B A D A N I A O P E R A C Y J N E I D E C Y Z J E

Nr 4 2003

Przemysław KOBYLAŃSKI*
Michał KULEJ*

IMPROVED SOLUTIONS FOR VEHICLE ROUTING AND SCHEDULING WITH FUZZY TIME WINDOWS AND FUZZY GOAL

In this paper, we consider a vehicle routing and scheduling problem with fuzzy time windows and a fuzzy goal. A two-stage method for obtaining the improved optimal solution to the problem under consideration is presented. This method uses the constraint programming as an effective tool for solving the problem.

1. Introduction

This paper deals with the following problem: For given set of customers some goods are delivered by vehicles from a point further called depot. The supply of goods to every customer is made by a fleet of K vehicles. Every vehicle takes the goods from the depot, next delivers the goods to selected customers and returns back to the depot, which constitutes a route. The routes cannot violate the capacity of the vehicles and exactly one vehicles and exactly one vehicle can realize the supply of goods to the customer. Some preferences concerning the time of delivery for each customer are given. These preferences, in contrast to Vehicle Routing with Time Windows [2], [6], [7], are modeled by trapezoidal fuzzy numbers.

The goal which, in the classical problem, is to minimize the total working time of the vehicles is also represented by a trapezoidal fuzzy number. Now, the problem consists in assigning a route and finding the time schedule for each vehicle while the least of the degrees of satisfaction of the time windows and the goal constraint are maximal. A mixed integer linear programming model of the problem and heuristic methods of

* Institute of Industrial Engineering and Management, Wrocław University of Technology, ul. Smoluchowskiego 25, 50-372 Wrocław, przemyslaw.kobylanski@pwr.wroc.pl, michal.kulej@pwr.wroc.pl

solution are given in [5]. The optimal solution called a max–min optimal solution has a drawback: while determining the optimal degree of satisfaction it does not try to obtain as high as possible a degree of satisfaction of other time window constraints. This frequently causes that for the max-min optimal solution the degrees of the satisfaction of temporal and goal constraints are fixed at a low level. In this paper, using the constraint programming we present, a method leading to an improved solution. The degree of satisfaction of the improved solution is the same as that of the max–min optimal solution while the degree of satisfaction of all the constraints (for the time windows or the goal) is as high as possible. The solution obtained with the method presented is a discrimin-optimal solution [1].

2. Problem formulation

To define the problem under consideration the following data characterizing the customers and vehicles are introduced:

- n number of customers,
- $N = \{1, 2, \ldots, n\}$ the set of customers,
- $N_0 = \{0\} \cup N$ index 0 indicates the depot,
- s_i service time at point $i \in N$,
- q_i demand of customer $i \in N$,
- TW_i trapezoidal fuzzy number (a_i, e_i, l_i, b_i) characterizing the preferences of customer i about the service start time,
- K number of available vehicles,
- $M = \{1, 2, \ldots, K\}$ the set of vehicles,
- Q_k capacity of the vehicle $k \in M$,
- t_{ij} travel time from customer i to customer j.

Decision maker preferences concerning the goal (minimizing the total working time of the vehicle) are given by the fuzzy number:

- $\tilde{G}$ trapezoidal fuzzy number $(0, 0, g_1, g_2)$.

Next, we define the notions of the route of vehicle, the schedule of the route, the solution of the problem and formulate the criterion.

A *tour* is represented by list of indices of the customer who are served on this tour. The list starts and ends with number 0, which represents the depot. An example tour is given in Figure 1.

The schedule $t = (S = t_0, t_1, \ldots, t_m, t_{m+1} = F)$ of the route $(0, 1, 2, \ldots, m, 0)$ defines the sequence of service start times of successive customers, where $S = t_0$, $t_{m+1} = F$ indicate, respectively, the start time and finish time of the route. The numbers t_i fulfil the following constraints:

$$a_0 \le t_0 = S\,,$$
$$t_i + s_i + t_{ii+1} \le t_{i+1},\ i = 0, 1, ..., m\,,$$
$$F = t_{m+1} \le b_0\,.$$

We assume that the time window of the depot [a_0, b_0] is crisp. For the route (0, 1, 2, 0) three example schedules are given in Figure 2. The minimal degree of satisfaction of time windows of customers 1 and 2 for the first schedule (S_1, t_1, t_2, F_1) is equal

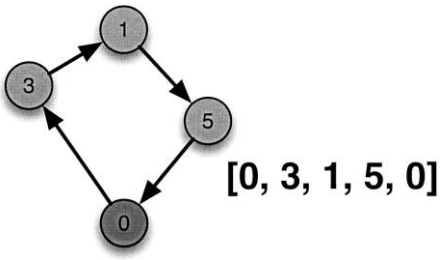


Fig. 1. The route of vehicle serving customers 3, 1, 5

to 0.25 = min {1, 0.25}, for the second schedule (S_1', t_1', t_2', F') it is equal to 0.5 = min {0.5, 0.5} and for the third one (S_1', t_1', t_2', F') it is equal to 0.5 = min {0.5, 0.5}. The last schedule (S_1', t_1', t_2', F') has the same degree of satisfaction of the time windows as the second one but has a shorter total realization time $F_1'' - S_1'' < F_1' - S_1'$.

