customers.

In case of 50 customer problems for multiobjective criteria's DR, RD and WM results obtained are presented in Table 5. The use of the two phase algorithm has provided six improved solutions with all the three multi-objective criteria's. Also DR criterion has produced solutions identical with the best-known in two cases. Thus in all out of ten selected problems, eight problems either yielded improved results in terms of lesser number of vehicles required than the best known solutions. However this improvement is at the expense of extra distance to be traveled. (Maximum reduction in the number of vehicles required is three and minimum one.) In remaining two problems solutions are within 10% of the best-known solution. Performance wise the three objectives may be ranked as WM, DR and RD, in that order. In the case of 100 customer problems for multiobjective criteria's results are shown in TABLE 6; proposed algorithms produced improved results in one problem out of five considered while one problem yielded identical solution as the best-known solution available in literature. In the present case the improvement is in terms of less total distance traveled than the best-known solution. However, it needs one more vehicle than the number of vehicles used in the best-known. In remaining three problems solutions are within 10% of the best-known solution. Performance wise WM objective is again superior to the remaining two. Figure 4 shows the geographical representation of solution obtained for problem R105 for 100 customers.

Lesser number of vehicles means saving in terms of needed man power requirement and vehicle maintenance cost. It also means less chaos at the depot. However, solutions which reported lesser number of vehicles reported more distance to travel, adding to fuel cost. However fuel cost is comparatively less expensive than arranging additional vehicles and man power needed to operate them. Moreover, it is easy for the management to handle the things with less number of vehicles.

6. CONCLUSIONS

Vehicle routing problem with time windows involves the optimization of routes for multiple vehicles so as to meet all constraints and to minimize the number of vehicles needed and total distance traveled. The proposed two phase algorithm is based on genetic algorithm and random search incorporating simulated annealing concept. Performance of proposed algorithms is comparable to those available in literature and in some cases even better in terms of number of vehicles which means less fuel, manpower and vehicle maintenance cost with more distance to travel. We solved each problem ten times and have presented the best and worst achieved solutions to indicate the range of variations in the solution obtained. The variation of the best solution from worst solution is generally in the range of 20 to 40 percent indicating thereby that it is advisable to solve a problem more than once to achieve the best results. This is due to the probabilistic nature of the techniques which search a near optimal solution. As for future work, it may be interesting to test proposed algorithm on some application of VRPTW.

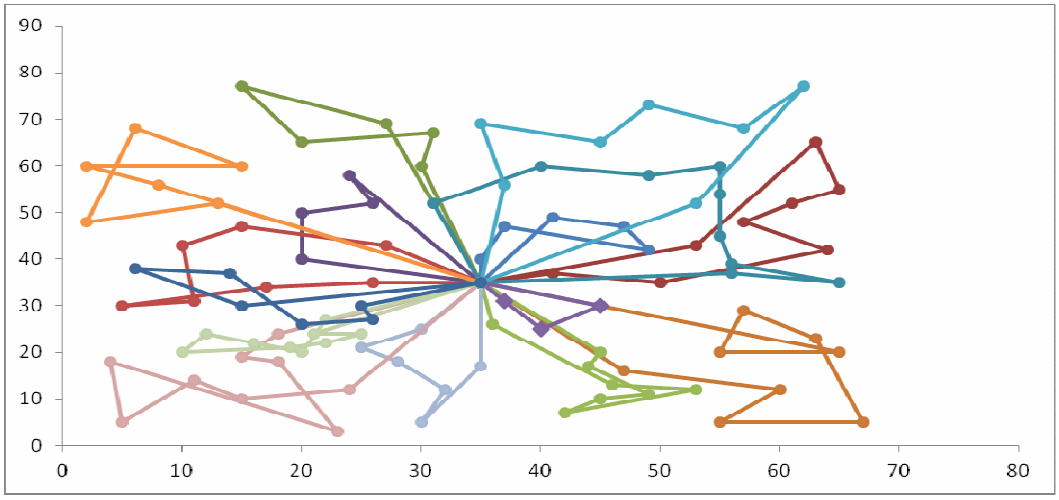

FIGURE 4: Geographical representation of solution obtained for problem R105 for 100 customers.

Problem	INVD				Best Known [Ref.] *(No. of Vehicles/ distance traveled)*
	After Genetic Algorithm Phase		**After Refining the best obtained from GA with SA in Phase II**		
	Best *(No. of Vehicles/ distance traveled)*	***Worst*** *(No. of Vehicles/ distance traveled)*	***Best*** *(No. of Vehicles / distance traveled)*	***Worst*** *(No. of Vehicles/ distance traveled)*	
C201	2/ 237.15	2/ 282.25	**2/ 214.7**	2/ 250.56	**2/ 214.7 [11]**
R101	8/ 625.03	9/ 661.34	8/ 623.52	9/ 645.20	**8/ 617.1 [11]**
R102	7/ 591.51	7/ 628.93	7/ 586.05	7/ 610.46	**7/ 547.1 [11]**
R105	6/ 569.70	6/ 587.76	**5/ 559.84**	6/ 575.00	**6/ 530.5 [11]**
R109	5/ 522.05	5/ 558.03	5/ 480.23	5/ 554.98	**5/ 441.3 [11]**
RC105	4/ 412.38	4/ 506.66	4/ 412.38	4/ 495.34	**4/ 411.3 [11]**
RC106	3/ 364.97	4/ 459.10	3/ 364.97	4/ 431.43	**3/ 345.5 [11]**
RC201	3/ 385.81	3/ 454.50	3/ 385.81	3/ 452.94	**3/ 360.2 [12]**

RC202	3/ 361.64	3/ 465.04	3/ 361.64	3/ 433.73	**3/ 338.0 [10]**
RC203	2/ 468.98	3/ 482.96	**2/ 411.47**	3/ 441.22	**3/ 326.9 [4]**
RC204	2/ 438.45	2/ 475.28	**2/ 401.05**	2/ 452.17	**3/ 299.7 [4]**
RC205	3/ 399.10	3/ 482.34	3/ 363.46	3/ 432.66	**3/ 338.0 [12]**
RC206	2/ 526.33	2/ 548.06	**2/ 503.28**	2/ 528.46	**3/ 324.0 [10]**
RC207	2/ 477.63	2/ 562.91	**2/ 433.23**	2/ 533.65	**3/ 298.3 [10]**
RC208	1/ 440.80	1/ 500.75	**1/ 427.26**	1/ 483.79	**2/ 269.1 [4]**

TABLE 1: Results generated by Two Phase Algorithm for Solomon's 25 customers set Problems (considering VRPTW as single-objective optimization problem).

