International Journal of Innovative Computing, Information and Control
Volume **8**, Number **8**, August **2012**

ICIC International ©2012 ISSN 1349-4198
pp. 5859–5872

A KNOWLEDGE REPRESENTATION METHOD FOR ALGORITHMS IN DSS FOR REAL-TIME VEHICLE ROUTING IN URBAN DISTRIBUTION

Lijun Sun and Xiangpei Hu

Institute of Systems Engineering
Faculty of Management and Economics
Dalian University of Technology
No. 2, Linggong Road, Ganjingzi District, Dalian 116024, P. R. China
{ slj; drhxp }@dlut.edu.cn

Received March 2011; revised August 2011

Abstract. *A knowledge representation method for local search algorithms that are suitable for solving real-time vehicle routing problems in urban distribution is proposed in the research. The first part of this method is using rules to represent policies obtained from experienced schedulers to link the situations of disruptions to the choice of algorithms, while the second part is using seven components to modularize an algorithm and constructing a generic procedure to control the flow of algorithms. With the algorithms represented by the method, the decision support system for real-time vehicle routing can handle different disruptions occurring under different distribution states in real time. Moreover, the method can facilitate the process of real-time modeling and the maintenance of algorithms in a Decision Support System for real-time vehicle routing.*

Keywords: Knowledge representation, Algorithm, Decision support system, Real-time vehicle routing

1. **Introduction.** Real-time vehicle routing is important in supporting supply chain execution systems, and in minimizing the related logistics risks [1], as unexpected events, such as demand changes, time window changes and vehicle breakdowns, constantly occur during the process of urban distribution. Giaglis et al. [1] have demonstrated that a good, near-optimal, distribution plan is necessary but not sufficient for high performance distribution. This needs to be complemented by the capability of making and implementing sophisticated decisions in real time in order to respond effectively to unexpected events.

As unexpected events are diversified and distribution states change constantly with the plan-executing process, different optimization models and algorithms will be used by the decision process of real-time vehicle routing. For example, for the breakdown of vehicle v_1 occurring under the distribution state shown by Figure 1(a), the policy of inserting remaining tasks of 9, 13, 11, 4, 7, and 2 to the other two routes is applicable, which will use Relocate algorithm to optimize the insertion process. For the breakdown occurring under the distribution state shown by Figure 1(b), the policy of dispatching the unused vehicle v_2 to finish task 2, is applicable, which will use the Dijkstra's algorithm to find out a shortest path between the depot and the task 2. For the breakdown of vehicle v_2 and the disabled road 6—16 occurring under the distribution state shown by Figure 1(c), the policy of inserting remaining tasks of 17, 10 to the other two routes and the policy of searching for another feasible road instead of the disabled one are applicable respectively. In the former policy, Relocate algorithm will be used, while in the latter policy, the intra-route algorithm, such as 2-opt Exchange or Or-Opt will be used. We name the unexpected events which will disrupt the current distribution plan disruptions.

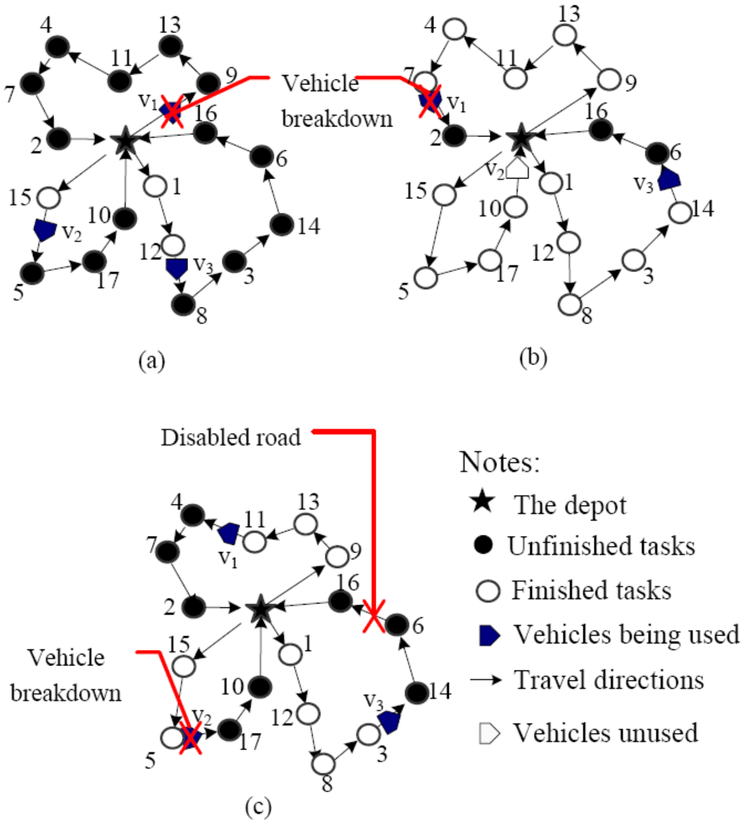

FIGURE 1. Unexpected events occurring under different distribution states

The above examples indicate that sometimes different algorithms are needed for solving the same kind of disruption occurring under different distribution states, and sometimes the same algorithm is needed for solving different kinds of disruptions.

