Co-Operative Co-Evolutionary System for Solving Dynamic VRPTW Problems with Crisis Situations

Rafał Dreżewski, Łukasz Dronka, Jarosław Koźlak

Department of Computer Science
AGH University of Science and Technology, Kraków, Poland
{drezew,kozlak}@agh.edu.pl

Abstract. Vehicle routing problems with time windows are NP-hard problems. Additional difficulties are introduced by dynamic client requests and crisis situations. One of the techniques used in order to solve such problems are evolutionary algorithms. In this paper a co-evolutionary algorithm with spatial population structure is presented. The system is verified with the use of dynamic vehicle routing problems with time windows and crisis situations.

1 Introduction

In the current economic climate, transportation of cargo and delivering services to the homes of clients is playing a very important and ever-increasing role. One of the most widely researched transport problems today is the Vehicle Routing Problem with Time Windows (VRPTW). The idea of the problem is to realize a given set of transport requests while maintaining the lowest possible costs with the number of used vehicles and the total distance travelled. Each vehicle is described by: a location which has to be visited, a period of time when this location should be visited (called *time window*), a capacity needed for transporting the parcel picked-up/delivered from/to a given location. The vehicles are described by their total capacity (the total size of parcels that may be transported at the same time) and a speed. The formal model of VRPTW problem may be found, for example, in [14].

VRPTW is a NP-hard problem [9]. It means, that the time needed to solve it using algorithms finding strict solutions grows exponentially with the size of the problem. For problems like this approximate solutions which could be found fast enough and would be strict enough need to be searched. NP-hard problems can be solved with the application of various heuristics and meta-heuristics—the point is that they make use of problem internal properties.

It is possible to distinguish two kinds of VRPTW: static and dynamic. In static, all transport requests are known before starting the process of assigning them to the vehicles, while in the dynamic case, requests arrive continuously during the simulation process. In such case the modeling of the respective vehicle's location and the status of realizing requests is necessary.

For the analysis of the practical problems associated with the planning and realization of transport requests an important element is taken into consideration, namely, the occurrence of different crisis situations and an attempt trying to limit their consequences. Crisis situations analyzed in this paper, are the following: traffic jams appearing on the routes of vehicles, vehicle breakdowns, delays of starting the request realization, prolonging the time of request realization in a given customer location and premature time windows closing.

2 Previous Research on Dynamic VRPTW Problems and Crisis Situations

The VRPTW has been the focus of analysis by many researchers because of its far-reaching and wide range of practical application. Especially, a particularly high number of solutions of its static version have been worked out. To verify the quality of algorithms, sets of transport requests tests, prepared by Solomon and extended by Gehring and Homberger [1], are used.

The heuristic methods used are usually made up of two stages. In the first one, the initial solutions are generated using different kinds of construction heuristics (for example route first cluster second, saving method, I1 heuristic, parallel I1 heuristic, sweep heuristic). Then, optimization heuristics are used to improve the initial solutions. Different approaches are used, e.g. tabu search, simulated annealing, evolutionary algorithms or ant colony approach. The algorithm proposed by Gehring and Homberger [7], based on evolutionary strategies, obtained especially good results. The overview of best VRPTW solving metaheuristic is presented in [4].

The dynamic versions of VRPTW were also studied [8, 3], but less work was done than for static version. In [3] an analysis of problem having mixed static-dynamic features and methods of request generation for such problem as well as measures of its dynamic degree is described.

When modeling crisis situations the the most often researched are the appearance of traffic jams, for this case a multi-agent approach was applied [6].

The analysis of the consequences of critical situations is also the subject of this work. The paper [5] contains an overview of various systems for solving VRPTW and PDPTW (Pickup and Delivery Problem with Time Windows—this problem is similar to the mentioned VRPTW, it is characterized by the fact that with each transport request two locations are associated: the location of pickup and the location of delivery, and these locations are assigned separate time windows) developed in our research group. The pilot version of the platform presented in this paper and the preliminary results of experiments (without crisis situations) obtained with the use of the platform were also described in [5].