Problem	**INVD**				**Best Known [Ref.]** *(No. of Vehicles/ distance traveled)*
	After Genetic Algorithm Phase		**After Refining the best obtained from GA with SA in Phase II**		
	Best *(No. of Vehicles/ distance traveled)*	***Worst*** *(No. of Vehicles/ distance traveled)*	***Best*** *(No. of Vehicles/ distance traveled)*	***Worst*** *(No. of Vehicles/ distance traveled)*	
C101	5/ 385.90	6/ 444.27	**5/ 362.5**	6/ 411.30	**5/ 362.5 [11]**
C201	3/ 408.73	3/ 390.26	3/ 390.26	3/ 434.72	**3/ 360.2 [12]**
C205	3/ 389.83	3/ 459.36	**3/ 360.2**	3/ 435.24	**3/ 360.2 [12]**
R101	12/ 1097.97	12/ 1190.42	12/ 1055.56	12/ 1178.27	**12/ 1044 [11]**
R201	5/ 1108.55	5/ 1262.96	**5/ 1099.81**	5/ 1206.59	**6/ 791.9 [10]**
R202	4/ 1127.16	4/ 1150.84	**4/ 1059.12**	4/ 1149.09	**5/ 698.5 [10]**
R203	4/ 1097.14	4/ 1172.92	**4/ 1047.22**	4/ 1125.25	**5/ 605.3 [4]**

R206	3/ 1023.04	3/ 1179.46	**3/ 949.08**	3/ 1167.75	**4/ 632.4 [4]**
R209	3/ 1165.67	3/ 1208.34	**3/ 1145.18**	3/ 1198.22	4/ 600.6 [10]
RC101	8/ 981.96	9/ 1024.66	8/ 981.96	9/ 1056.65	**8/ 944 [11]**

TABLE 2: Results generated by Two Phase Algorithm for Solomon's 50 customers set Problems (considering VRPTW as single-objective optimization problem).

Problem	**INVD**				**Best Known [Ref.]** *(No. of Vehicles/ distance traveled)*
	After Genetic Algorithm Phase		**After Refining the best obtained from GA with SA in Phase II**		
	Best *(No. of Vehicles/ distance traveled)*	***Worst*** *(No. of Vehicles/ distance traveled)*	***Best*** *(No. of Vehicles/ distance traveled)*	***Worst*** *(No. of Vehicles/ distance traveled)*	
C101	10/ 838.07	11/ 1054.11	**10/ 828.94**	11/ 1022.68	**10/ 828.94 [16]**
R101	21/ 1788.7	21/ 1877.92	20/ 1736.60	21/ 1869.44	**19/ 1650.8 [16]**
R102	18/ 1516.67	19/ 1664.11	18/ 1516.67	19/ 1661.76	**17/ 1486.12 [16]**
R105	15/ 1391.85	17/ 1546.56	15/ 1380.05	17/ 1539.5	**14/ 1377.11 [18]**
RC101	15/ 1775.79	18/ 1960.92	**15/ 1640.98**	18/ 1953.77	**14/ 1696.94 [20]**

TABLE 3: Results generated by Two Phase Algorithm for Solomon's 100 customers set Problems (considering VRPTW as single-objective optimization problem).

Problem	**DR**		**RD**		**WM**		**Best Known [Ref.]**
	Best (No. of Vehicles/ distance traveled)	***Worst*** (No. of Vehicles/ distance traveled)	***Best*** (No. of Vehicles/ distance traveled)	***Worst*** (No. of Vehicles/ distance traveled)	***Best*** (No. of Vehicles/ distance traveled)	***Worst*** (No. of Vehicles/ distance traveled)	
C201	**2/ 214.7**	2/ 250.56	**2/ 214.7**	2/ 250.56	**2/ 214.7**	2/ 282.25	**2/ 214.7 [11]**

R101	8/ 618.33	8/ 665.83	8/ 623.52	8/ 677.80	8/ 629.95	8/ 651.44	**8/ 617.1 [11]**
R102	7/ 586.00	7/ 620.21	7/ 590.41	7/ 668.86	7/ 584.20	7/ 628.93	**7/ 547.1 [11]**
R105	**5/ 588.57**	6/ 628.72	**5/ 591.89**	6/ 609.90	**5/ 556.72**	6/ 612.57	**6/ 530.5 [11]**
R109	5/ 480.09	5/ 575.80	5/ 483.94	5/ 557.77	5/ 461.71	5/ 566.09	**5/ 441.3 [11]**
RC105	4/ 419.72	4/ 508.64	4/ 421.73	4/ 507.84	4/ 419.72	4/ 506.66	**4/ 411.3 [11]**
RC106	3/ 363.82	4/ 443.43	3/ 371.21	4/ 455.30	3/ 375.74	4/ 463.63	**3/ 345.5 [11]**
RC201	**2/ 519.35**	2/ 621.95	**2/ 519.35**	2/ 612.85	**2/ 496.37**	2/ 608.03	**3/ 360.2 [12]**
RC202	**2/ 478.37**	2/ 560.42	**2/ 509.95**	2/ 597.46	**2/ 478.84**	2/ 554.53	**3/ 338.0 [10]**
RC203	**2/ 419.02 1/ 524.38**	2/ 568.73	**2/ 427.57 1/ 524.38**	2/ 572.42	**2/ 441.76 1/ 521.53**	2/ 525.48	**3/ 326.9 [4]**
RC204	**2/ 382.41**	2/ 485.68	**2/ 391.66**	2/ 483.27	**2/ 386.53**	2/ 485.59	**3/ 299.7 [4]**
RC205	**2/ 549.29**	2/ 697.92	**2/ 537.83**	2/ 681.46	**2/ 546.23**	2/ 653.75	**3/ 338.0 [12]**
RC206	**2/ 497.02 1/ 565.59**	1/ 648.04	**2/ 510.77 1/ 566.86**	1/ 664.53	**2/ 483.93 1/ 565.59**	1/ 653.26	**3/ 324.0 [10]**
RC207	**2/ 433.76**	2/ 558.87	**2/ 435.32**	2/ 560.72	**2/ 424.39**	2/ 562.91	**3/ 298.3 [10]**
RC208	**1/ 420.53**	1/ 483.79	**1/ 425.94**	1/ 500.75	**1/ 427.78**	1/ 504.51	**2/ 269.1 [4]**

TABLE 4: Results generated by Two Phase Algorithm for Solomon's 25 customers set Problems (considering VRPTW as multi-objective optimization problem).