Therefore, linking the situation of disruption to the choice of algorithms is a key problem for real-time rerouting. Hence, one of the objectives of knowledge representation is to support the process of identifying the situation of disruption and choosing an appropriate algorithm.

Furthermore, in practice, when a disruption occurs, the original problems' constraints would be violated and feasible solutions could not be obtained, in which case constraints relaxing and objective conceding are inevitable, in order to obtain an operable solution. Usually, a distribution plan with on-time deliveries is prior to the one with time-violation deliveries, while the one with time-violation deliveries is prior to the one with none deliveries. How to relax constraints and concede other objectives is the key issue in the process of modeling, in which an algorithm plays an important role. An algorithm could be utilized from two aspects in this process. It could be firstly used to check if a feasible solution satisfying current constraints could be obtained by a test run. If it could not, relaxed constraints would be checked. The process iterated until the final objective and constraints were identified. Secondly, with the final running, it could find out the near optimal solution based on the identified objective and constraints. How these two aspects could be realized in a decision-making process depends on how algorithms are represented in a Decision Support System (DSS).

Therefore, the second objective of knowledge representation method for algorithms in the DSS for real-time vehicle routing is to make them be efficiently and effectively utilized by the decision process.

This paper presents a knowledge representation method for algorithms in DSS for real-time vehicle routing in urban distribution, which achieves the two objectives mentioned above. The remaining part of the paper is organized as follows. Related work is reviewed in Section 2. The problem that the research concerns is stated, and the preconditions of the research are defined in Section 3. The knowledge representation method for algorithms is proposed in Section 4. In Section 5, computational experiment is done and the advantages of the proposed method are discussed. Finally in Section 6, conclusions are drawn and future work is introduced.

2. Related Work.

In most DSS for optimization problems, algorithms are represented in procedural scheme. Closely related with the specific problem to be solved, the procedural representation is efficient when an algorithm runs. However, as an algorithm represented by the procedural scheme is problem-specific, once the disruption to be handled is changed, its code has to be changed, which is inflexible for the reuse of the algorithm by different disruptions. Besides the procedural representation method, the similar research branch for modeling and solving optimization problems is the algebraic modeling language, which was pioneered by GAMS [2], AMPL [3], AIMMS [4,5], and followed by some complementary methods, like AMPL extension language [6] and MILANO [7]. These representation methods separated the part of the data from a model, which made parameters be input flexibly so that the model could be used by different problems of the same kind. However, efficient methods for solving real-time vehicle routing problems are local search heuristics, which embody some parts of constraints and objectives of algebraic models implicitly in heuristic knowledge, so the algebraic modeling method cannot be used directly by this kind of problem. A different knowledge representation method for this kind of algorithms is needed.

Existing literature on Real-time Vehicle Routing Problems (RVRP) focuses on the following aspects: algorithms, models, strategies and DSS. With respect to new customer requests, Ichoua et al. [8] proposed a vehicle diversion strategy and employed Tabu search algorithm to solve it. Yang et al. [9] proposed a mixed-integer programming formulation for the offline version of the problem and then considered and compared five rolling horizon strategies for the real-time version. Fang et al. developed an Isochrone-based decision method for determining the vehicle on road that can immediately respond to the new request [10]. With respect to vehicle breakdowns, Li et al. [11,12] developed a Lagrangian relaxation based-heuristic to get the strategy for rescheduling one or more vehicles to serve that trip and other service trips originally scheduled for the disabled vehicle. Besides the literature purely focusing on algorithms and models, there are some researches involving the design of DSS for real-time vehicle routing or rerouting, such as the systems of Li et al. [13], Zeimpekis and Giaglis et al. [14-16], Giaglis et al. [1], Fleischmann et al. [17], Du et al. [18] and Hu et al. [19]. In these DSSs, the core modules are also algorithms and models. In summary, these researches are gradually improving the optimization techniques in the area of RVRP. However, few of them can handle various disruptions occurring under different distribution states, as they usually use a specific model and algorithm to solve a specific kind of problem under a predefined distribution state. In order to equip the DSSs with the capability of handling different kinds of disruptions, organizing different kinds of algorithms in an efficient way to facilitate the real-time modeling process is the key issue.

3. Problem Statement and Preconditions.

As vehicle routing problems have many variants, in the paper, we confine the problem to the following conditions. There is one distribution center, which owns enough homogenous goods and several homogenous vehicles. Dispatched vehicles are redundantly loaded before starting their tasks for dealing with unexpected demands. The total demands of the customers in one route are less than or equal to the capacity of a vehicle. The service for customer i should start within a time window $[e_i, l_i]$ requested by the customer. The discussed problem is when a scheduled plan is being executed, an unexpected event that disrupts the current plan occurs, in which case, the decision process for handling the disruption by real-time rerouting is needed. Before presenting the knowledge representation method, some preconditions are necessary.

Precondition 1. Only local search algorithms will be used by the decision process for handling disruptions.