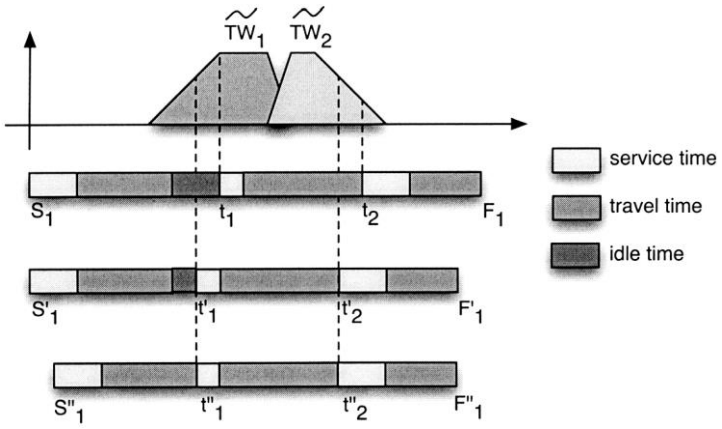


Fig. 2. Schedules of the route [0, 1, 2, 0]

Let S be the set of tours and $P(S)$ the set of customers which are served on tours from set S (see Fig. 3).

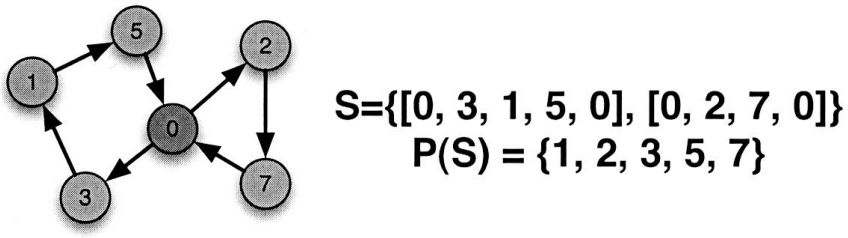


Fig. 3. The set of two tours

Now, we proceed to formulate the criterion. Let $\mathsf{T}(S)$ be the set of all possible schedules for tours from set S. For the schedule $t \in \mathsf{T}(S)$, t_i denotes the start time of serving customer i and TOTAL(t) is the total time of realization of all tours in S. We will denote by $\mathrm{SAT}_{\tilde{G}}(t)$ the maximal degree of satisfaction of all the temporal constraints, which depends on fuzzy time windows $\tilde{T}\tilde{W}_i$, for every customer $i \in P(S)$ and on the satisfaction of the goal $\tilde{G}$ by the value of TOTAL(t):

$$\mathrm{SAT}_{\tilde{G}}(S) = \sup_{t \in \mathsf{T}(S)} \min \{ \min_{i \in P(S)} \mu_{\tilde{T}\tilde{W}_i}(t_i),\ \mu_{\tilde{G}}(\mathrm{TOTAL}(t)) \}.$$

Now, we can state the problem as follows:

Find the set S and K tours and schedules such that $P(S) = N$ (every customer is served by exactly one vehicle) and degree of satisfaction $\mathrm{SAT}_{\tilde{G}}(S)$ is maximal.

The mixed integer programming model [5] is as follows. We assume further that $i, j \in N$, $k \in M$. Let us define the following binary decision variables;

- $x_{ijk} = 1$ iff customer j follows customer i in the sequence visited by vehicle k,
- $y_{ik} = 1$ iff customer i is visited by vehicle k,

and the real decision variables:

- t_i service start time of the customer $i \in N$,
- S_k start time of the route of vehicle k,
- F_k finish time of the route of vehicle k.

For $i \in N$ we define the following auxiliary real variables:

- x_0 total time of realization of all tours,
- y_0 such level α that α-cut of $\tilde{G}$ contains x_0,
- y_i such level α that α-cut of $\tilde{T}\tilde{W}_i$ contains t_i,
- y lower bound of all levels of $\tilde{T}\tilde{W}_i$ and $\tilde{G}$.