Problem	DR		RD		WM		Best Known [Ref.]
	Best (No. of Vehicles/ distance traveled)	***Worst*** (No. of Vehicles/ distance traveled)	***Best*** (No. of Vehicles/ distance traveled)	***Worst*** (No. of Vehicles/ distance traveled)	***Best*** (No. of Vehicles/ distance traveled)	***Worst*** (No. of Vehicles/ distance traveled)	
C101	**5/ 362.5**	6/ 443.36	5/ 372.42	6/ 458.45	5/ 386.27	6/ 469.57	**5/ 362.5 [11]**
C201	**2/ 501.13**	2/ 561.43	**2/ 546.73**	2/ 561.43	**2/ 501.13**	2/ 561.43	**3/ 360.2 [12]**
C205	**3/ 360.2**	3/ 471.24	3/ 389.83	3/ 476.49	3/ 385.30	3/ 473.31	**3/ 360.2 [12]**
R101	12/ 1080.71	12/ 1205.93	12/ 1093.91	12/ 1178.27	12/ 1088.71	12/ 1200.00	**12/ 1044 [11]**
R201	3/ 1220.29	3/ 1337.42	3/ 1222.49	3/ 1344.47	3/ 1208.38	3/ 1338.00	**6/ 791.9 [10]**
R202	**3/ 1054.11**	3/ 1231.76	**3/ 1057.32**	3/ 1252.27	**3/ 1041.92**	3/ 1205.38	**5/ 698.5 [10]**
R203	**4/ 1025.96**	4/ 1146.74	**4/ 1039.82**	4/ 1173.38	**3/ 1233.18**	3/ 1336.97	**5/ 605.3 [4]**
R206	**3/ 959.32**	3/ 1197.85	**3/ 964.95**	3/ 1138.22	**3/** 897.43	3/ 1181.75	**4/ 632.4 [4]**
R209	**2/ 1187.20**	3/ 1208.34	**2/ 1182.49**	3/ 1198.22	**2/ 1157.29**	3/ 1196.76	**4/ 600.6 [10]**
RC101	8/ 982.62	10/ 1043.85	8/ 986.68	10/ 1037.84	8/ 977.51	10/ 1039.26	**8/ 944 [11]**

TABLE 5: Results generated by Two Phase Algorithm for Solomon's 50 customers set Problems (considering VRPTW as multi- objective optimization problem).

Problem	DR		RD		WM		Best Known [Ref.]
	Best (No. of Vehicles/ distance traveled)	***Worst*** (No. of Vehicles/ distance traveled)	***Best*** (No. of Vehicles/ distance traveled)	***Worst*** (No. of Vehicles/ distance traveled)	***Best*** (No. of Vehicles/ distance traveled)	***Worst*** (No. of Vehicles/ distance traveled)	
C101	10/ 829.70	11/ 1021.22	10/ 838.07	11/ 1035.71	**10/ 828.94**	12/ 1062.10	**10/ 828.94 [16]**
R101	20/ 1751.74	21/ 1891.10	20/ 1760.08	21/ 1873.43	20/ 1740.18	21/ 1874.66	**19/ 1650.8 [16]**
R102	18/ 1503.71	19/ 1723.94	18/ 1519.92	20/ 1747.98	18/ 1487.29	20/ 1718.42	**17/ 1486.12 [16]**
R105	15/ 1381.23	17/ 1567.47	15/ 1391.85	17/ 1554.88	**15/ 1370.02**	17/ 1558.35	**14/ 1377.11 [18]**
RC101	15/ 1723.18	18/ 1998.63	15/ 1738.55	18/ 1983.91	15/ 1717.31	18/ 1979.89	**14/ 1696.94 [20]**

TABLE 6: Results generated by Two Phase Algorithm for Solomon's 100 customers set Problems (considering VRPTW as multi-objective optimization problem).

7. REFERENCES

[1] J. Berger and M. Barkaoui, "A parallel hybrid genetic algorithm for the vehicle routing problem with time windows" Computer Operation Research, vol. 31, pp. 2037-2053, 2004.

[2] J. L. Blanton and R. L.Wainwright, "Multiple vehicle routing with time and capacity constraints using genetic algorithms". Proceedings of the 5th International Conference on Genetic Algorithms, USA, pp. 452-459, June 1993.

[3] O. Bräysy and M. Gendreau, "Vehicle routing problem with time windows, Part II: Metaheuristics ", Transportation Science, vol. 39, pp. 119–139, 2005

[4] A. Chabrier, "Vehicle Routing Problem with Elementary Shortest Path based Column Generation." Computers and Operations Research, vol. 33(10), pp. 2972-2990, 2006.

[5] J. F. Cordeau, G. Laporte and A., Mercier, "A Unified Tabu Search for Vehicle Routing Problem with Time Windows", Journal of the Operational Research Society vol. 52 pp. 928-936, 2001

[6] J. F. Cordeau, G. Desaulniers, J. Desrosiers, M.M. Solomon and F. Soumis, "The VRP with Time Windows" , In: The Vehicle Routing Problem, SIAM Monographs on Discrete Mathematics and Applications, Toth, P. and D. Vigo (Eds.), Philadelphia, USA, pp. 157-193, 2001

[7] L. M. Gambardella, E. Taillard, G. Agazzi, " MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows", New Ideas in Optimization, London: McGraw-Hill, pp. 63-76, 1999

[8] J. H. Holland, “Adaptation in Natural and Artificial System”. Ann Arbor, Michigan: The University of Michigan Press, 1975.

[9] J. Hömberger, H. Gehring, “A Two-Phase Hybrid Metaheuristic for the Vehicle Routing Problem with Time Windows”, European Journal of Operations Research, vol. 162(1), pp. 220-238, 2005.

[10] B. Kallehauge, J. Larsen, and O. B. G. Madsen, "Lagrangean duality and non-differentiable optimization applied on routing with time windows ." Computers and Operations Research, vol. 33(5),pp. 1464-1487,2006.

[11] N. Kohl, J. Desrosiers, O. B. G. Madsen, M.M. Solomon and F. Soumis, "2-Path Cuts for the Vehicle Routing Problem with Time Windows," Transportation Science, vol. 33 (1) , pp. 101-116 ,1999.

[12] J. Larsen "Parallelization of the vehicle routing problem with time windows." Ph.D. Thesis IMM-PHD-1999-62, Department of Mathematical Modelling, Technical University of Denmark,Lyngby, Denmark, 1999.

[13] B. Ombuki, B.J. Ross, and F. Hansher, “Multi-objective Genetic Algorithm for Vehicle Routing Problem with Time Windows”, Applied Intelligence, vol. 24(1), pp. 17-30, 2006.

[14] B. Minocha and S. Tripathi, “Vehicle Routing Problem with Time Windows: An Evolutionary Algorithmic Approach”, Algorithmic Operations Research, vol. 1(2), pp. 1-15, 2006.

[15] J. Potvin and S. Bengio, “The Vehicle Routing Problem with Time Windows part II: Genetic Search. INFORMS Journal on Computing, vol. 8(2), pp. 165-172, 1996.

[16] Y. Rochat and E. Taillard, “Probabilistic Diversification and Intensification in Local Search for Vehicle Routing” Journal of Heuristics vol. 1, pp. 147−167, 1995.

[17] S. Ropke and D. Pisinger, “A General Heuristics for Vehicle Routing Problems”, Computers & Operations Research, vol. 34(8), pp. 2403-2435, 2007

[18] P. Shaw, “Using Constraint Programming and Local Search Methods to solve Vehicle Routing Problems”, Principles and Practice of Constraint Programming, Springer-Verlag , pp. 417-431, 1998

[19] M. M. Solomon, “Algorithms for Vehicle Routing and Scheduling Problems with Time Window Constraints”, Operations Research, vol. 35(2), pp. 254-265, 1987

[20] E. D. Taillard, P. Badeau, M. Gendreau, F. Guertin and J. Potvin, “A Tabu Search Heuristics for the Vehicle Routing Problem with Soft Time Windows”. Transportation Science vol. 31, pp. 170-186, 1997.

