Implementation of a decision support model of the supply chain management in the environment of constraint logic programming

Jarosław Wikarek

Institute of Management Control Systems, Kielce University of Technology, Poland

Abstract: The article presents the details of the implementation of the concept of a decision support model in the supply chain. To implement the model, the CLP (Constraint Logic Programming) framework called Eclipse was used. The novel way of constraints propagation is discussed, which for this class of problems improves significantly the efficiency of a search for a solution. The most important predicates implementing the ‚model are presented and characterized. Several numerical examples are included to illustrate the implementation of the approach.

Keywords: Supply Chain Management (SCM), Decision Support, Constraint Satisfaction Problem (CSP), Constraint Logic Programming (CLP)

The concept of constraints was used originally in physics and combinatorial optimization. It was first applied in computer science for describing interactive drawing system Sketchpad in 1963. In the following decade, several experimental languages were proposed that used the notion of constraints and concept of constraint solving. At time in the field of Artificial Intelligence (AI), the concept of constraint satisfaction problem (CSP) was formulated and used to describe computer vision. In the nineties, there was a rapid development of constraint-based environments. Commercial systems were created such as CHIP, ILOG and freeware like Eclipse.

1. Introduction

According to the author of this paper, the CSP [1] offers a very good framework for representing the knowledge and information needed for the supply chain management.

A CSP consists of a set of variables and a set of constraints that must be satisfied by those constraints. In the supply chain domain, many business rules can be easily represented as constraints. CSPs are used often in constraint programming. Constraint programming is the use of constraints as a programming language to encode and solve problems. Most problems that the constraint programming concerns belong to the group that conventional programming techniques finds the hardest. Time needed to solve such problems using unconstrained search increases exponentially with the problem size. The aim of this paper is to present the detailed implementation of the CSP-based model for Supply Chain Management (SCM) in the constraint logic programming (CLP) framework. In addition, this paper presents a new way of problem representation, which, together with a novel approach to constraints propagation, significantly improves the efficiency of searching for solutions.

The use of the constraint-based environment for modeling and solving decision problems in SCM is interesting for two reasons. First, it is possible to implement various types of constraints: linear, non-linear, logical, etc. Through the flexible use of methods and tools CLP-class systems, higher efficiency of searching for solutions can be obtained.

2. Constraint logic programming

Constraint Logic Programming (CLP) is a programming paradigm that represents a successful attempt to merge the best features of logic programming (LP) and constraint solving. CLP is also a tool for solving constraint satisfaction problems (CSP)[2, 5]. For the important combinatorial case, the following features characterize a CSP:

- a finite set S of integer variables X_1, .. ,X_n, with values from finite domains D_1, .. ,D_n,
- a set of constraints between variables: The i-th constraint $C_i(X_{i1}$, ..,X_{ik}) between k variables from S is given by a relation defined as subset of the Cartesian product $D_{i1} \times$, ..,$\times D_{ik}$ that determines variable values corresponding to each other in a sense defined by the problem considered,
- a CSP solution is given by any assignment of domain values to variables that satisfies all constraints.

Developing from LP to CLP, the concept of unification is generalized to constraint solving: the relationship between a goal and a clause (to be used in a resolution step) can be described not only via term equations but also via more general statements, i.e. constraints.

The semantics of constraint logic programs can be defined in terms of a virtual interpreter that maintains a pair <G, S> during execution. The first element of this pair is called a current goal; the second element is called constraint store. The current goal contains the literals the interpreter is trying to prove and may also contain some constraints it is trying to satisfy; the constraint store contains all constraints the interpreter has assumed satisfiable. At the beginning, the current goal is the goal

and the constraint store is empty. The interpreter proceeds by removing the first element from the current goal and analyzing it. In the end, this analysis should produce a successful termination or a failure. This analysis could involve recursive calls and addition of new literals to the current goal and new constraint to the constraint store. The interpreter backtracks if a failure is generated. A successful termination is generated when the current goal is empty and the constraint store is satisfiable. CLP can use Artificial Intelligence (AI) techniques to improve the search: propagation, data-driven computation, "forward checking" and "lookahead"[1].