3 Algorithms for Solving dynamic VRPTW Problems

In the system presented in this paper, co-operative co-evolutionary algorithm (CCA) [11] is used to solve dynamic VRPTW problem. The basic CCA algorithm may be described by the following pseudo-code:

```
For each subpopulation S Do:
   Initialise population Ps(0)
   Evaluate all individuals from Ps(0) (by forming groups
      composed of the given individual from S and the chosen
      representants of all other subpopulations)
End_For
While termination condition not met Repeat:
   For each subpopulation S Do:
      Select a set of parents Xs(t) for next generation
      Apply genetic operators to the individuals of Xs(t)
         obtaining a set of descendants Ds(t)
      Evaluate individuals from Ds(t) (in the same way
         as in the case of Ps(0))
      Combine Ps(t) and Ds(t) obtaining Ps(t+1)
   End_For
End_While
```

The algorithm is based on the co-evolutionary algorithm for VRPTW problems proposed in [10]. In such an approach two subpopulations (species) are used. Individuals from the first subpopulation represent the number of clients in each route. Information encoded in the genotypes of individuals from the second subpopulation controls which clients, and in what order, appear in every single route. An individual from the first subpopulation is correct if the sum of all his genes' values is greater or equal to the number of clients. An individual from the second subpopulation is correct if it contains permutation of all clients. Combination of two individuals coming from opposite subpopulations results in a complete solution. The reverse of this process is a separation of a complete solution, which results in two individuals from the opposite subpopulations.

Figure 1 illustrates two individuals from the opposite subpopulations. A value of the first gene of individual from the first subpopulation is the number of clients of first vehicle's route. As we can see, the route of the first vehicle consists of three clients and the first three genes of the individual from the second subpopulation strictly describe the shape of the route: $\{0, 3, 2, 7, 0\}$. The other routes are created in the analogical way. The data contained in the genotype of the individual from the second subpopulation is used to create three routes only. However this is not a problem, because a complete solution is created. Genes which are not used in the complete solution are considered being redundant and they are ignored.

Because it is not possible to predict the number of routes in advance, it is assumed, that the number of genes of individuals from the first subpopulation is half of the number of clients. A maximum length of a route is also limited to this

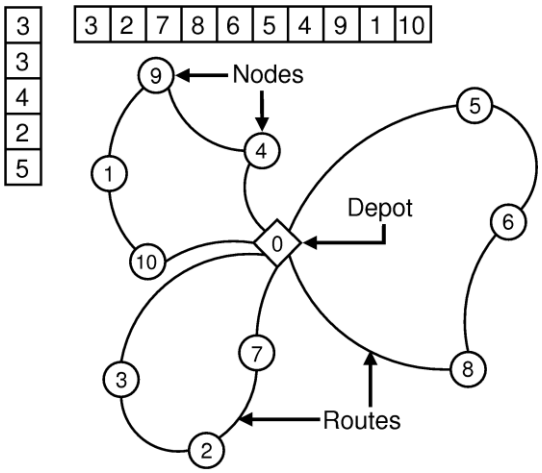


Fig. 1. The individuals from the first (left) and the second (top) subpopulation. Graph illustrates the solution, which is the result of combining given individuals [10]

value. With these assumptions, it is possible, that a shortage of clients during the process of complete solution constructing will occur. In this case the strategy of filling up missing genes with random numbers from the range $\langle 2; n/2 \rangle$ (where n is the number of clients) is used.

The co-evolutionary algorithm described above was used together with an island model parallel evolutionary algorithm [2]. In such a model the whole population of individuals is divided into subpopulations living on different islands (computation nodes). The individuals can migrate between islands. In our case also different fitness function on each island is used.

4 Prediction Techniques for Preventing and Avoiding Crisis Situations

The function approximation approach is used as the main prediction technique in the presented system. The goal of approximation is finding dependence between the analyzed continuous attributes and other continuous attributes of the analyzed phenomenon. After computing such a dependence we are able to predict a value of interesting attributes for new sets of attributes of the analyzed phenomenon. The application of this method is based on the assumption that the occurrence and length of crisis situations probably depends on a well formed set of continuous parameters. The key issue in obtaining a good quality of prediction with the use of approximation techniques is the selection of an appropriate set of parameters. Both ignoring important parameters and taking into account unimportant parameters often results in a slow learning process and the decrease in prediction quality. In real conditions, correlation should be analyzed to define which parameters are significant.