[21] K. C. Tan, L. H. Lee, K. Ou, “Hybrid Genetic Algorithms in Solving Vehicle Routing Problems with Time Window Constraint”, Asia-Pacific Journal of Operation Research 18:121-130, 2001

[22] S. Thangiah, “Vehicle Routing with Time Windows using Genetic Algorithms”. In Application Handbook of Genetic Algorithms: New Frontiers, vol. 2, Lance Chambers (Ed.): CRC Press, Boca Raton, 1995, pp. 253-277.

[23] G.B. Dantzig and J. H.Ramser, “The Truck Dispatching Problem”, Management Science, vol. 6 (1), pp. 80–91, 1959.

[24] G.B. Alvarenga, G.R. Mateus and G. De. Tomi, “Finding near optimal Solutions for Vehicle Routing Problems with Time Windows using Hybrid Genetic Algorithm”, Internet: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.11.46&rep=rep1&type=pdf , 2003,[Jan. 25, 2005]

[25] Moccia, L., Cordeau, J.F. and Laporte, G., 2011. "An incremental tabu search heuristic for the generalized vehicle routing problem with time windows" Journal of the Operational Research Society, advance online publication 4 May 2011; doi: 10.1057/jors.2011.25

[26] Chen, C. H., Ting, C.J. " A Hybrid Ant Colony System for Vehicle Routing Problem with Time Windows." Journal of the Eastern Asia Society for Transportation Studies, 2005, 6:2822-2836.

[27] Zhang, Q., Zhen, T., Zhu, Y., Zhang, W. and Ma, Z., 2010. A Hybrid Intelligent Algorithm for the Vehicle Routing with Time Windows. 2010 International Forum on Information Technology and Applications, IEEE, pp. 47-54.

[28] P.J. van Laarhoven and E.H. Aarts, "Simulated Annealing" U.S.A : Dordrecht and Boston and Norwell, MA, 1987.

An Algorithm of Policy Gradient Reinforcement Learning with a Fuzzy Controller in Policies

Harukazu Igarashi *arashi50@sic.shibaura-it.ac.jp*
Department of Information Science and Engineering
Shibaura Institute of Technology
Tokyo, 135-8548, Japan

Seiji Ishihara *ishihara_s@mail.dendai.ac.jp*
Division of Science, Department of Science and Engineering
Tokyo Denki University
Saitama, 350-0394, Japan

Abstract

Typical fuzzy reinforcement learning algorithms take value-function based approaches, such as fuzzy Q-learning in Markov Decision Processes (MDPs), and use constant or linear functions in the consequent parts of fuzzy rules. Instead of taking such approaches, we propose a fuzzy reinforcement learning algorithm in another approach. That is the policy gradient approach. Our method can handle fuzzy sets even in the consequent part and also learn the rule weights of fuzzy rules. Specifically, we derived learning rules of membership functions and rule weights for both cases when input/output variables to/from the control system are discrete and continuous.

Keywords: Reinforcement Learning, Policy Gradient Method, Fuzzy Inference, Membership Function.

1. INTRODUCTION

Much work [3-7] has been done combining fuzzy control [1] and reinforcement learning algorithms [2]. Combining benefits fuzzy control systems in that parameters included in membership functions in fuzzy control rules can be learned by reinforcement learning even if there is no teacher data in the input and output of the control system. For reinforcement learning, fuzzy rules expressed by linguistic terms are very convenient for experts to introduce a priori knowledge in the rule database and for system users to understand the if-then rules. In particular, they are appropriate for building control systems with continuous and layered system states [8].

Most combining methods proposed thus far have taken approaches using value-based reinforcement learning, such as *Q*-learning that assumes Markov Decision Processes (MDPs) for the environments and the policies of agents [3-8]. The fuzzy rules in those works usually describe system states in the antecedent part and parameter values [3,4] or functions [5] corresponding to *Q* values in the consequent part. However, fuzzy sets were not allowed to describe output variables in the consequent part. The system calculated *Q* values only for discrete actions of agents. Moreover, there were no weight parameters representing the confidence or importance of the rules that can reinforce suitable rules, suppress unsuitable rules, generate new rules, and remove unnecessary rules.

In reinforcement learning, there is another approach other than the value-based approach. The policy gradient method, which originates from Williams' REINFORCE algorithm [9], is an approach that computes the policy gradient with respect to parameters in the policy function and improves the policy by adjusting the parameters in the gradient direction [9-11]. A combining method is proposed by Wang et al. [12] using a policy gradient method called the GPOMDP algorithm that was proposed by Baxter and Bartlett [10]. However, agent actions were restricted

to be discrete, fuzzy sets were not allowed to be used in the consequent part of the control rules, and the weight parameters of rules were not considered at all.

To compensate for these imperfections, this paper proposes a combining method of fuzzy control and reinforcement learning based on the policy gradient method described in Ref. [11]. Our combining method allows fuzzy sets for describing agent actions in the consequent part, and can also learn the weight parameters of the rules.

This paper is organized as follows: Section 2 describes a policy gradient method to be extended to the fuzzy control system in later sections. Details of the extension are described in Section 3. Learning rules of discrete and continuous membership functions are derived in Sections 4 and 5, respectively. Section 4 also describes the learning rules of the weight parameters of the fuzzy control rules. Section 6 discusses management of the rules and Section 7 is a summary of this paper and our future work.

2. POLICY GRADIENT METHOD

2.1 Policy and Learning Rules

A policy gradient method is a kind of reinforcement learning scheme originated from Williams' REINFORCE algorithm [9]. The method locally increases and maximizes the expected reward by calculating the derivatives of the expected reward function of the parameters included in a stochastic policy function. This method has a firm mathematical basis and is easily applied to many learning problems. It can be used for learning problems even in the non-MDPs by Igarashi et al. [11][13]. In their work, they proposed a policy gradient method that calculates the derivatives of the expected reward function per episode—not per unit time—to maximize the expected reward that does not have Markovian property. The reward r given at time t can depend on not only the current state $s(t)$, but also history $h(t)$, which is a set of all the past states and actions of an agent in an episode. Moreover, this method can be applied to cases with non-Markovian state transition probabilities of the environment and non-Markovian policies of an agent. It has been applied to pursuit problems, where the policy function consists of state-action rules with weight coefficients and potential functions of agent states [13].

Following Ref. [11], let us assume discrete time t, agent state $s(t)$, and agent action $a(t)$. A stochastic policy given by a Boltzmann distribution function,

$$\pi\left(a(t);s(t),h(t),\omega\right) \equiv \frac{e^{-E\left(a(t);s(t),h(t),\omega\right)/T}}{\sum_{a\in A} e^{-E\left(a;s(t),h(t),\omega\right)/T}} \tag{1}$$

controls an agent's action. ω are parameters to be learned and T is a parameter called *temperature*. The learning rule to maximize the expected reward per episode,

$$\Delta\omega = \varepsilon \cdot r \cdot \sum_{t=0}^{L-1} e_{\omega}(t), \tag{2}$$

is shown in Refs. [9] and [11], where L is a time-step size, called the episode length, and ε is a small positive number called the learning rate. $e_{\omega}(t)$ are characteristic eligibilities [9] defined by

$$e_{\omega}(t) \equiv \partial \ln \pi\left(a(t);s(t),h(t),\omega\right)/\partial\omega. \tag{3}$$

Parameters ω are updated at the end of each episode by the learning rules in (2).