In this research, only local search algorithms will be considered. Larsen et al. argued that in dynamic settings, waiting for a long time in order to get a high quality solution is not possible, because the dispatcher wishes to know the solution to the current problem as soon as possible (preferably within minutes or seconds) [20]. The running-time constraint implies that rerouting and reassignments are often done by using local improvement heuristics like insertion and k-interchange [20]. Giaglis et al. also had the similar viewpoint that local plan adjustment may provide more cost-effective solutions without unnecessarily disturbing the overall initial plan [1]. Although the outcome of local search algorithms depends on initial solutions, it is not a limitation here, as the initial solution is the current distribution plan with high quality produced by a metaheuristic algorithm, e.g., Tabu search, genetic algorithms, evolution strategies [21]. More specifically, local search algorithms can be divided into two categories [22]: intra-route algorithms, which exchange or insert nodes or links within a route; and inter-route algorithms, which in contrast do that between routes. Bräysy and Gendreau's survey [23] also grouped most popular local search algorithms by the two categories. Table 1 summarizes the categories of local search algorithms mentioned by Bräysy and Gendreau [23].

Precondition 2. The real-time distribution state is represented by the method of three-layer tree structure of <Object—State—Set>.

The kind of the disruption and the distribution state when the disruption occurs determine which algorithm should be used by the policy for handling the disruption. Furthermore, the distribution state provides the algorithm with most of the input parameters. Hence, it is necessary to obtain the distribution state in real time before handling the disruption. The <Object—State—Set> tree structure shown in Figure 2 can be used to obtain the distribution state in real time.

4. **Knowledge Representation Method for Algorithms.**

4.1. **Knowledge representation for selecting algorithms.**

Policies obtained from experienced schedulers for handling different disruptions should be specifically defined to link the situations of disruptions to the choice of algorithms, which is the first part of the knowledge representation for algorithms.

Table 1. Categories of local search algorithms mentioned by Bräysy and Gendreau [23]

Algorithm Category	Algorithms
Inter-route improvement	2-opt*, Relocate, Exchange, CROSS-Exchange, GENI-Exchange, Cyclic Transfer
Intra-route improvement	2-opt Exchange, Or-Opt

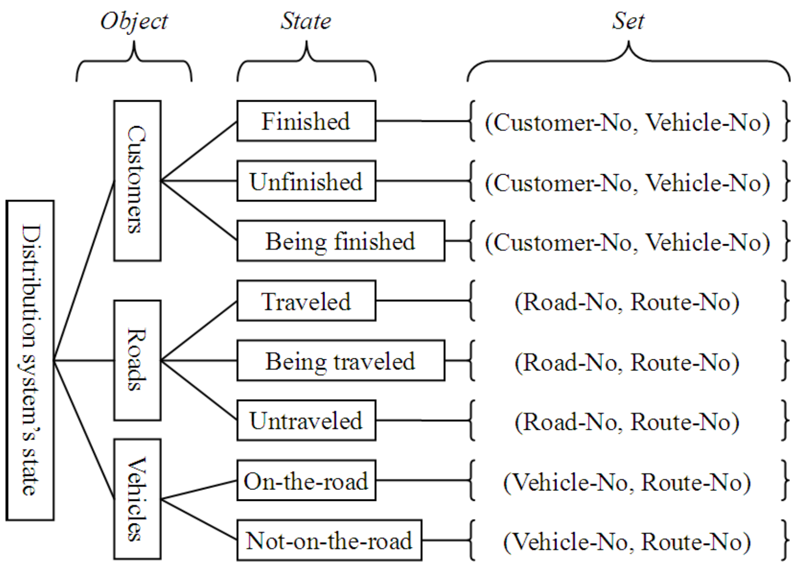

Figure 2. Representation of the real-time distribution state

Schedulers' policies are categorized by five kinds of disruptions occurring frequently during the distribution process. They are demand increasing (DI), order cancelling (OC), road unusability (RU), time violation (TV), and vehicle disability (VD). P-DI, P-OC, P-RU, P-TV, and P-VD are the corresponding policy sets for handling the five kinds of disruptions. Each policy set contains several specific rules used to link the specific situations of disruptions to the choice of algorithms. For example, parts of specific rules in P-DI policy set are as follows:

IF the initial task of the current customer has been finished, AND there is a single vehicle's redundant load could satisfy the increased demand, THEN insert this demand into one of the remaining routes.

IF the initial task of the current customer has been finished, AND there is not any vehicle's redundant load could satisfy the increased demand, THEN split this demand and insert those split parts to the remaining routes one by one.

...

These rules are represented by Horn clauses in PROLOG (Programming In Logic) language as follows.

```
p_DI(Operation,C_no,Quantity,Current_vehicle):-
demand_increasing_disruption(C_no,Increased_demand),
customer(C_no,_,Increased_demand, Current_vehicle,1),
vehicle(Route_no,_,Redundant_load,1),
Route_no\=Current_vehicle,
Redundant_Load>=Increased_demand,
Quantity=Increased_demand,
Operation="Relocate".
p_DI(Operation,C_no,Quantity,Current_vehicle):-
demand_increasing_disruption(C_no,Increased_demand),
customer(C_no, _,Increased_demand,Current_vehicle,1),
Quantity=Increased_demand,
```

Operation="Relocate-split".
...

In a rule, the premises define the situations, and the conclusion provides the operation which may include an algorithm and the other parameters to be transferred to the algorithm. After a rule is matched successfully, further optimization may needed, which will be realized by the algorithm.