Constraints for the tours and the capacity of the vehicles:

$$\sum_{i \in N_0} x_{irk} - \sum_{j \in N_0} x_{rjk} = 0, \quad \text{for} \quad r \in N, k \in M,$$

$$\sum_{i\in N} x_{i0k} = \sum_{j\in N} x_{0jk} = 1, \quad \text{for} \quad k \in M,$$

$$\sum_{k\in M} y_{ik} = 1, \quad \text{for} \quad i \in N,$$

$$\sum_{j\in N_0} x_{ijk} = y_{ik}, \quad \text{for} \quad i \in N, k \in M,$$

$$\sum_{i\in N} q_i \cdot y_{ik} \le Q_k, \quad \text{for} \quad k \in M.$$

Constraints for the schedules:

$$\begin{aligned} S_k &\ge e_0, \\ F_k &\le l_0, \\ t_i + s_i + t_{ij} - (1 - x_{ijk}) \cdot T &\le t_j, \\ S_k + s_0 + t_{0j} - (1 - x_{0jk}) \cdot T &\le t_j, \\ t_i + s_i + t_{i0} - (1 - x_{i0k}) \cdot T &\le F_k, \\ a_i + (e_i - a_i) \cdot y_i &\le t_i, \\ b_i - (b_i - l_i) \cdot y_i &\le t_i, \end{aligned}$$

for $i, j \in N$ and $k \in M$.

$$\begin{aligned} \sum_{k\in M} (F_k - S_k) &= x_0, \\ x_0 &\ge 0, \\ g_2 - (g_2 - g_1) \cdot y_0 &\ge x_0. \end{aligned}$$

Constraints for the evaluation of the levels of satisfaction:

$$\begin{aligned} y &\le y_i, \quad \text{for} \quad i \in N, \\ y &\le y_0. \end{aligned}$$

The objective function is stated as follows:

$$y \to \max.$$

Maximal value y^* is equal to $\mathrm{SAT}_{\tilde{G}}(S^*)$, where S^* is the optimal set of tours with the schedules which satisfy the temporal and goal constraints to the highest degree. The above model can be solved by commercial packages for integer programming but only for small size problems. For practical problems heuristic algorithms have to be developed.

3. Constraint programming

Constraint programming is an easy way to express the problem considered in a declarative rather than algorithmic manner. It consists in imposing a set of constraints on the decision variables from the mathematical model of the problem. Programming systems (such as IF/PROLOG, CHIP, SICSTUS, PROLOG, ILOG SOLVER and many others) supply methods for searching solutions which satisfy the imposed set of constraints.

In computational examples the IF/PROLOG logic programming language is used. We present some fundamental issues of the constraint programming by an example of package `const_linear` [3], which is used to impose linear constraints on rational value variables.

3.1. Variables and domains

Names of variables start with capital letters. Each variable has a set of possible values. This set is called the domain of variable.

If the domain of variable X has more than one element, the predicate `is_constraint(X)` is true.

Rational number P/Q is written with prefix 0r as 0rP/Q, where P and Q are integer numbers from the range $\pm 10^{300\,000}$. It warrants the accuracy of solutions without the aggregation of rounding errors.

3.2. Linear terms, constraints and objective function

The syntax of a linear term is as follows:

```
linear term ::=   variable
                  rational constant
                  + linear term
                  - linear term
                  linear term + linear term
                  linear term - linear term
                  linear term * linear term
                  linear term / linear term
                  (linear term)
```

Multiplication of two variables is allowed in the linear term but it is delayed until one of them has been instantiated (division of two variables is forbidden).

The syntax of a linear constraint is as follows:

```
linear constraint ::= linear term $=  linear term
                      linear term $<= linear term
                      linear term $>= linear term
                      linear term $<  linear term
                      linear term $>  linear term
                      linear term $\= linear term
```

Let `L` be the lists of variables [X_1, X_2, …, X_n]. The following predicates are defined:

`all_positive(L)` variables have positive values,
`all_negative(L)` variables have negative values,
`all_different(L)` variables have different values.

Let `LT` be a linear lerm. To find the maximal value of `LT` the predicate `linear_maximize(LT, M)` is used, where `M` is the variable to which the value found is assigned. To find the minimal value `linear_minimize(LT, M)` is used.

3.3. Constraint programming vs. simplex method

Simplex method searches for the optimal solution proceeding from one basic solution to another. When the optimum is obtained we have only one optimal solution known to us. It is possible to enumerate all alternative optimal basic solutions through changing the basis.

In the constraint programming the optimal solution is obtained through symbolic operations: variable elimination (projection), constraint propagation and redundant constraints elimination. The optimal solution is not represented as a basic vector but as a set of constraints, which are fulfilled by the optimal solutions only.

Example 1. *Find the maximal value of* $x + y$ *subject to* $x + y \leq 5$, *where x and y are unrestricted real variables*.

The linear programming model for the above example in CPLEX format is as follows:

```
Maximize
 obj: x + y
Subject To
 c1: x + y <= 5
Bounds
       x Free
       y Free
End
```

The simplex method from CPLEX library finds the following solution:

```
CPLEX> display solution variables 1-2
Variable Name           Solution Value
x                             5.000000
All other variables in the range 1-2 are zero.
```

The solution $x = 5$, $y = 0$ is only one from a continuum set of optimal solutions.