We tested two function approximators:

- Multilayer perceptron with a backpropagation algorithm [12]. The number of artificial neurons in the input layer is equal to the number of continuous parameters, on which a crisis depends. Single hidden layer consists of 15 neurons with sigmoid activation function. The output layer consists of a single neuron with a sigmoid activation function—the output is a predicted crisis length. Learning rate decreases while the learning process progresses.
- Tile Coding [13]. The number of tile dimensions is equal to the number of continuous parameters, on which the crisis depends. The size of tiles and number of tilings depend on the type of crisis—both the requirement of strict predictions and memory limitations are taken into consideration. In order to update function values for tiles the delta rule is used [15]:

$$w_i(x) = w_i(x) + \beta(value - w_i(x)) \tag{1}$$

where $w_i(x)$ is the function value for tile, β is the learning rate, and the $value$ is the value returned by environment.

5 System for Solving Dynamic VRPTW Problems with Crisis Situations

The system presented in this paper was implemented with the use of more general, Java based component platform for solving VRP problems. The platform consists of several components, of which the most important are:

- Static problems component.
- Dynamic problems simulation component.
- Soft time windows component—soft time windows allows us to expand time windows for clients and depot. Expanding time windows results in penalties during solutions evaluation.
- Crisis situations simulation component.
- Computational component—algorithms for solving VRP problems are loaded as plugins. Although for this platform there were mostly implemented evolutionary algorithms, the component architecture of the platform allows us to add also non-evolutionary algorithms, like tabu search, etc.
- Operators loaded as plugins—operators are all processes, which input and output is an individual (or a set of individuals). These include all genetic operators like recombination, mutation, etc.

In all experiments presented in this paper the CCA algorithm with the island model—as described in section 3—was used as the computational component (see fig. 2). In order to exchange the best individuals between islands the set of individuals from each island was copied to repository. Individuals from the repository were then sent to other islands, where they replaced the worst individuals. Computations on distinct islands can take place on a single machine or on several network connected machines. Distributed computing uses Java RMI technology.

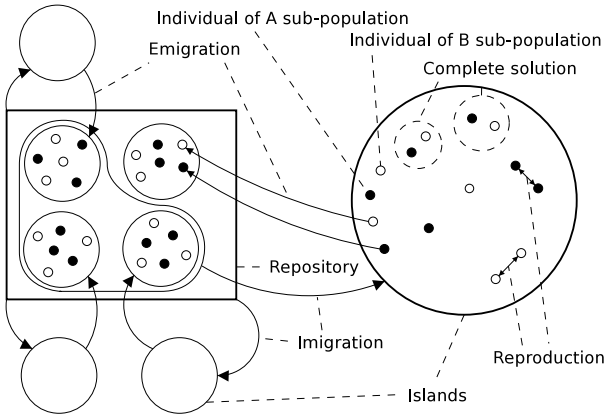


Fig. 2. Computational component: co-evolutionary algorithm and island model

5.1 Dynamic Problems Simulation

The component for dynamic problem simulation realizes an event-driven simulation model with discrete time. Dynamic problems are generated on the basis of static Solomon's test problems. Part of the clients' service requests are declared before the beginning of the simulation—the remaining requests are declared during simulation in two sets of requests. First one arrives when the 1/3 of the simulation has passed and the second when 2/3 of the simulation has passed.

The way of solving the dynamic problem is performing alternate simulations of vehicle movement and the static problem planning algorithm for the set of unserved clients. During the simulation of vehicle movement, vehicle locations and cargo amounts are updated. This updated information is then taken into account during the planning of the next static solution. Clients already served are removed from the set of unserved clients during the simulation of vehicle movement.

The solution of dynamic problem can also be described by the following pseudo-code:

```
Schedule requests declarations
The unserved clients set is empty
While(there are undeclared requests in the schedule)
   Declare requests
   Add clients declaring requests to the unserved clients set
   Run the static problem planning algorithm for
      the set of unserved clients
   Simulate vehicles movement
End_while
```

5.2 Simulation of Crisis Situations

Five types of crisis situations were implemented within the component responsible for simulating such situations. These situations differ from each other in the following ways:

- Place where crisis situations appear—clients or sections of routes between clients.
- Parameters, on which probability of the occurrence of crisis situations and its length depends. Dependence between particular types of crisis situations and their parameters is essential to predict the crisis situations properly. Selecting a set of parameters for the given type of crisis situation means, that there is a high probability, that the crisis situation of a particular type depends on every parameter of the chosen set;
- Time of the crisis situation occurring.