4.2. **Knowledge representation for using algorithms in real-time modeling.** Local search algorithms have the common process for generating a new distribution plan when they are used by policies for handling disruptions, which is shown in Figure 3. Figure 3 has two parts. The dashed lines in the middle of the figure relate the two parts. The left part indicates the common process, in which each rectangle and the rhombus stand for a step. The right part lists the core component that is needed by each step to fulfill its task.

Step 1 defines the initial solution, which is the start point of a local search algorithm. The component INITIALIZATION is used to obtain the necessary parameters. It is the basic input of an algorithm. The component is necessary for separating the data from the flow of an algorithm.

Step 2 generates the neighborhood solutions. This task needs an OPERATOR to define how a neighborhood solution is created. Different algorithms have different operators. For example, swap $(r_i - c_i, r_j - c_j)$ and insert $(r_i - c_i, r_j - c_{(j-1)} - c_j)$ are two different operators that respectively belong to swap algorithm and insertion algorithm. For local search algorithms, all operations can be summarized into two categories, deleting roads, i.e., links, from the current routes, or adding roads to the current routes. Therefore, we

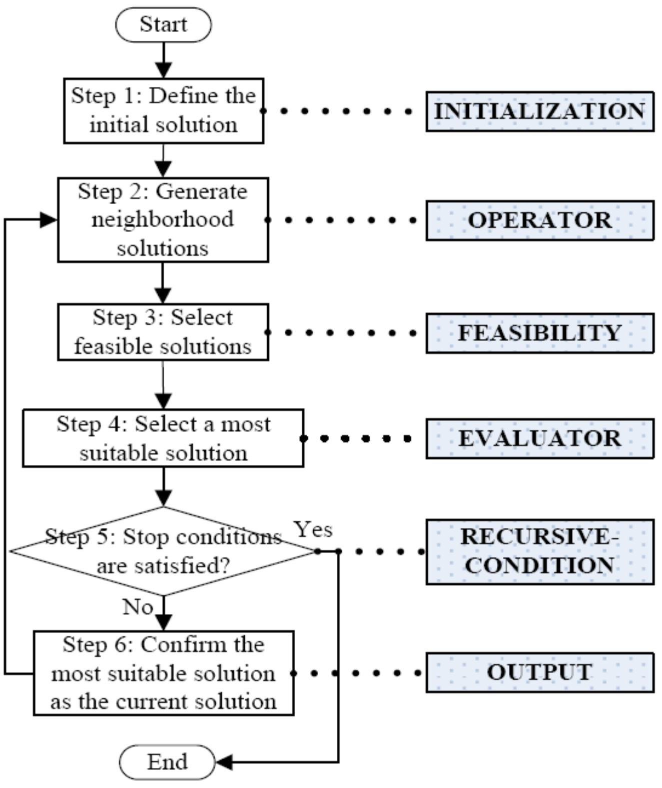
Start
Step 1: Define the initial solution
INITIALIZATION
Step 2: Generate neighborhood solutions
OPERATOR
Step 3: Select feasible solutions
FEASIBILITY
Step 4: Select a most suitable solution
EVALUATOR
Step 5: Stop conditions are satisfied?
Yes
No
RECURSIVE-CONDITION
Step 6: Confirm the most suitable solution as the current solution
OUTPUT
End

Figure 3. The common process of local search algorithms and the core components

use two facets, "Deleted-roads" and "Added-roads", which are the sets that aggregate the deleted roads and the added roads respectively, to describe this component.

Step 3 selects the feasible solutions from the neighborhood solutions. The fulfillment of the task depends on the component of FEASIBILITY, which is used to check if the constraints can be satisfied by a neighborhood solution. The content of the component is determined by the constraints of the problem.

Step 4 selects a most suitable solution from the feasible solutions as the next current solution for generating next neighborhood solutions. The component of EVALUATOR is used to compare the feasible solutions with the current solution and find out the better one. The content of the component is determined by the optimization objective of the problem.

Step 5 checks whether the recursive condition has been reached or not, as algorithms will recursively run until some condition is reached. The component of RECURSIVE-CONDITION defines the condition. Its data type is Boolean, in which "true" means continuing the recursion, while "false" means stopping the recursion.

Step 6 confirms the finally selected solution as the current solution if the recursive condition is true. This task will be fulfilled by the component of OUTPUT.

Besides the six components mentioned above, the identifier of an algorithm is necessary. We use NAME as an algorithm's identifier. It is used to link the conclusion of a rule with the corresponding algorithm, and to locate the other six components of the algorithm.