2.2 Policy Expressed by If-Then Rules

Let policy π, which determines an agent's action at each time, be expressed by the data base of if-then type rules as "if an agent's state is *s*, then it takes action *a*." In this paper, we deal with only π that does not depend on history $h(t)$. However, discussion in Section 3 and later can be easily extended to non-Markovian policies.

Rule *i* is discriminated by discrete state *s* and discrete action *a* as $i=(s,a)$, and has weight parameter $\theta_i = \theta(s,a)$. If more than one rule matches the current agent state $s(t)$, their firing probabilities depend on the rule weight parameters. Such stochastic policy can be given if objective function $E(a(t);s(t),\theta)$ is defined by

$$E\left(a(t);s(t),\theta\right) = -\theta\left(s(t),a(t)\right). \tag{4}$$

This objective function can be written as [13]

$$E\left(a(t);s(t),\theta\right) = -\sum_{s}\sum_{a}\theta(s,a)\delta_{s,s(t)}\delta_{a,a(t)}, \tag{5}$$

where $\delta_{x,y}$ takes 1 if $x=y$ and 0 otherwise. The right-hand side is considered as an integration method of results of the inference rules. The learning rule of weight parameter $\theta(s,a)$ was derived in [13] as

$$\Delta\theta(s,a) = \varepsilon \cdot r \cdot \sum_{t=0}^{L-1} e_{\theta(s,a)}(t), \tag{6}$$

where

$$e_{\theta(s,a)}(t) = \delta_{s,s(t)}\left[\delta_{a,a(t)} - \pi\left(a;s(t),\theta\right)\right] / T. \tag{7}$$

3. POLICY GRADIENT METHOD EXTENDED TO FUZZY CONTROL SYSTEMS

3.1 Basic Principles

Much work [3-7] has been done combining fuzzy control and reinforcement learning algorithms. A typical method is Fuzzy Q-Learning proposed by Jouffe [3]. However, fuzzy sets were not allowed to describe output variables in the consequent part. Actions must be discretized and Q parameters must be prepared for all actions.

In this paper, we propose an inference system for combining fuzzy control and reinforcement learning based on the following four characteristics:

i) allowing fuzzy-set expressions in both the antecedent part and the consequent part in system control rules;

ii) selecting the output of the system by a stochastic policy;

iii) learning the membership functions of fuzzy sets in both the antecedent part and the consequent part; and

iv) taking account of rule weight parameters and learning them.

The stochastic selection in ii) has been already introduced in [4] rather than determining the output value by the centroid computation that is frequently used, but sometimes reported to bring incorrect and undesirable results. Learning membership functions in the antecedent part is not dealt with in Refs. [3] and [12]. The introduction of rule weights in iv) is for growing or removing control rules by learning.

We consider the following control rules of the system:

Rule i:

if (x_1 is A_1^i) and …and (x_M is A_M^i) then (y_1 is B_1^i) and…and (y_N is B_N^i) with θ_i, (8)

where $x=(x_1,\ldots,x_M)/y=(y_1,\ldots,y_N)$ is the input/output of the control system and corresponds to state *s*/action *a* in reinforcement learning. This paper deals with the case where membership functions $\{A_j^i\}$ and $\{B_j^i\}$ do not depend on each other. Rules do not share an identical membership function. However, the same formalization in this paper is possible and easily extended to cases where multiple rules share an identical membership function with each other.

3.2 Objective Function and Policy

Instead of (4), we propose the following objective function:

$$E\left(y(t);x(t),\theta,A,B\right)=-\sum_{i=1}^{n_R}\theta_i A^i\left(x(t)\right)B^i\left(y(t)\right), \tag{9}$$

where n_R is the number of rules in the rule database and $x(t)/y(t)$ is the input/output of the control system at time *t*. $A^i(x)/B^i(y)$ is the degree of truth value of the antecedent/consequent part in the *i*-th rule and is defined by the products of membership functions $\{A_j^i\}/\{B_j^i\}$ as

$$A^i\left(x\right)\equiv\prod_{j=1}^{M}A_j^i\left(x_j\right) \tag{10}$$

and

$$B^i\left(y\right)\equiv\prod_{j=1}^{N}B_j^i\left(y_j\right). \tag{11}$$

The product in (10)/(11) means that the truth value of the antecedent/consequent part is calculated by the product of the degrees of inclusion in fuzzy sets A_j^i/B_j^i of input/output variable x_j/y_j.

The objective function in (9) indicates how much all rules support output value *y* when *x* is input to the control system. If you compare (9) with (5), θ_i, $A^i(x)$, and $B^i(y)$ in (9) correspond to $\theta(s,a)$, $\delta_{s,s(t)}$, and $\delta_{a,a(t)}$ in (5), respectively. This means that the objective function in (9) is a natural extension of (5) to fuzzy inference systems. Table 1 shows a correspondence relation between the proposed combining method and the policy gradient method described in Refs. [11] and [13].

	Combining method	**Policy gradient[11][13]**
label of variables	fuzzy	non-fuzzy
(a) antecedent part	input $x(t)$	state $s(t)$
(b) consequent part	output $y(t)$	action $a(t)$
rule identifier	i	(s,a)
rule weight	θ_i	$\theta(s,a)$
truth value of (a)	$A^i(x(t))$	$\delta_{s,s(t)}$
truth value of (b)	$B^i(y(t))$	$\delta_{a,a(t)}$

TABLE 1: Correspondence relation between the proposed combining method and the policy gradient method described in Refs. [11] and [13].

The control system determines output $y(t)$ for input $x(t)$ stochastically. The policy for the system is given by a Boltzmann distribution function with the objective function in (9), i.e.,

$$\pi\left(y\left(t\right);x\left(t\right),\theta,A,B\right)\equiv\frac{e^{-E\left(y(t);x(t),\theta,A,B\right)/T}}{\sum_{y}e^{-E\left(y;x(t),\theta,A,B\right)/T}}. \tag{12}$$

Imai et al. applied the policy gradient method in Ref. [13] to pursuit problems [14]. They combined state-action rules from several knowledge resources to speed up learning. In their work, the grid world, where hunter and prey agents move around, was divided into crisp coarse regions. They used the regions as states in state-action rules that control the hunters' actions. A set of state-action rules defined on a different set of regions produces a different knowledge source. A state in the original state space activates rules on the knowledge sources and the inference results of the rules are combined to determine the hunter agents' actions. This work is a special case in our combining method. If $A^i(x)$ in (9) is a binary function that identifies whether an agent's position x is included in the i-th region and B^i(y) in (9) is a binary function that identifies whether an agent's action y is included in an action set {left, right, up, down}, objective function $E(y;x,\theta,A,B)$ in (9) corresponds exactly to the objective function combining rules described by the multiple knowledge sources proposed in Ref. [14].