Implemented crisis situations, include (only selected are described in more details):

- Occurrence of traffic jams within some sections of the route, between two clients—depends on two discrete parameters: clients, which identify the particular section of route and one continuous parameter: occurrence time (we assume that the occurrence of traffic jams depends on the time of the day).
- Vehicle breakdowns.
- Delayed readiness of the client (time window opening delayed).
- Prolongation of the service time—depends on one discrete parameter: a particular client, and two continuous parameters: the cargo weight to unload and the occurrence time.
- Premature time window closure.

For each set of values of discrete parameters of crisis, separate crisis length generators were created and used. For each generator and for each continuous parameter a random function was selected and used to compute the probability of crisis situation occurrence and its length. Several types of functions can be used, for example trigonometric, linear, or power with random parameters. The random parameters of the function were selected in such a manner that the function had values from the range $\langle 0, 1 \rangle$ in the whole domain (possible values of crisis' continuous parameter). We assume that the probability of a crisis situation occurring is equal to the geometric mean of crisis' continuous parameters' values. The length of the crisis is equal to the geometric mean of the crisis' continuous parameters' values multiplied by the absolute value of a random number from the normal distribution and some chosen factor.

For each set of values of crisis discrete parameters separate approximators were created and used. Attributes of approximation are continuous parameters of crisis and the crisis length. The latter one is attributed to the prediction. During the phase of vehicles movement simulation, each time when a crisis situation occurring is possible, the crisis length is computed in the manner described in the previous paragraph. When a crisis does not occur it is set to zero. A crisis length

is used to correct the parameters of simulation (eg. traffic jam—time of passing the road is made longer). Then the approximator responsible for predicting the crisis length for the crisis type and the set of values of discrete parameters is determined. The chosen approximator is trained: training attributes are the set of values of continuous parameters of crisis and computed crisis length.

The use of separate approximators for each set of values of discrete parameters of a crisis disqualifies the trivial assumption that there is a straight road between every pair of clients. When considering traffic jams, it would mean that we have to create squared clients number of approximators. Thus we provide the possibility of limiting the number of roads in such a way that the road net is the connected graph (accessibility of all clients from the depot is assured). The crisis situations component includes the algorithm for generating the set of roads between clients in such a way that the distance between any particular two clients is close to a straight road length. The predicted delay caused by the traffic jams between two particular clients is the sum of predicted delays caused by traffic jams on all roads between the mentioned clients.

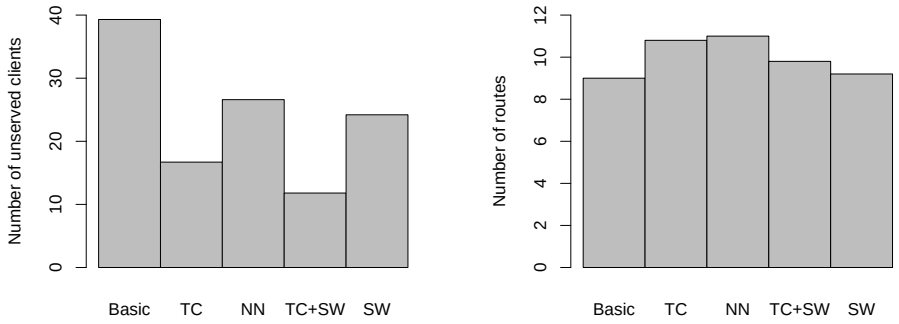
Two methods of avoiding crisis situations with the application of predictors were implemented:

- Correction of routes with the use of predicted crises' lengths during the stage of combining two individuals from opposite populations into a single solution—clients, where a successful service is doubtful because of the high risk of crisis situations occurring are moved to another position in the route or even to another route.
- Correction of fitness function values with the use of predicted crises' lengths—fitness value for a solution, where the risk of the crisis situation occurring is high, is decreased.

6 Experimental Results

The most significant consequence of the crisis situations occurring is the appearance of vehicle delays. Because of these, vehicles arrive at the client later than the planned algorithm assumed, often after closing the clients' time windows. In the last case the client is not served and it greatly decreases the solution quality. For this reason the number of unserved clients is considered as a main optimization parameter—apart from the number of routes. Experiments have been carried out on the set of dynamic problem tests with the diversified dynamism level, generated on the basis of Solomon's static problems set.