To solve Example 1 with the constraint programming, the following goal is given to IF/PROLOG system:

```
:- import const_linear
[user] ?- X+Y $=< 5, linear_maximize (X+Y, Z).
```

The answer is as follows:

```
X       = __2
Y       = __3
Z       = 5
```

which says that the maximal value of the objective function is 5 but variables x and y have domains containing more than one value (there are more than one optimal solutions).

With the following goal in SICSTUS PROLOG:

```
| ?- use_module (library (clpr)).
| ?-clpr: {X+Y=:=5},  sup(X+Y, Z)
```

we get the answer:

```
Z = 5.0,
{Y=5.0-X}
yes
```

with the maximal value $z = 5$ and the equation $y = 5 - x$, which describes the set of all the optimal solutions in Example 1.

4. Improved solutions

Let us consider the following simple example:

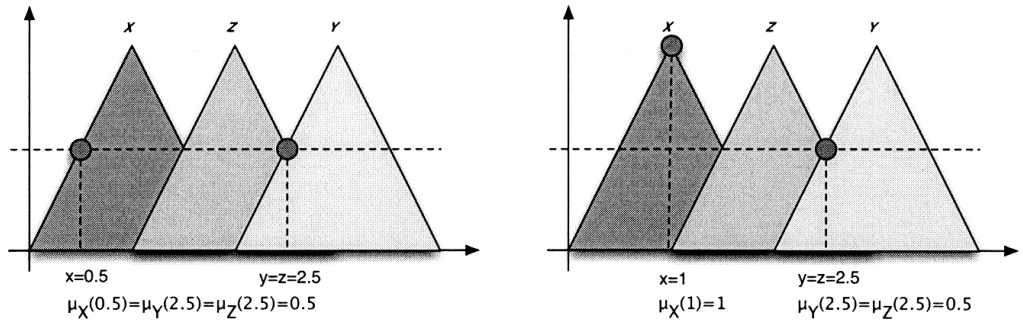
Example 2. *Three fuzzy triangular numbers* $\tilde{X}$ = (0, 1, 2), $\tilde{Y}$ = (2, 3, 4), $\tilde{Z}$ = (1, 2, 3) *are given. Find realizations x, y, z of* $\tilde{X}$, $\tilde{Y}$, $\tilde{Z}$, *which fulfil the condition* $x \le z \wedge y \le z$ *and such that their degrees of membership functions* $\mu_{\tilde{X}}(x)$, $\mu_{\tilde{Y}}(y)$, $\mu_{\tilde{Z}}(z)$ *are as high as possible*.

The linear programming model for the above example is as follows:

$$\lambda \mapsto \max,$$
$$\lambda \le \lambda_1,$$
$$\lambda \le \lambda_2,$$
$$\lambda \le \lambda_3,$$
$$\lambda_1 \le x \le 2 - \lambda_1,$$
$$2 + \lambda_2 \le y \le 4 - \lambda_2,$$
$$1 + \lambda_3 \le z \le 3 - \lambda_3,$$
$$x \le z,$$
$$y \le z,$$

where λ_1, λ_2, λ_3 are possible levels for the realizations x, y, z.

Library CPLEX finds the optimal solution $\lambda^* = 0.5$, $x = 0.5$, $y = z = 2.5$ (see Figure 4a). Fuzzy number $\tilde{X}$ has realized itself as value $x = 0.5$ with $\mu_{\tilde{X}}(x) = 0.5$ but thre is a better value $x = 1$ with $\mu_{\tilde{X}}(1) = 1$.



a) b)

Fig. 4. Max-min optimal solution a) and improved solution b)

The following goal in IF/PROLOG solves the above linear programming model:

```
[user] ?-  X $>= L1, X $=< 2-L1,
           Y $>= 2+L2, Y $=< 4-L2,
           Z $>= 1+L3, Z $=< 3-L3,
           X $=< Z, Y $=< Z,
           L $=< L1, L $=< L2, L $=< L3,
           linear_maximize(L, MAX).
```

In the answer:

```
X   = __3
L1  = __2
Y   = 0r5/2
L2  = 0r1/2
Z   = 0r5/2
L3  = 0r1/2
L   = 0r1/2
MAX = 0r1/2
```

the values __3 ans __2 mean that there are more than one optimal solution for different values of variables `X` and `L1`.

To obtain a solution with the maximal value of the membership function of realization *x*, the predicate `linear_maximize(L1, MAXL1)` has to be added at the end of the goal:

```
[user] ?- X $>= L1, X $=< 2-L1,
          Y $>= 2+L2, Y $=< 4-L2,
          Z $>= 1+L3, Z $=< 3-L3,
          X $=< Z, Y $=< Z,
          L $=< L1, L $=< L2, L $=< L3,
          linear_maximize(L, MAX),
          Linear_maximize(L1, MAXL1).
```

In the answer:

```
X     = 1
L1    = 1
Y     = 0r5/2
L2    = 0r1/2
Z     = 0r5/2
L3    = 0r1/2
L     = 0r1/2
MAX   = 0r1/2
MAXL1 =1
```

the realization of $\tilde{X}$ is equal to 1 (see Fig. 4b).