3.3 Characteristic Eligibilities

Policy π in (12) includes weight θ_i and membership function $A^i(x)/B^i(y)$ in the antecedent/consequent part of the i-th rule. These parameters are denoted by ω. The learning rule of ω is given by (2) and (3) in the policy gradient method. Substituting (12) into (3), characteristic eligibility $e_\omega(t)$ at time t is written as

$$e_{\omega}\left(t\right)=-\frac{1}{T}\left[\frac{\partial E\left(y\left(t\right);x\left(t\right),\theta,A,B\right)}{\partial\omega}-\left\langle\frac{\partial E\left(y;x\left(t\right),\theta,A,B\right)}{\partial\omega}\right\rangle\right], \tag{13}$$

where

$$\left\langle\frac{\partial E\left(y;x\left(t\right),\theta,A,B\right)}{\partial\omega}\right\rangle\equiv\sum_{y}\frac{\partial E\left(y;x\left(t\right),\theta,A,B\right)}{\partial\omega}\pi\left(y;x\left(t\right),\theta,A,B\right). \tag{14}$$

We derive the detailed expression of $e_\omega(t)$ in Sections 4 and 5. In Section 4, input x and output y are discrete variables and, in Section 5, they are continuous.

4. LEARNING RULES

4.1 Rule Weight Parameter θ_i

Characteristic eligibilities $e_\omega(t)$ are obtained by substituting objective function (9) into (13). Substituting the expression of $e_\omega(t)$ into learning rule (2), a learning rule of rule weight parameter θ_i is given as

$$\Delta\theta_i=\varepsilon\cdot r\cdot\sum_{t=0}^{L-1}e_{\theta_i}\left(t\right), \tag{15}$$

where

$$e_{\theta_i}\left(t\right)=A^i\left(x\left(t\right)\right)\left[B^i\left(y\left(t\right)\right)-\left\langle B^i\right\rangle\left(t\right)\right]/T \tag{16}$$

and

$$\left\langle B^i \right\rangle(t) \equiv \sum_y B^i(y)\pi\left(y; x(t), \theta, A, B\right). \tag{17}$$

The meanings of learning rules (15)-(17) are as follows: The degrees of reward r and the truth value in the antecedent part, $A^i(x(t))$, control the amount of update of θ_i. Rule weights that match $x(t)$ very well that were actually input to the control system in an episode are strongly reinforced. The degree of truth value in the consequent part, $B^i(y)$, determines the direction of the update. If the truth value $B^i(y)$ with respect to $y=y(t)$ is stronger than the expectation $\langle B^i \rangle(t)$ defined by (17), the i-th rule's weight θ_i is reinforced. If not, θ_i is suppressed.

This means that rules with large truth values both in matching input $x(t)$ and output $y(t)$ selected by the control system are largely reinforced in episodes where high reward values are given to the system. However, rules that do not match output $y(t)$ actually selected in the episodes are suppressed as the rules produce undesirable output y even if the rules match input $x(t)$ very well.

Let us note that one can easily confirm that Eqs. (16) and (17) lead to Eqs. (6) and (7) by using the correspondences in Table 1. This affirms that the objective function proposed in (9) is a very natural extension from the objective function (4) in the non-fuzzy policy gradient method to one expressed by fuzzy sets.

4.2 Membership Functions in the Antecedent Part

Substituting (9) and (10) into (13), the gradient of the objective function with respect to $A^i_j(x)$, which is a value of a membership function at input x in the antecedent part of fuzzy control rules, and its expectation value are given as

$$\frac{\partial E\left(y(t); x(t), \theta, A, B\right)}{\partial A^i_j(x)} = \frac{\partial}{\partial A^i_j(x)}\left[-\sum_{k=1}^{N}\theta_k A^k\left(x(t)\right)B^k\left(y(t)\right)\right] = -\theta_i \cdot \delta_{x(t),x}\prod_{l\neq j} A^i_l(x)\cdot B^i\left(y(t)\right) \tag{18}$$

and

$$\left\langle \frac{\partial E\left(y; x(t), \theta, A, B\right)}{\partial A^i_j(x)} \right\rangle = -\theta_i \cdot \delta_{x(t),x}\prod_{l\neq j} A^i_l(x)\cdot \left\langle B^i \right\rangle(t). \tag{19}$$

Substituting (18) and (19) into (2) and (13), a learning rule for $A^i_j(x)$ and its characteristic eligibilities are given as

$$\Delta A^i{}_j(x) = \varepsilon \cdot r \cdot \sum_{t=0}^{L-1} e_{A^i{}_j(x)}(t) \tag{20}$$

and

$$e_{A^i{}_j(x)}(t) = \theta_i \cdot \delta_{x(t),x}\prod_{l\neq j} A^i_l(x)\cdot\left[B^i\left(y(t)\right) - \left\langle B^i \right\rangle(t)\right]\Big/T. \tag{21}$$

The meanings of learning rules (20) and (21) are as follows: Only values of $A^i_j(x)$ at values of x actually input to the control system during an episode are updated by the learning rule in (20) and (21). As in the case of learning rule weight parameters discussed in Section 4.1., degrees of reward r, rule weight θ_i, and truth value in the antecedent part, $A^i(x(t))$, control the amount of

update of $A^i_j(x)$. That increases for a large reward, a large rule weight, and how well the rule matches $x(t)$ that appeared in an episode, except for the j-th component $A^i_j(x)$.

The degree of truth value in the consequent part, $B^i(y)$, determines the update direction. If the truth value $B^i(y)$ at $y=y(t)$ is larger than the expectation value $\langle B^i \rangle(t)$ defined by (17), the i-th rule's weight θ_i is reinforced. If not, $A^i_j(x)$ is suppressed.

4.3 Membership Functions in the Consequent Part

Substituting (9) and (11) into (13), the gradient of the objective function with respect to $B^i_j(y)$, which is the value of the membership function at input y in the consequent part of fuzzy control rules, and its expectation are given as

$$\frac{\partial E\left(y(t);x(t),\theta,A,B\right)}{\partial B^i_j(y)} = \frac{\partial}{\partial B^i_j(y)}\left[-\sum_{k=1}^{N}\theta_k A^k\left(x(t)\right)B^k\left(y(t)\right)\right] = -\theta_i A^i\left(x(t)\right)\delta_{y(t),y}\prod_{l\neq j}B^i_l(y) \quad (22)$$

and

$$\left\langle\frac{\partial E\left(y;x(t),\theta,A,B\right)}{\partial B^i_j(y)}\right\rangle = -\theta_i A^i\left(x(t)\right)\prod_{l\neq j}B^i_l(y)\pi\left(y;x(t),\theta,A,B\right). \quad (23)$$

Substituting (22) and (23) into (2) and (13), the learning rules for $B^i_j(y)$ and its characteristic eligibilities are given as

$$\Delta B^i_j(y) = \varepsilon\cdot r\cdot\sum_{t=0}^{L-1}e_{B^i_j(y)}(t) \quad (24)$$

and

$$e_{B^i_j(y)}(t) = \theta_i\cdot A^i\left(x(t)\right)\cdot\left[\delta_{y(t),y}-\pi\left(y;x(t),\theta,A,B\right)\right]\prod_{l\neq j}B^i_l(y)\Big/T \quad . \quad (25)$$

The meanings of learning rules (24) and (25) are as follows: As in the case of learning $A^i_j(x)$, degrees of reward r, rule weight θ_l, and the truth value in the antecedent part controls the amount of update of $B^i_j(y)$. In addition, the degree of truth value in the consequent part, except j-th component $B^i_j(y)$, also controls the amount. Therefore, it increases for a large reward, a large rule weight, and how well the rule matches $x(t)$ and $y(t)$, except the j-th component that appeared in an episode. Moreover, $B^i_j(y)$ is reinforced if $y=y(t)$, while $B^i_j(y)$'s at values of output y that competed against $y(t)$ are all suppressed.