Results for R2 Solomon's problem class are listed in fig. 6. Very good results have been achieved with Tile Coding predictor (TC). The number of unserved clients has been reduced by more then half (when compared to algorithm without any predictor—"Basic") and only the small increase of the route numbers has been observed. Good results have also been achieved with the use of soft time windows (SW). In this case, time windows are expanded by 50% (SW = 0.5) of its initial width. The best results however can be observed in the case of using Tile Coding predictor and soft time windows simultaneously (TC+SW) with SW



a) b)

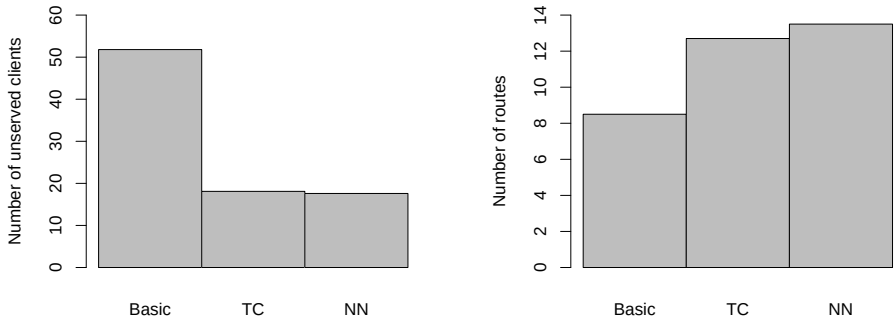
Fig. 3. Average results of 33 experiments carried out on every configuration, for R2 Solomon's problem class: unserved clients (a), and routes (b) for different prediction techniques

= 0.5. This results in over three times reduced values of the unserved number of clients parameter.

Worse results than in the case of using Tile Coding predictor have been achieved with the use of the neural network predictor (NN). The number of unserved clients was reduced by about 30%. This is a result of the fact that the crisis situations generators generate very dispersed crises lengths (values of approximators target functions). If most of these lengths are zeros then the crisis situations do not appear. The neural network often learns incorrectly in the case of such an input data and then it approximates target functions erroneously. The neural network learns much better when input data is less dispersed. In order to verify this thesis the set of experiments was carried out. Crisis generators were simplified in such a manner that crisis situations always appear, but factors used to compute crisis length were reduced by 50%. Results of such experiments are presented in fig. 6.

Results show that Tile Coding and neural network approximation qualities are similar. Although neural network appears to be a considerably less versatile approximator than Tile Coding, it has an important advantage—it does not demand as much memory as the Tile Coding, which is the property that could be decisively significant when function with many continuous parameters is approximated.

Results for other Solomon classes are listed in the table 1. It appears that the most efficient and the most versatile method of avoiding crisis situations is the application of Tile Coding approximation to predict crisis' lengths in order to correct routes and fitness function values. This technique has achieved best



a) b)

Fig. 4. Average results of 33 experiments carried out on every configuration, for R2 Solomon's problem class: unserved clients (a), and routes (b) for different prediction techniques. Crisis situation always appears

effects for each experiment configuration, resulting in 46%–60% (it depends on Solomon's problem class) reduction of the number of unserved clients and the insignificant increase of the number of routes.

Good results have been also achieved with the use of soft time windows, resulting in 46%–60% reduction of the number of unserved clients (when the time window was expanded by half of its initial width—SW = 0.5) and 25%–47% (when the time window initial width was doubled—SW = 1). Additionally, described method results in the route number reduction, because it expands the interval of time in which the client can be served and thus relaxes the constraints of the problem. It is worth paying attention to the worse results achieved in the case of experiments with Solomon's class 1 problems in comparison to corresponding Solomon's class 2 problems. The reasons of such behavior are narrower time windows in Solomon's class 1 problems. Because of the assumption that the possibility of expanding time window depends on its initial width, narrow time windows are less extensible. Despite the good results being achieved, it is worth remembering, that expanding time windows is the simplification of the problem conditions and it is harmful for clients. That is the reason why these results are considered inferior to others.