The above analysis shows that a local search algorithm can be made from the seven components, NAME, INITIALIZATION, OPERATOR, FEASIBILITY, EVALUATOR, RECURSIVE-CONDITION, and OUTPUT. Besides, a generic procedure is needed for controlling the flow of algorithms. As frames provide a convenient way to combine declarations and procedures within a knowledge representation scheme, it is suitable for representing algorithms. Hence we created a frame-based representation method. As

Table 2. Modularization of relocate algorithm

```
NAME
  <Relocate>
INITIALIZATION
  Customer-to-be-relocated: @x
  Quantity-to-be-relocated: @q
  Untraveled-roads: {[@start-node,@end-node,@vehicle-id]}
OPERATOR
  For [@i,@j,@k]∈Untraveled-roads
  Deleted-roads: {[@i,@j,@k]}
  Added-roads: {[@i,@x,@k], [@x,@j,@k]}
  Flag [@i,@j,@k]
  /* Flag marks those elements that have been examined in the set of untraveled-roads.*/
RECURSIVE-CONDITION: Boolean
  If ∃[@i,@j,@k]∈Untraveled-roads hasn't been examined, True;
  Otherwise, False
FEASIBILIT: Boolean
  Check @C
EVALUATOR: real
  Calculate @O
OUTPUT(@Untraveled-roads)
  Untraveled-roads deduct Deleted-roads plus Added-roads
/*Variables C and O stand for the constraints and objectives respectively. */
```

analyzed above, the representation should contain two parts. One part is the seven components used to modularize an algorithm. The other part is a generic procedure used to control the flow of algorithms. The generic procedure will be instantiated by the seven components of an algorithm when the algorithm is confirmed by a policy.

Table 2 takes the Relocate algorithm as an example to show the modularization of an algorithm, in which the sentences between /* and */ are explanations, and @ is prefixed to variables.

5. **Advantages of the Representation.**

The knowledge representation method has three advantages from the perspective of supporting the decision process of handling disruptions in real time.

1) Algorithms represented by it can be flexibly used by the decision process to efficiently handle different disruptions under different distribution states.

2) It facilitates real-time modeling and problem-solving process.

3) It facilitates the maintenance of algorithms.

In order to prove these advantages, computational experiments and results analysis have been done in the following sections.

5.1. **Computational experiment.**

Two sets of disruption problems were generated randomly based on Professor Cordeau's VRP benchmark problems. One set is DI disruptions, and the other set is OC disruptions. The benchmark problems for Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) are used, which include 56 problem description files (http://neo.lcc.uma.es/radi-aeb/WebVRP/dat a/instances/cordeau/C-vrptw.zip) and 56 solution files (http://neo.lcc.uma.es/radi-aeb/WebVRP/data/instances /cordeau/C-vrptw-sol.zip). These problems are grouped by C type, R type and RC type. Problem sets C have the clustered customers. Sets R have the customers whose locations were generated uniformly and randomly over a square. Sets RC have a combination of randomly placed and clustered customers. We selected C101 and C104 from C type, R104 and R107 from R type, RC104 and RC108 from RC type. The reason for selecting these problems is that their solutions contain the same amount of routes, 10, which needs 10 vehicles to serve. As in practice a distribution center or corporation usually owns a limited number of vehicles, we assume that this number is 10. Before a disruption occurs, the distribution plan will be executed following the 10 routes. As disruptions occur randomly, in order to ensure an environment close to the real world, the time when and the position where the disruption will occur will also be generated randomly. Table 3 shows some part of the generated data.

The experiment was conducted in a computer with the 32 bits Windows 7 operating system running on the following hardware: CPU – Intel Dual Core with 2.8 GHz, 2.00 GB of RAM.

5.2. **Computational results.**

As algorithms are modularized by the knowledge representation method, the components of INITIALIZATION, OPERATOR and RECURSIVE-CONDITION could be used by a test run for checking if a feasible solution satisfying current constraints could be obtained. If it could not, relaxed constraints would be checked. Figure 4 describes a part of this constraint-checking process, in which Otd stands for the objective of "minimizing travel distance"; Otv stands for the objective of "minimizing time violation"; Cc stands for the constraint of "capacity" and Ctw stands for the constraint of "time windows". The output of the test run is the identified constraints and objectives. Hence, when no solutions can be obtained by complying with initial constraints, constraints will be relaxed and another objective may also be conceded by the test run of the algorithm with the proposed knowledge representation, while the traditional method

TABLE 3. Some part of the generated data

Problem Type	DI				OC		
	Route No.	Position	Occurring Time	Increased Demand	Route No.	Position	Occurring Time
C101	9	75	105	24	4	15	278
C101	8	93	116	12	6	89	681
C101	7	37	520	70	10	70	281
...	...	...	...	...	...	...	...
C104	10	75	753	34	3	15	278
C104	9	93	116	12	1	89	681
C104	6	37	520	70	8	70	281
...	...	...	...	...	...	...	...
R104	9	75	78	48	9	75	31
R104	10	93	31	67	10	93	34
R104	10	37	27	78	6	83	1
...	...	...	...	...	...	...	...
R107	1	75	200	48	1	43	21
R107	6	93	178	112	4	51	26
R107	8	37	121	70	6	84	71
...	...	...	...	...	...	...	...
RC104	8	75	92	114	8	75	96
RC104	6	93	199	83	9	89	16
RC104	7	37	163	70	7	43	48
...	...	...	...	...	...	...	...
RC108	2	75	118	90	2	75	136
RC108	5	93	216	83	3	48	74
RC108	7	37	3	70	3	89	1