5. LEARNING RULES OF PARAMETERS IN CONTINUOUS MEMBERSHIP FUNCTIONS

Learning rules in Section 4 are derived under an assumption that input x and output y are discrete variables. In this section, we derive learning rules when x and y are continuous. In such a case, membership functions $A^i_j(x)$ and $B^i_j(y)$ are continuous functions of x and y. For example, we consider the membership functions as

$$\mu(x;b,c,m) \equiv 1\Big/\left[1+b\left(x-c\right)^m\right], \quad (26)$$

which are frequently used in fuzzy control. In (26), m is an even integer and the targets of learning are parameters b (>0) and c.

Now, membership functions $A^i_j(x)$ and $B^i_j(y)$ are continuous functions, shown in (26), and have parameters α^i_j and β^i_j for *b* and *c* in (26). We define an objective function as

$$E\left(y(t);x(t),\theta,A,B\right)=-\sum_{i=1}^{n_R}\theta_i A^i\left(x(t);\alpha^i\right)B^i\left(y(t);\beta^i\right), \tag{27}$$

where

$$A^i\left(x;\alpha^i\right)\equiv\prod_{j=1}^{M}A^i_j\left(x_j;\alpha^i_j\right) \tag{28}$$

and

$$B^i\left(y;\beta^i\right)\equiv\prod_{j=1}^{N}B^i_j\left(y_j;\beta^i_j\right). \tag{29}$$

Substituting (27) into (13) and (14), characteristic eligibilities are obtained as

$$e_{\alpha^i_j}(t)=(1/T)\cdot\theta_i A^i\left(x(t);\alpha^i\right)\cdot\partial\ln A^i_j\left(x_j(t);\alpha^i_j\right)\Big/\partial\alpha^i_j\cdot\left[B^i\left(y(t);\beta^i\right)-\left\langle B^i\left(y;\beta^i\right)\right\rangle(t)\right] \tag{30}$$

and

$$e_{\beta^i_j}(t)=(1/T)\cdot\theta_i A^i\left(x(t);\alpha^i\right)\cdot \left[\prod_{l\neq j}B^i_l\left(y(t);\beta^i\right)\partial B^i_j\left(y_j(t);\beta^i_j\right)\Big/\partial\beta^i_j-\left\langle\prod_{l\neq j}B^i_l\left(y;\beta^i\right)\partial B^i_j\left(y_j;\beta^i_j\right)\Big/\partial\beta^i_j\right\rangle(t)\right], \tag{31}$$

where

$$\left\langle f(y)\right\rangle(t)\equiv\int_{-\infty}^{+\infty}f(y)\pi\left(y;x(t),\theta,A,B\right)dy. \tag{32}$$

Characteristic eligibilities of rule weight θ_i are given by (16), but (17) is replaced by (32). That is

$$e_{\theta_i}(t)=A^i\left(x(t)\right)\left[B^i\left(y(t)\right)-\left\langle B^i\right\rangle(t)\right]\Big/T \tag{33}$$

and

$$\left\langle B^i(y)\right\rangle(t)\equiv\int_{-\infty}^{+\infty}B^i(y)\pi\left(y;x(t),\theta,A,B\right)dy. \tag{34}$$

Learning rules of α^i_j, β^i_j, and θ_i are given by substituting (30), (31), and (33) into (2) as

$$\Delta\alpha^i_j=\varepsilon\cdot r\cdot\sum_{t=0}^{L-1}e_{\alpha^i_j}(t), \tag{35}$$

$$\Delta\beta_j^i = \varepsilon \cdot r \cdot \sum_{t=0}^{L-1} e_{\beta_j^i}(t) \tag{36}$$

and

$$\Delta\theta_i = \varepsilon \cdot r \cdot \sum_{t=0}^{L-1} e_{\theta_i}(t). \tag{37}$$

6. MANAGEMENT OF RULES

In this section, we consider the management of fuzzy rules in the policy. Rule management means how to remove unnecessary rules, generate new rules, and merge two rules.

Rules should be removed if their weight parameters are decreased to near zero by learning. Rules whose truth values in the antecedent part and the consequent part are constantly very small in all episodes are also to be removed from the database of fuzzy control rules in the policy. If all rules cannot match input x, then a good idea generates a fuzzy rule whose antecedent part includes a fuzzy set whose center locates at x [15].

There are two ideas for merging two rules into one rule. First, if there is a strong positive correlation in the truth value of the antecedent and the consequent part between two rules, then remove one of the two rules. Second, if there is a strong positive correlation in the truth value of the consequent part between two rules and their membership functions $A^i(x)$ are adjacent to each other, then one can merge their membership functions into a new $A^i(x)$ and replace the two rules with a new one.

7. CONCLUSION

This paper has extended a policy gradient method with weight parameters of *if-then* type controlling rules in the objective function to a case where the rules are described by fuzzy sets. Learning rules of membership functions and rule weights are derived for both cases when input/output variables to/from the control system are discrete and continuous.

In the future, we plan to verify the learning rules derived here by applying them to examples such as robot soccer games [16] and car velocity control problems. Moreover, we will try to extend the theory proposed here to multistage fuzzy inference, multilayer control systems, and multiagent systems.

8. REFERENCES

[1] R. R. Yager and L. A. Zadeh. *An Introduction to Fuzzy Logic Applications in Intelligent Systems*. Norwell, MA, USA: Kluwer Academic Publishers, 1992.

[2] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[3] L. Jouffe. “Fuzzy Inference System Learning by Reinforcement Methods.” *IEEE Transactions on Systems*, *Man*, *and Cybernetics*, vol. 28, No. 3, pp. 338-355, 1998.

[4] C. Oh, T. Nakashima, and H. Ishibuchi. “Initialization of Q-values by Fuzzy Rules for Accelerating Q-learning.” in Proc. IEEE World Congress on Computational Intelligence, vol. 3, 1998, pp. 2051-2056.

[5] T. Horiuchi, A. Fujino, O. Katai, and T. Sawaragi. “Fuzzy Interpolation-based Q-learning with Continuous States and Actions,” in Proc. the Fifth Inter. Conf. on Fuzzy Systems, 1996, vol. 1, pp. 594-600.