7 Concluding Remarks

In the paper the system for solving dynamic vehicle routing problems with time windows and crisis situations was presented. The results of experiments, carried out on test sets for dynamic problems (generated by implemented algorithm on

Table 1. Average results of 30 experiments carried out on every configuration, classified by Solomon's problem classes

Class		Basic	TC	SW = 0.5	SW = 1
C1	Number of routes	15.3	19.9	13.9	13
	Number of unserved clients	32.1	14.3	26.5	23.8
C2	Number of routes	9.9	11.6	9.7	9.8
	Number of unserved clients	30	12.1	22.7	18.6
R1	Number of routes	23.6	27.4	17.8	15.6
	Number of unserved clients	39.7	20.6	32.2	29.5
RC1	Number of routes	23.9	26.9	19.1	16.2
	Number of unserved clients	35.8	16.9	29.4	26.9
RC2	Number of routes	10.4	11.4	9.6	9.3
	Number of unserved clients	31.5	17	20.3	16.6

the basis of the well-known and frequently used Solomon's static problems set) have proved that methods of avoiding crisis situations can limit the negative influence on the quality of the VRPTW dynamic problem solution. The main goal of experiments was to verify the quality of implemented techniques of predicting and avoiding crisis situations. The presented results clearly show, which one of the methods is the best and the most versatile. Additionally they determine, in what conditions other methods, less successful in a general case, could be used.

Of course these are only preliminary results and further research and experiments are needed in order to additionally verify proposed mechanisms and compare them to other prediction techniques. Also the agent-based realization of the presented system is included in the future plans. It seems that such decentralized co-evolutionary multi-agent system would have even stronger adaptive capabilities what could be the great advantage—especially in the case of hard dynamic real-life problems.

References

1. Benchmarks - vehicle routing and travelling salesperson problems. http://www.sintef.no/static/am/opti/projects/top/.
2. P. Adamidis. Parallel evolutionary algorithms: A review. In *Proceedings of the 4th Hellenic-European Conference on Computer Mathematics and its Applications (HERCMA 1998)*, Athens, Greece, 1998.
3. R. Bent and P. Hentenryck. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operational Research*, 52:977–987, 6 2004.

4. O. Braysy. Genetic algorithms for the vehicle routing problem with time windows. *Arpakannus. Special issue on Bioinformatics and Genetic Algorithms*, pages 33–38, 1 2001.
5. R. Dreżewski, M. Kisiel-Dorohinicki, and J. Koźlak. Agent-based and evolutionary planning techniques supporting crises management in transportation systems. In A. Dolgui, G. Morel, and C. Pereira, editors, *Proceedings of 12th IFAC symposium on INformation COntrol problems in Manufacturing (INCOM'2006)*, volume 1, Saint-Etienne, France, May 2006. Ecole Nationale Supérieure des Mines de Saint-Etienne and IFAC.
6. K. Fischer, J. Muller, and M. Pischel. Cooperative transportation scheduling: an application domain for DAI. *Applied Artificial Intelligence*, 10:1–33, 1996.
7. J. Homberger and H. Gehring. Two evolutionary meta-heuristics for the vehicle routing problem with time windows. *INFORMS Journal in Computing*, 37:297–318, 3 1999.
8. A. Larsen. The dynamic vehicle routing problem. PhD thesis, Department of Mathematical Modelling, The Technical University of Denmark, 2000.
9. J. K. Lenstra and A. R. Kan. Complexity of Vehicle Routing and Scheduling Problems. *Networks*, 11:221–227, 1981.
10. P. Machado, J. Tavares, F. B. Pereira, and E. Costa. Vehicle routing problem: Doing it the evolutionary way. In W. B. Langdon, et al., editor, *Proceedings of the 2002 Genetic and Evolutionary Computation Conference*, San Francisco, CA, 2002. Morgan Kaufmann.
11. M. A. Potter and K. A. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, 2000.
12. D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: Foundations*. MIT Press, 1986.
13. R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.
14. S. Thangiah. Vehicle routing with time windows using genetic algorithms. In *Application Handbook of Genetic Algorithms: New Frontiers, Volume II.*, pages 253–277. CRC Press, 1995.
15. B. Widrow and M. E. Hoff. Adaptive switching circuits. In *Neurocomputing: foundations of research*. MIT Press, 1988.