In [1], the improved solution for the fuzzy constraints satisfaction problem is defined and algorithms for solving it are proposed.

For the problem considered we propose Algorithm 1. It makes use of predicate `is_constraint(Var)`, which is true when its argument is a variable with domains containing more than one value.

Algorithm 1. Finds the improved solution where λ_1, λ_2, ..., λ_n are the degrees of satisfaction of realizations

1: $L_0 \leftarrow \{\lambda_1, \lambda_2, ..., \lambda_n\}$
2: k $\leftarrow$ 0
3: **while** $L_k \neq \varnothing$ **do**
4: let y_k be a new variable
5: for all $\lambda \in L_k$ do
6: impose constraint $y_k \leq \lambda$
7: **end for**
8: call `linear_maximize`(y_k,_)
9: $L_{k+1} \leftarrow \{\lambda \in L_k | $`is_constraint(`$\lambda$`)`$\}$
10: k $\leftarrow$ +1
11: **end while**

Property 1. *In line 8 of Algorithm 1 at least one variable from set L_k gets a value, so in line 9 set L_{k+1} is a proper subset of the set L_k.*

Proof: All the variables in set L_k have values bounded to interval [0, 1]. The value of variable y_k is their lower bound. If there are no acute inequalities, then the maximal value of y_k is given to at least one variable from set L_k. ®

Proposition 1. *Algorithm 1 performs at most n iterations*.

Proof: The proposition follows immediately from Property 1. ®

The solution found by Algorithm 1 has the following important property:

Property 2. *Let* $\mathbf{u} = (u_1, u_2, \ldots, u_n)$ *be the vector of values of variables λ_1, λ_2, ..., λ_n found by Algorithm 1. If* $\mathbf{v} = (v_1, v_2, \ldots, v_n)$ *is any other vector of feasible values of variables λ_1, λ_2, ..., λ_n, then if for some variable λ_i condition $v_i > u_i$ is fulfilled, then for some other variable λ_j condition $v_j < u_j \leq u_i$ is fulfilled.*

Proof: Suppose that

$$\exists_i (v_i > u_i) \wedge \forall_j (u_j \leq u_i \rightarrow v_j \geq u_j) . \tag{1}$$

Let $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ be the permutation of indices $\{1, 2, \ldots, n\}$ such that $u_{\pi_1} \leq u_{\pi_2} \leq \ldots \leq u_{\pi_n}$. The permutation π has the property that, for any $1 \leq p < q \leq n$, variable λ_{π_p} acquires its value in Algorithm 1 not later than variable λ_{π_q} does.

If vectors **u** and **v** there exists more than one value of i which fulfils (1), then let i be such a value that u_i is the smallest.

Let π_l be the position of u_i in the sequence $(u_{\pi_1}, u_{\pi_2}, \ldots, u_{\pi_n})$. Selection of i which fulfils (1) and corresponds to the minimal value u_i assures that $u_{\pi_k} = v_{\pi_k}$, for all $k = 1, 2, \ldots, l - 1$:

$$\begin{array}{ccccccccccc} u_{\pi_1} & \le & u_{\pi_2} & \le & \ldots & \le & u_{\pi_{l-1}} & \le & u_{\pi_l} & \le \ldots \le & u_{\pi_n} \\ \| & & \| & & & & \| & & \wedge & & \\ v_{\pi_1} & \le & v_{\pi_2} & \le & \ldots & \le & v_{\pi_{l-1}} & < & v_{\pi_l} & \ldots & v_{\pi_n} \end{array}$$

Variable λ_{π_l} has reached in line 8 of Algorithm 1 a value u_{π_l}, which is maximal, but this is inconsistent with the fact that the value v_{π_l} is also feasible and $v_{\pi_l} > u_{\pi_l}$. ®

In [4], a new computer representation of fuzzy numbers and fuzzy constraint was proposed. It was used Kobylański and Zieliński in package `const_fuzzy` for fuzzy Modelling. Algorithm 1 was implemented in `const_fuzzy` to find improved solutions.

The package `const_fuzzy` contains the following predicates:

- `fuzzy(F, [A, B, C, …])` – defines fuzzy number *F* (interval, triangular or trapezoidal) with parameters *A*, *B*, *C*, …,
- `fconstr(FC, [X, Y, Z, …], C)` – defines fuzzy constraint *FC* with logical condition *C* on fuzzy numbers *X*, *Y*, *Z*, …,
- `poss(FC, P, V)` – computes (by Algorithm 1) the maximal degree *P* of satisfaction of fuzzy constraint *FC* and assigns to variable *V* a list of realizations of fuzzy numbers in *FC* and values of their membership functions.