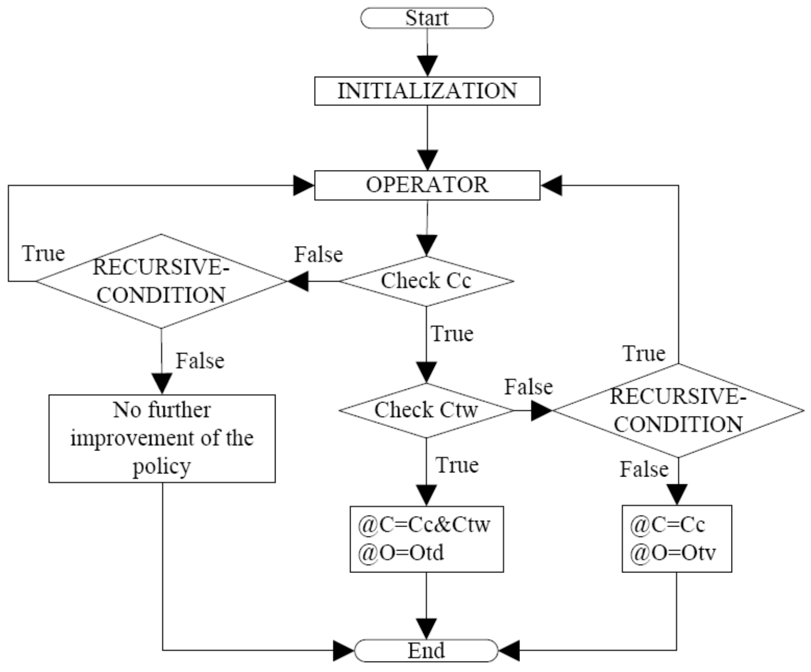

FIGURE 4. A part of the test run of an algorithm in the modeling process

used to solve disruptions never allows constraints to be relaxed. Computational results were obtained by the two methods respectively.

Table 4 compared the results of the proposed method with those of the traditional method from the following 4 criteria when they are used to handle DI disruptions. The results of the two different methods are aggregated respectively by the six problems in this table.

Table 4. The comparison between the aggregation results of 100 DI cases obtained respectively by the proposed method and traditional method

Problem type	Total increased demands	Method	① Average cost	② Average amount of the affected customers	③ Total rejected demands	④ Rejected rate	CPU(S) MIN/ MAX/AVG
C101	5567	Proposed	63.24	0.9	119	2.14%	0.0090/0.0316/ 0.0188
		Traditional	44.69	0	4825	86.67%	
C104	5443	Proposed	73.81	1.92	109	2.00%	0.0095/0.0289/ 0.0197
		Traditional	28.42	0	5087	93.46%	
R104	6070	Proposed	34.33	0.43	0	0.00%	0.0159/0.0289/ 0.0186
		Traditional	27.40	0	2813	46.34%	
R107	6830	Proposed	35.06	0.26	105	1.54%	
		Traditional	29.47	0	1802	26.38%	0.0092/0.0287/ 0.0195
RC104	5233	Proposed	50.72	0.53	47	0.90%	
		Traditional	33.56	0	2867	54.79%	0.0095/0.0434/ 0.0185
RC108	5887	Proposed	61.17	0.62	100	1.70%	
		Traditional	32.49	0	3688	62.65%	0.0101/0.0378/ 0.0197

① "Average cost": It means that for the 100 DI disruptions in a problem, the average increased travel distance of the 100 solutions achieved respectively by the two methods, compared with the original solution.

② "Average amount of the affected customers": In a solution to a DI disruption, in order to satisfy all demands, some customers' time windows have to be violated. This kind of customers is defined as the affected customers. The criterion compares the average amount of the affected customers of the 100 solutions achieved respectively by the two methods.

③ "Total rejected demands": Limited by the capacity of a vehicle and the amount of vehicles, some of the increased demand cannot be satisfied in the current distribution period, which has to be rejected. This criterion compares the total rejected demands of the 100 solutions achieved respectively by the two methods.

④ "Rejected rate" = total rejected demands/total increased demands $\times$ 100%.

Besides, the "CPU seconds" (CPU(S)) is used to measure the running time of CPU for achieving a solution by the proposed method. Among the 100 CPU seconds for the 100 solutions, MIN records the shortest one, and MAX records the longest one, while AVE calculates the average value of them.

Table 5 compared the results of the proposed method with those of the traditional method from the two criteria, Average cost and Average amount of the affected customers,

Table 5. The comparison between the aggregation results of 100 OC cases obtained respectively by the proposed method and traditional method

Problem type	Method	① Average cost	② Average amount of the affected customers	CPU(S)MIN/MAX/AVG
C101	Proposed	3.98	2.51	0.0152/0.0912/0.0445
	Traditional	0.00	6.03	
C104	Proposed	10.06	1.26	0.0153/0.1246/0.0448
	Traditional	0.00	1.97	
R104	Proposed	9.34	1.73	0.0154/0.0952/0.0295
	Traditional	0.00	2.55	
R107	Proposed	11.26	1.47	0.0152/0.0994/0.0333
	Traditional	0.00	1.97	
RC104	Proposed	14.31	1.35	0.0152/0.1287/0.0379
	Traditional	0.00	1.8	
RC108	Proposed	7.49	1.13	0.0152/0.0949/0.0386
	Traditional	0.00	1.36	

when they are used to handle OC disruptions. As there are no solutions to OC disruptions by rerouting if no constraints are allowed to be violated, the best way to handle this kind of disruption in tradition is to keep the original solutions, which results in no cost. "CPU seconds" is also recorded for the proposed method.