[6] Y. Hoshino and K. Kamei. “A Proposal of Reinforcement Learning with Fuzzy Environment Evaluation Rules and Its Application to Chess.” *J. of Japan Society for Fuzzy Theory and Systems*, vol. 13, no. 6, pp. 626-632, 2001. (in Japanese)

[7] H. R. Berenji. “A Reinforcement Learning-based Architecture for Fuzzy Logic Control.” *Int. J. Approx. Reasoning*, vol. 6, pp. 267–292, 1992.

[8] H. R. Berenji and D. Vengerov. “Cooperation and Coordination Between Fuzzy Reinforcement Learning Agents in Continuous State Partially Observable Markov Decision Processes,” in 1999 IEEE Int. Fuzzy Systems Conf. Proc., 1999, vol. 2, pp. 621-627.

[9] R.J. Williams. “Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning.” *Machine Learning*, vol. 8, pp. 229-256, 1992.

[10] J. Baxter and P. L. Bartlett. “Infinite-Horizon Policy- Gradient Estimation,” *Journal of Artificial Intelligence Research*, vol. 15, pp. 319-350, 2001.

[11] H. Igarashi, S. Ishihara, and M. Kimura. “Reinforcement Learning in Non-Markov Decision Processes-Statistical Properties of Characteristic Eligibility.” *IEICE Transactions on Information and Systems*, vol. J90-D, no. 9, pp. 2271-2280, 2007. (in Japanese)

(This paper is translated into English and included in *The Research Reports of Shibaura Institute of Technology, Natural Sciences and Engineering*, vol. 52, no. 2, pp. 1-7, 2008.)

[12] X. Wang, X. Xu, and H. He. “Policy Gradient Fuzzy Reinforcement Learning,” in Proc. 3rd Inter. Conf. on Machine Learning and Cybernetics, 2004, pp. 992-995.

[13] S. Ishihara and H. Igarashi, “Applying the Policy Gradient Method to Behavior Learning in Multiagent Systems: The Pursuit Problem.” *Systems and Computers in Japan*, vol. 37, no. 10, pp. 101-109, 2006.

[14] S. Imai, H. Igarashi, and S. Ishihara. “Policy-Gradient Method Integrating Abstract Information in Policy Function and Its Application to Pursuit Games with a Tunnel of Static Obstacles.” *IEICE Transactions on Information and Systems*, vol. J94-D, no. 6, pp. 968-976, 2011. (in Japanese).

This paper is translated into English and included in *The Research Reports of Shibaura Institute of Technology, Natural Sciences and Engineering*, vol. 52, no. 2, pp. 7-12, 2011.

[15] Y. Hosoya, T. Yamamura, M. Umano, and K. Seta. “Reinforcement Learning Based on Dynamic Construction of the Fuzzy State Space-Adjustment of Fuzzy Sets of States-,“ in Proc. of the 22nd Fuzzy System Symposium (CD-ROM), vol. 22, 8D3-1, 2006. (in Japanese).

[16] M. Sugimoto, H. Igarashi, S. Ishihara, K. Tanaka. “Policy Gradient Reinforcement Learning with a Fuzzy Controller for Policy: Decision Making in RoboCup Soccer Small Size League,” presented at the 29th Fuzzy System Symposium, Osaka, Japan, 2013. (in Japanese).

INSTRUCTIONS TO CONTRIBUTORS

The main aim of International Journal of Artificial Intelligence and Expert Systems (IJAE) is to provide a platform to AI & Expert Systems (ES) scientists and professionals to share their research and report new advances in the field of AI and ES. IJAE is a refereed journal producing well-written original research articles and studies, high quality papers as well as state-of-the-art surveys related to AI and ES. By establishing an effective channel of communication between theoretical researchers and practitioners, IJAE provides necessary support to practitioners in the design and development of intelligent and expert systems, and the difficulties faced by the practitioners in using the theoretical results provide feedback to the theoreticians to revalidate their models. IJAE thus meets the demand of both theoretical and applied researchers in artificial intelligence, soft computing and expert systems.

IJAE is a broad journal covering all branches of Artificial Intelligence and Expert Systems and its application in the topics including but not limited to technology & computing, fuzzy logic, expert systems, neural networks, reasoning and evolution, automatic control, mechatronics, robotics, web intelligence applications, heuristic and AI planning strategies and tools, computational theories of learning, intelligent system architectures.

To build its International reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJAE.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Started with Volume 4, 2013, IJAE appears with more focused issues related to artificial intelligence and expert systems studies. Besides normal publications, IJAE intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

We are open to contributions, proposals for any topic as well as for editors and reviewers. We understand that it is through the effort of volunteers that CSC Journals continues to grow and flourish.

LIST OF TOPICS

The realm of International Journal of Artificial Intelligence and Expert Systems (IJAE) extends, but not limited, to the following:

- AI for Web Intelligence Applications
- AI Parallel Processing Tools
- AI Tools for Computer Vision and Speech Understand
- Application in VLSI Algorithms and Mobile Communication
- Case-based reasoning
- Derivative-free Optimization Algorithms
- Evolutionary and Swarm Algorithms
- Expert Systems Components
- Fuzzy Sets and logic
- Hybridization of Intelligent Models/algorithms
- Inference
- Intelligent Planning
- AI in Bioinformatics
- AI Tools for CAD and VLSI Analysis/Design/Testing
- AI Tools for Multimedia
- Automated Reasoning
- Data and Web Mining
- Emotional Intelligence
- Expert System Development Stages
- Expert-System Development Lifecycle
- Heuristic and AI Planning Strategies and Tools
- Image Understanding
- Integrated/Hybrid AI Approaches
- Intelligent Search

- Intelligent System Architectures
- Knowledge-Based Systems
- Logic Programming
- Multi-agent Systems
- Neural Networks for AI
- Parallel and Distributed Realization of Intelligence
- Reasoning and Evolution of Knowledge Bases
- Rule-Based Systems
- Uncertainty
- Knowledge Acquisition
- Knowledge-Based/Expert Systems
- Machine learning
- Neural Computing
- Object-Oriented Programming for AI
- Problem solving Methods
- Rough Sets
- Self-Healing and Autonomous Systems
- Visual/linguistic Perception

CALL FOR PAPERS

Volume: 4 - **Issue: 2**

i. Paper Submission: July 30, 2013 **ii. Author Notification:** September 15, 2013

iii. Issue Publication: October 2013

CONTACT INFORMATION

Computer Science Journals Sdn BhD
B-5-8 Plaza Mont Kiara, Mont Kiara
50480, Kuala Lumpur, MALAYSIA

Phone: 006 03 6207 1607
006 03 2782 6991

Fax: 006 03 6207 1697

Email: cscpress@cscjournals.org

CSC PUBLISHERS © 2013
COMPUTER SCIENCE JOURNALS SDN BHD
B-5-8 PLAZA MONT KIARA
MONT KIARA
50480, KUALA LUMPUR
MALAYSIA

PHONE: 006 03 6207 1607
006 03 2782 6991

FAX: 006 03 6207 1697
EMAIL: cscpress@cscjournals.org