The following dialog, which makes use of predicates from package `const_fuzzy`, solves Example 2:

```
[user] ?-   fuzzy(X, [0, 1, 2]),
            fuzzy(Y, [2, 3, 4]),
            fuzzy(Z, [1, 2, 3]),
            fconstr(FC, [X, Y, Z], (X =< Z, Y =< Z)),
            poss(FC, POSS, VECT).

POSS      = 0r1/2
VECT      = [f(1,1), f(0r5/2, 0r1/2), f(0r5/2, 0r1/2)]
```

List VECT = [$f(x, \lambda_1)$, $f(y, \lambda_2)$, $f(z, \lambda_3)$] = [$f(1, 1)$, $f(5/2, 1/2)$, $f(5/2, 1/2)$] represents a solution which is shown in Figure 4b.

5. Two stage method

In [5], two heuristic for the problem considered were presented. In this paper, a second stage is proposed. In the first stage, the set of tours is constructed and in the second stage, an improved schedule with the degrees of satisfactions of all the temporal constraints being as high as possible is found.

5.1. Tours construction

The heuristic algorithms proposed make use of the following two functions:
- INSERT(i, T) – creates a new tour by inserting customer i into tour T [8],
- MERGE(T_1, T_2) – creates a new tour by merging two tours T_1 and T_2.

5.1.1. Construction by insertion

The algorithm starts with a set of tours S_0 which consists of K empty tours [0, 0].

In the following iterations $i = 1, 2, \ldots, n$, there is created a new set of tours S_i by insertion of one of the customers who have not been served yet into one of the K tours, from the set S_{i-1}, in one of possible positions. The selection of the customer, tour and position is made in such a way that the value of $\mathrm{SAT}_{\tilde{G}}(S_i)$ is maximized.

5.1.2. Construction by merging

The algorithm starts with a set of tours S_0 which consists of n tours [0, i, 0], for every customer $i = 1, 2, \ldots, n$.

In the following iterations $i = 1, 2, \ldots, n - K$, there is created a new set of tours S_i by replacing two tours from the set S_{i-1} with their merge. The selection of the two tours is made in such a way that the value of $\mathrm{SAT}_{\tilde{G}}(S_i)$ is maximized.

Let $L_1 \,\|\, L_2$ and REVERSE(L) denote respectively concatenation of two lists and the reverse of list. I the computational experiments the operation MERGE was carried out in the following way:

If T_1 = [0]‖L_1‖[0] and T_2 = [0]‖L_2‖[0] are two tours, then MERGE(T_1, T_2) = [0]‖L‖[0], where L is one of the following eight lists:

L_1‖L2,	L2‖L1,
L1‖REVERSE(L2),	REVERSE(L2)‖L1,
REVERSE (L1)‖L2,	L2‖REVERSE(L1),
REVERSE(L1)‖REVERSE(L2),	REVERSE(L2)‖REVERSE(L1).

5.2. Improved schedule

For the set of tour which was constructed in the first stage, the max-min optimal schedule is improved.

The set of tours is stored as the set of K lists L_1, L_2, …, L_K. Each list L_i contains indexes of customers serviced by the i-th vehicle, n_i is the number of customers on list L_i and L_{ij} is the index of the j-th customer on list L_i.

In Figure 5, an example of two tours is presented, where $n_1 = 3$, $n_2 = 2$, $L_{11} = 3$, $L_{12} = 1$, $L_{13} = 5$, $L_{21} = 2$, $L_{22} = 4$.

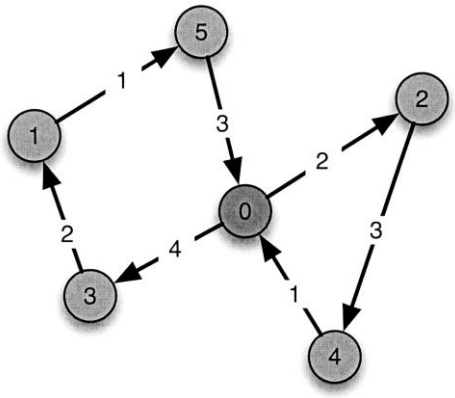


Fig. 5. Two tours [0, 3, 1, 5, 0] and [0, 2, 4, 0]
(the travel times are given on arcs)

The depot is opened in interval $[e_0, l_0]$ and for each customer $i \in N$ the time window $\tilde{T}\tilde{W}_i$ as a fuzzy triangular number (a_i, b_i, c_i, d_i) is given. The service time in the depot and at the customer equals s_i, for $i \in N_0$.

For $i, j \in N_0$ the travel time t_{ij} between the depot and the customers is a crisp real number.