5.3. **Discussions.** Table 4 indicated that when handling DI disruptions, the proposed method used little cost to handle almost all increased demands, while the traditional method rejected almost all demands. For example, for C101, the total increased demands are 5567. The proposed method cost 63.24 and violated 0.9 customer's time window in average, while just rejected the amount of demands, 119, with the rejected rate of 2.14%. However, although the traditional method cost 44.69, less than that of the proposed method, and violated 0 customer's time window in average, it rejected the amount of demands, 4825, with the rejected rate of 86.67%, which is very high and will lead to great customer dissatisfaction in the long run. Table 5 indicated that when handling OC disruptions, the proposed method just used little cost to greatly induce the amount of affected customers in average, while the traditional method affected more customers. For example, for C101, the proposed method just cost 3.98 in average and reduced the amount of affected customers from 6.03 to 2.51 compared with the traditional method. The results of the proposed method are preferable in practice to those of the traditional one, because the latter affects more customers, which leads to more customer dissatisfaction and thus does harm to the distribution corporation in the long run.

The above results indicate that the algorithms represented by the proposed method can efficiently handle different disruptions under different distribution states. As the objectives and constraints could be flexibly relaxed by the representation method, the solutions obtained by the proposed method are more practical and operable than those obtained by the traditional method.

The CPU(S) in Tables 4 and 5 shows that the minimum running time for handling DI disruptions is 0.0090, the maximum one is 0.0434, while the minimum running time for handling OC disruptions is 0.0152 and the maximum one is 0.1287. This indicates that the modeling and problem-solving process can be achieved in real time. Hence the second advantage of the proposed method has been proved.

Besides the above mentioned advantages, the maintenance of local search algorithms in DSS for RVRP becomes simple with the algorithm representation method. For example, when some customers in a disabled route need to be inserted to the other routes, an algorithm should firstly delete a customer from its current route and then insert it to another route. In this case, the algorithm will be achieved by inheriting the Relocate algorithm based on the frame-based representation. Table 6 shows the new algorithm.

Table 6. The delete-insertion algorithm

```
NAME
   <Delete-Insertion>
   ISA<Relocate>
OPERATOR
   Deleted-roads: $\cup\{[@a_x,@x,@r_x],[@x,@p_x,@r_x]\}$
   Added-roads: $\cup[@a_x,@p_x,@r_x]$
/*$@a_x$ stands for the anterior customer of customer $x$, and $@p_x$ stands for the posterior
customer of it in a route.*/
```

By the inheriting feature of the frame scheme, all components in the Relocate algorithm frame are duplicated automatically by the function of "***ISA***<Relocate>". The only changed thing is the additional elements in the sets of Deleted-roads and Added-roads, which is realized by the "∪" operation. The generic procedure is applicable to the new algorithm without any changes.

The example obviously demonstrates that the representation method compresses the code of algorithms, which means that some algorithms can be realized by inheriting existing relative algorithms. The code of son algorithms only records the different parts from their ancestors. Moreover, the separation of the control flow and algorithms' components facilitates the maintenance process, too. When altering an algorithm, people do not need to care its flow, but focus on the components of it. The characteristics of inheritance and separation greatly decrease the labor for the maintenance of algorithms.

6. Conclusions and Future Work.

The research proposes a knowledge representation method for the algorithms suitable for real-time vehicle routing, which firstly uses rules to represent policies obtained from experienced schedulers to link the situations of disruptions to the choice of algorithms, and then uses seven components to modularize an algorithm and a generic procedure to realize the control flow of algorithms. The knowledge representation method has the following three advantages.

1) Algorithms represented by the knowledge representation method can be used to handle different disruptions occurring under different distribution states. The objectives and constraints could be flexibly relaxed by the representation method, and thus more practical results could be achieved, which overcomes the deficiency of traditional method that just uses a fixed model and algorithm to solve a specific kind of problem under a predefined distribution state and thus usually cannot obtain operable solutions in practice.

2) It facilitates the real-time modeling process for supporting the decision process of real-time vehicle routing. The computational results show that the running time of the problem-solving process is short enough for real-time modeling. This advantage equips the DSS for RVRP with the capability of automated or semi-automated modeling in real time.

3) Moreover, it facilitates the update and maintenance of algorithms. First of all, some algorithms can be achieved by inheriting existing relative algorithms. This compresses the code of algorithms, as the code of son algorithms only needs to record the different parts from their ancestors. Secondly, as the control flow and the components of an algorithm are separated, when people modify an algorithm, they do not need to change the whole algorithm's structure or care the algorithm's flow, but only to modify the corresponding components of it. This advantage is particularly important for the DSS for real-time vehicle routing in urban distribution, in which many algorithms need to be managed.

In order to efficiently and effectively utilize the algorithms represented by the proposed method, future work includes two aspects. As knowledge is different for different kinds of disruptions, the development of knowledgebase is a time-consuming task. In this research, only the knowledge for algorithms used to handle two kinds of disruptions, DI and OC, has been gathered and represented. Hence one aspect of the future work is to try to perfect the modeling system for handling all kinds of disruptions. The other one is the design of the man-machine interaction system for the acquirement and maintenance of algorithms.