In the paper for modeling and solving decision problems Eclipse open-source software was used [3]. Eclipse is a software system for the development and deployment of constraint programming and constraint logic programming applications. It contains several constraint solver libraries, a high-level modeling and control language, interfaces to third-party solvers, a development environment and interfaces for embedding into host environments.

3. Implementation of the model

Implementation of the model for decision support in SCM was made in Eclipse. A detailed description of the model, a discussion of constraints, parameters and decision variables are presented in [4] and tab. 1. In the construction of the model, the following assumptions are valid:

- the shared information process in the supply chain consists of resources (capacity, versatility, costs), inventory (capacity, versatility, costs, time), production (capacity, versatility, costs), product (volume), transport (cost, mode, time), demand, etc.,
- the part of the supply chain has a structure as in fig. 1,
- the transport is multimodal (several modes of transport; a limited number of means of transport for each mode),
- the environmental aspects of use of transport modes,
- different products are combined in one batch of transport,
- the cost of supplies is presented in the form of a function (in this approach linear function of fixed and variable costs),
- restrictions on the common distribution of certain products can occur.

There are also a few assumptions about the implementation of the model:

- the knowledge related to the supply chain management is presented in a linear and logical constraints,
- a decision model is formulated as a constraint satisfaction problem (CSP),
- a novel method of constraints propagation is used, which fundamentally improves the efficiency of finding the solution.

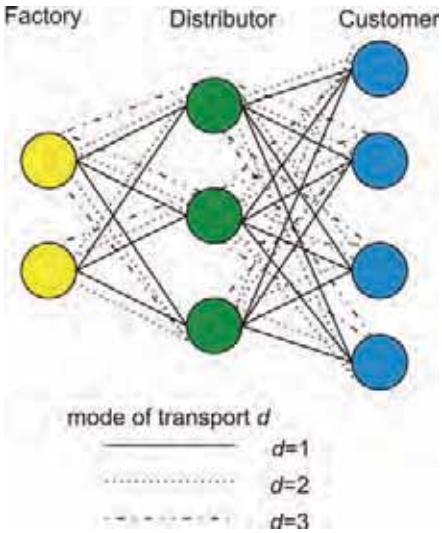

Fig. 1. The simplified structure of the supply chain network (all routes)

Rys. 1. Uproszczona struktura łańcucha dostaw (wszystkie marszruty)

Tab. 1. Summary indices, parameters and decision variables of the model

Tab. 1. Indeksy, parametry i zmienne decyzyjne modelu

Symbol	Description
Indices	
k	product type (k = 1..O)
j	delivery point/customer/city (j = 1..M)
i	manufacturer/factory (i = 1..N)
s	distributor /distribution center (s = 1..E)
d	mode of transport (d = 1..L)
N	number of manufacturers/factories
M	number of delivery points/customers
E	number of distributors
O	number of product types
L	number of mode of transport
Input parameters	
F_s	the fixed cost of distributor/distribution center *s* (s = 1..E)
P_k	the area/volume occupied by product *k* (k = 1..O)
V_s	distributor *s* maximum capacity/volume (s = 1..E)
$W_{i,k}$	production capacity at factory *i* for product *k* (i = 1..N) (k = 1..O)
$C_{i,k}$	the cost of product *k* at factory *i* (i = 1..N) (k = 1..O)
$R_{s,k}$	if distributor *s* (s = 1..E) can deliver product *k* (k = 1..O) then $R_{sk} = 1$, otherwise $R_{sk} = 0$
$Tp_{s,k}$	the time needed for distributor *s* (s = 1..E) to prepare the shipment of product *k* (k = 1..O)
$Tc_{j,k}$	the cut-off time of delivery to the delivery point/customer *j* (j = 1..M) of product *k* (k = 1..O)
$Z_{j,k}$	customer demand/order *j* (j = 1..M) for product *k* (k = 1..O)
Zt_d	the number of transport units using mode of transport *d* (d = 1..L)
Pt_d	the capacity of transport unit using mode of transport *d* (d = 1..L)