The improved schedule is computed by Algorithm 2, which uses Algorithm 1 in predicate `poss/3`.

Algorithm 2. Computing the improved schedule.

1: **for all** $i \in N$ **do**
2: let T_i be a new variable
3: call `fuzzy` (T_i, [a_i, b_i, c_i, d_i])
4: **end for**
5: let G be a new variable
6: call `fuzzy` (G, [0, 0, g_1, g_2])
7: **for all** $k = 1, 2, \ldots, K$ **do**
8: let S_k be a new variable
9: call `fuzzy` (S_k, [e_0, l_0])
10: let F_k be a new variable
11: call `fuzzy` (F_k, [e_0, l_0])
12: **end for**

13: let φ_0 be a constraint $G = \Sigma_{k=1}^{K}(F_k - S_k)$
14: **for all** $k = 1, 2, \ldots, K$ **do**
15: let φ_k be a constraint $(T_{L_{k,1}} \geq S_k + s_0 + t_{0,L_{k,1}}) \wedge \bigwedge_{i=2}^{n_k}(T_{L_{ki}} \geq T_{L_{k,i-1}} + s_{L_{k,i-1}} + t_{L_{k,i-1},L_{ki}})$
$\wedge (F_k \geq T_{L_{k,n_k}} + s_{L_{k,n_k}} + t_{L_{k,n_k},0})$
16: **end for**
17: Let L be a list $[T_1, T_2, \ldots, T_n, S_1, S_2, F_2, \ldots, S_k, F_k, G]$
18: call `fconstr`$(FC, L, \varphi_0 \wedge \varphi_1 \wedge \ldots \wedge \varphi_k)$
19: call poss(FC, P, V)

Constraint φ_0 (in line 13 of Algorithm 2) corresponds to the satisfaction of the total time of realization of all the tours and constraint φ_k (in line 15) corresponds to the feasibility of the schedule for the k-th vehicle.

The following example shows hot the improved schedule could be found with IF/PROLOG:

Example 3. *Customers* $N = \{1, 2, 3, 4, 5\}$ *have the same triangular time window* (2, 5, 8). *They are served by two vehicles and the set of tours is* $S = \{[0, 3, 1, 5, 0], [0, 2, 4, 0]\}$ (*travel times are given in Fig. 5*). *All the service times are equal to* $0(s_i = 0,$ *for* $i \in N_0)$.

The depot is opened from 0 to 11 and the satisfaction of the total realization time is expressed with fuzzy triangular number $\tilde{G}$ = (0, 0, 10, 20).

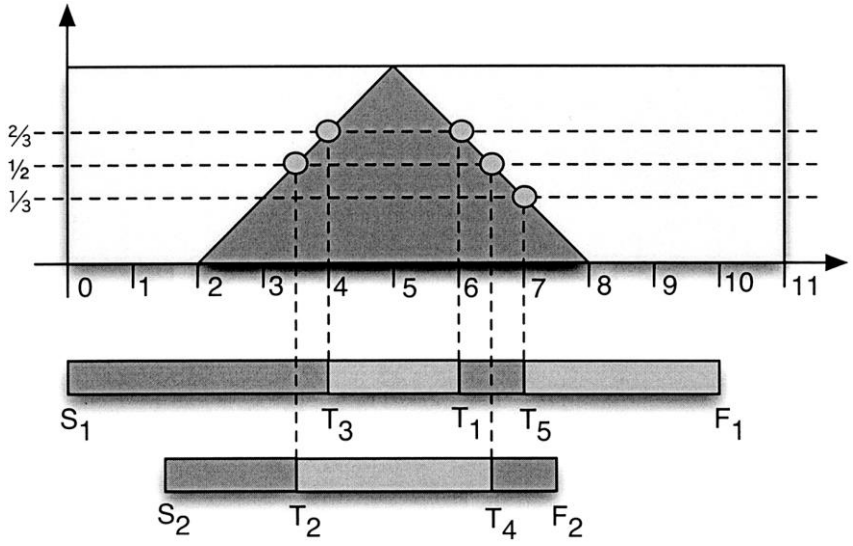


Fig. 6. Improved schedule for tours in Example 3

The following query in IF/PROLOG solves Example 3.

```
[user] ?- fuzzy(T1, [2, 5, 8])
          fuzzy(T2, [2, 5, 8]),
          fuzzy(T3, [2, 5, 8]),
          fuzzy(T4, [2, 5, 8]),
          fuzzy(T5, [2, 5, 8]),
          fuzzy(G, [0, 0, 10, 20]),
          fuzzy(S1, [0, 11]),
          fuzzy(S2, [0, 11]),
          fuzzy(F1, [0, 11]),
          fuzzy(F2, [0, 11]),
          fconstr(FC, [T1, T2, T3, T4, t5, S1, F1, S2, F2, G], (
               G = (F1-S1) + (F2-S2),
               T3 >= S1+4, T1 >= T3 +2, T5 >= T1+1, F1 >= T5+3,
               T2 >= S2+2, T4 >= T2 +3, F2 >= T4+1)),
          poss(FC, P, V).
```