Acknowledgment. This work has been partially supported by the grants from the National Natural Science Funds for Distinguished Young Scholar (No. 70725004), Natural Science Foundation of China (Nos. 70890080, 70890083), and the Fundamental Research Funds for the Central Universities (No. DUT11RC(3)41). The authors also gratefully acknowledge the helpful comments and suggestions from the reviewers and AE, which have great contributions to the improvement of the presentation and the quality of the paper.

REFERENCES

[1] G. M. Giaglis, I. Minis, A. Tatarakis and V. Zeimpekis, Minimizing logistics risk through real-time vehicle routing and mobile technologies: Research to date and future trends, *International Journal of Physical Distribution & Logistics Management*, vol.34, no.9, pp.749-764, 2004.

[2] A. Brooke, D. Kendrick and A. Meeraus, *GAMS-A User's Guide: Release 2.25*, The Scientific Press, San Francisco, CA, 1992.

[3] R. Fourer, D. M. Gay and B. W. Kernighan, A modeling language for mathematical programming, *Management Science*, vol.36, no.5, pp.519-554, 1990.

[4] J. Bisschop and R. Entriken, *AIMMS – The Modeling System*, Paragon Decision Technology, 1993.

[5] J. Bisschop, *AIMMS Optimization Modeling*, lulu.com, US, 2006.

[6] M. Desrochers, C. V. Hones, J. K. Lenstra, M. W. P. Savelsbergh and L. Stougie, Towards a model and algorithm management system for vehicle routing and scheduling problems, *Decision Support Systems*, vol.25, no.2, pp.109-133, 1999.

[7] P. Sengupta, *MILANO, An Object-Oriented Algebraic Modeling System*, http://www.ips.invensys.com/en/products/ots/Documents/Milano.pdf, Simulation Sciences Inc., 2009.

[8] S. Ichoua, M. Gendreau and J. Y. Potvin, Diversion issues in real-time vehicle dispatching, *Transportation Science*, vol.34, no.4, pp.426-438, 2000.

[9] J. Yang, P. Jaillet and H. Mahmassani, Real-time multivehicle truckload pickup and delivery problems, *Transportation Science*, vol.38, no.2, pp.135-148, 2004.

[10] Y. Fang, X. Hu, L. Sun and Q. Ding, An isochrone-based decision method for real-time vehicle rerouting problem with immediate requests in urban distribution, *ICIC Express Letters*, vol.3, no.4(B), pp.1299-1304, 2009.

[11] J.-Q. Li, P. B. Mirchandani and D. Borenstein, Real-time vehicle rerouting problems with time windows, *European Journal of Operational Research*, vol.194, no.3, pp.711-727, 2009.

[12] J.-Q. Li, P. B. Mirchandani and D. Borenstein, A Lagrangian heuristic for the real-time vehicle rescheduling problem, *Transportation Research Part E*, vol.45, no.3, pp.419-433, 2009.

[13] J.-Q. Li, D. Borenstein and P. B. Mirchandani, A decision support system for the single-depot vehicle rescheduling problem, *Computers & Operations Research*, vol.34, no.4, pp.1008-1032, 2007.

[14] V. Zeimpekis, G. M. Giaglis and I. Minis, A dynamic real-time fleet management system for incident handling in city logistics, *Vehicular Technology Conference*, vol.5, pp.2900-2904, 2005.

[15] V. Zeimpekis and G. M. Giaglis, A dynamic real-time vehicle routing system for distribution operations, *Consumer Driven Electronic Transformation*, 2005.

[16] V. Zeimpekis and G. M. Giaglis, Urban dynamic real-time distribution services insights from SMEs, *Journal of Enterprise Information Management*, vol.19, no.4, pp.367-388, 2006.

[17] B. Fleischmann, S. Gnutzmann and E. Sandvoß, Dynamic vehicle routing based on online traffic information, *Transportation Science*, vol.38, no.4, pp.420-433, 2004.

[18] T. Du, F. K. Wang and P.-Y. Lu, A real-time vehicle-dispatching system for consolidating milk runs, *Transportation Research Part E: Logistics and Transportation Review*, vol.43, no.5, pp.565-577, 2007.

[19] X. Hu, M. Huang and A. Z. Zeng, An intelligent solution system for a vehicle routing problem in urban distribution, *International Journal of Innovative Computing, Information and Control*, vol.3, no.1, pp.189-198, 2007.

[20] A. Larsen, O. B. G. Madsen and M. M. Solomon, Classification of dynamic vehicle routing systems, in *Dynamic Fleet Management: Concepts, Systems, Algorithms & Case Studies*, V. Zeimpekis, C. D. Tarantilis, G. M. Giaglis and I. Minis (eds.), Springer, 2007.

[21] O. Bräysy and M. Gendreau, Vehicle routing problem with time windows, Part II: Metaheuristics, *Transportation Science*, vol.39, no.1, pp.119-139, 2005.

[22] T. C. Du, E. Y. Li and D. Chou, Dynamic vehicle routing for online B2C delivery, *Omega*, vol.33, no.1, pp.33-45, 2005.

[23] O. Bräysy and M. Gendreau, Vehicle routing problem with time windows, Part I: Route construction and local search algorithms, *Transportation Science*, vol.39, no.1, pp.104-118, 2005.