$Tf_{i,s,d}$	the time of delivery from manufacturer i to distributor s using mode of transport d (i = 1..N) (s = 1..E) (d = 1..L)
$K_{1i,s,k,d}$	the variable cost of delivery of product k from manufacturer i to distributor s using mode of transport d (d = 1..L) (i = 1..N) (s = 1..E) (k = 1..O)
$R_{1i,s,d}$	if manufacturer i can deliver to distributor s using mode of transport d then $R1_{isd}$=1, otherwise $R1_{isd} = 0$ (d = 1..L) (s = 1..E) (i = 1..N)
$A_{i,s,d}$	the fixed cost of delivery from manufacturer i to distributor s using mode of transport d (d = 1..L) (i = 1..N) (s = 1..E)
$Koa_{s,j,d}$	the total cost of delivery from distributor s to customer j using mode of transport d (d = 1..L) (s = 1..E) (j = 1..M)
$Tm_{s,j,d}$	the time of delivery from distributor s to customer j using mode of transport d (d = 1..L) (s = 1..E) (j = 1..M)
$K2_{s,j,k,d}$	the variable cost of delivery of product k from distributor s to customer j using mode of transport d (d = 1..L) (s = 1..E) (k = 1..O) (j = 1..M)
$R2_{sjd}$	if distributor s can deliver to customer j using mode of transport d then $R2_{sjd}$=1, otherwise $R2_{s,j,d}$=0 (d = 1..L) (s = 1..E) (j = 1..M)
$G_{s,j,d}$	the fixed cost of delivery from distributor s to customer j using mode of transport d (s = 1..E) (j = 1..M) (k = 1..O)
$Kog_{s,j,d}$	the total cost of delivery from distributor s to customer j using mode of transport d (d = 1..L) (s = 1..E) (j = 1.M) (k = 1..O)
Od_d	the environmental cost of using mode of transport d (d = 1..L)
Decision variables	
$X_{i,s,k,d}$	delivery quantity of product k from manufacturer i to distributor s using mode of transport d
$Xa_{i,s,d}$	if delivery is from manufacturer i to distributor s using mode of transport d then $Xa_{i,s,d} = 1$, otherwise $Xa_{i,s,d} = 0$
$Xb_{i,s,d}$	the number of courses from manufacturer i to distributor s using mode of transport d
$Y_{s,j,k,d}$	delivery quantity of product k from distributor s to customer j using mode of transport d
$Ya_{s,j,d}$	if delivery is from distributor s to customer j using mode of transport d then $Ya_{s,j,d} = 1$, otherwise $Ya_{s,j,d} = 0$
$Yb_{s,j,d}$	the number of courses from distributor s to customer j using mode of transport d
Tc_s	if distributor s participates in deliveries, then $Tc_s = 1$, otherwise $Tc_s = 0$
CW	Arbitrarily large constant

In the classical method of implementation (fig. 2) on the basis of the facts contained in the files *orders.ecl* and *configuration.ecl*, adequate representation of the problem is generated and, together with the facts, used in the file *op.ecl*. The file o*p.ecl* contains a set of predicates implementing the decision model under constraints [4].

The proposed novel implementations of the problem introduced an additional step of generation marked with a dashed line in fig. 3. The generation process is based on the facts of the files *configuration.ecl* and *orders.ecl* and results in placing all feasible routes as well as other feasible facts in files *routes.ecl* and *others.ecl* in a sequential order. In this approach, the representation of the problem is also different because it contains only one value that is not set while in the classical approach there are five such values. Then all feasible facts, and the facts of *orders.ecl* file are transferred to the main file *opn.ecl* (fig. 3). The intermediate step associated with the generation of feasible facts based on the knowledge of the problem structure fundamentally increases the scope of propagation of constraints and narrows the domains of decision variables.

The structure of the main predicates developed in the process of implementation and their descriptions are shown in tab. 2.