In the answer the degree of satisfaction of all the constraints is equal to $P = 1/3$ and vector contains the following improved schedule (presented on Gantt diagram in Figure 6):

$$T_1 = 6, \quad \mu_{\tilde{\tilde{TW}}_1}(T_1) = \frac{2}{3}, \quad T_2 = \frac{7}{2}, \quad \mu_{\tilde{\tilde{TW}}_2}(T_2) = \frac{1}{2},$$

$$T_3 = 4, \quad \mu_{\tilde{\tilde{TW}}_3}(T_3) = \frac{2}{3}, \quad T_4 = \frac{13}{2}, \quad \mu_{\tilde{\tilde{TW}}_4}(T_4) = \frac{1}{2},$$

$$T_5 = 7, \quad \mu_{\tilde{\tilde{TW}}_5}(T_5) = \frac{1}{3}, \quad S_1 = 0, \quad F_1 = 10,$$

$$S_5 = \frac{3}{2}, \quad F_2 = \frac{15}{2}, \quad F = 16, \quad \mu_{\tilde{G}}(G) = \frac{2}{5}.$$

6. Conclusions

The method proposed in this paper for obtaining an improved solution to the problem considered eliminates the main drawback of the max-min optimal solution relying on the low degree of satisfaction of the time windows or the goal constraints.

The crucial problem especially in transportation problems is obtaining a solution satisfying to the highest possible degree the time window constraints (then the preferences of the customers are taken into account) and the goal constraint (then the preferences of transport agent concerning the work time of the vehicle are taken into account).

In our opinion the constraint programming is the best tool for the implementation of the method of obtaining the improved solution.

Acknowledgements

This research was supported by grant no. 7T1102120 from the State Committee for Scientific Research (Komitet Badań Naukowych).

References

[1] DUBOIS D., FORTEMPS Ph., *Computing improved optimal solutions to max-min flexible constraint satisfaction problems*, European Journal of Operations Research, 1999, 118, 95–126.
[2] FISHER M.L., JÖRNSTEN K.O., MADSEN O.B.G., *Vehicle Routing with Time Windows: Two Optimization Algorithms*, Operations Research, 1997, Vol. 45, No. 3, May–June.
[3] IF/PROLOG V5.0 Constraint Package. *Siemens Nixdorf Informationssysteme AG*, 1994.
[4] KOBYLAŃSKI P., ZIELIŃSKI P., *Fuzzy modelling with constraint technology*, proceedings of EUROFUSE Workshop on Information Systems, Varenna, Italy 2002.
[5] KOBYLAŃSKI P., KULEJ M., *Vehicle Routing and Scheduling with Fuzzy Time Windows and Fuzzy Goal* [in:] Proceedings of an International Conference in Fuzzy Logic and technology, 10–12 September 2003, Zittau, Germany.
[6] KULEJ M., FLORKIEWICZ B., *A heuristic algorithm for the multi-trip vehicle routing and scheduling problem with time windows*, Central European Journal for Operations Research and Economics, 1997, Vol. 5, No.3/4, 295–315.
[7] SOLOMON M.M., *On the Worst-Case Performance of Some Heuristics for the Vehicle Routing and Scheduling Problem with Time Window Constraints*, Networks, 1986, 16, 161–174.
[8] SOLOMON M.M., *Algorithms for Vehicle Routing and Scheduling Problems with Time Windows Constraints*, Operations Research, 1987, 35, 254–265.
[9] ZIMMERMANN H.-J., *Fuzzy Set Theory and Applications*, Kluwer Academic Publishers, 1991, Boston–Dordrecht–London.

Poprawione rozwiązania dla zagadnienia planowania i harmonogramowania tras dla samochodów z rozmytymi oknami czasowymi klientów i rozmytym celem

W artykule omówiono problem planowania i harmonogramowania tras dla samochodów w warunkach istnienia rozmytych okien czasowych klientów i rozmytego celu. Sformułowano model mieszany programowania całkowitoliczbowego bazujący na kryterium max-min i wykorzystujący zasadę uogólniania Zadeha. Rozwiązanie optymalne tego modelu, nazywane rozwiązaniem max-min optymalnym, ma tendencję obniżania stopni satysfakcji ograniczeń czasowych lub celu. W celu wyeliminowania tego mankamentu zastosowano koncepcję poprawionych rozwiązań optymalnych (Dubois i Fortemps, 1999). Zaproponowano dwuetapową metodę znajdowania takich rozwiązań, opartą na programowaniu z więzami jako efektywnym narzędziem do rozwiązywania rozpatrywanego problemu.