Tab. 2. Predicate descriptions
Tab. 2. Opis predykatów i faktów

Predicate name
Description
product(name, capacity)
The predicate of facts describing the volume of individual products
customer(name_c)
The predicate of facts describing the customers
distributor(name_d,capacity,cost)
The predicate of facts describing the capacity and cost of individual distributors
factory(name_f)
The predicate of facts describing the factories
transport_unit(name_tu, capacity, quantity, cost)
The predicate of facts describing the capacity, quantity, and cost of individual transport units
factory_Distributor_Transport_unit (name_f,name_d,name_tu, cost, time)
Predicates describing the costs and time possible connections between the factory and the distributor carried out the selected means of transport
distributor_Customer_Transport_unit(name_d, name_c, name_tu, cost, time)
Predicates describing the costs and time possible connections between the distributor and the customer carried out the selected means of transport
distributor_product(name_d,product,time)
Predicates specify whether the product and service time by a given distributor
factory_product(name_f, product, capacity, cost)
Predicates defining the capacity and cost of the product
exclusion_d(name_d, product, product)
Lock simultaneous distribution of two selected products by the distributor
order(name_o, product, name_c, time, quantity)
Orders specifying customer demand for a product and delivery date

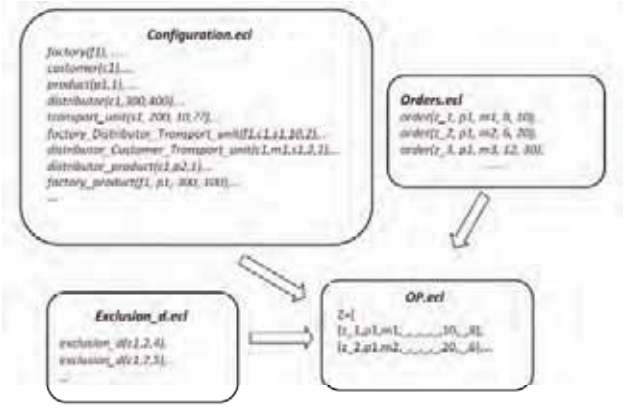

Fig. 2. Block diagram of the implementation of the decision-making model in the framework CLP – classical approach

Rys. 2. Schemat blokowy implementacji modelu decyzyjnego w środowisku CLP – podejście klasyczne

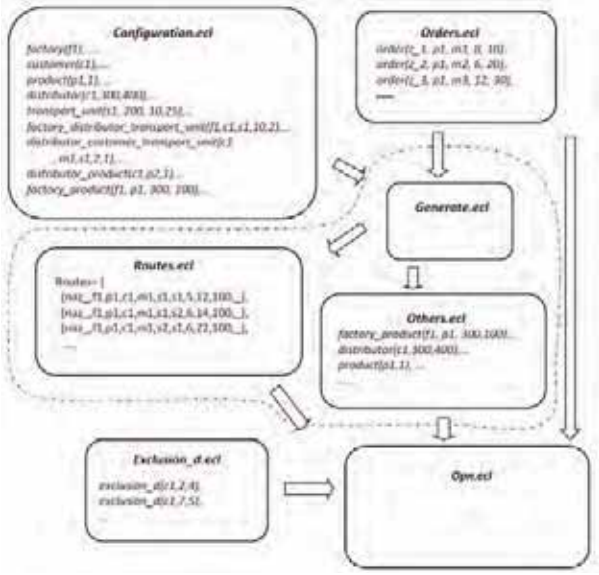

Fig. 3. Block diagram of the implementation of the decision-making model in the CLP framework – the novel approach, extra step marked by the dashed line

Rys. 3. Schemat blokowy implementacji modelu decyzyjnego w środowisku CLP – nowe podejście z zaznaczonym dodatkowym krokiem

The difference between the two approaches is not just a different implementation schema (fig. 2–3), but also a different representation of the problem in the form of terms. In the classical approach, the problem takes on a character representation of the vector (fig. 4a), which is created and processed in the main file *op.ecl*. There, in the search for solutions, using the methods of the constraints propagation and labelling further vectors are created for each of the variables. As shown in fig. 4b, for each vector there are 5 variables to be determined, defining the size of the delivery, factories and distributors involved in the supply and transport.

The approach proposed in the paper (fig. 3) developed another form of representation of the problem. Representation of a set of vectors representing the feasible supply routes is supplemented by the size of the delivery (fig. 5). This set is the file *routes.ecl* generated earlier via *generate.ecl* and then, in the form of vectors with fixed values for factories, distributors, transportation is forwarded to the main file *ecn.ecl*. The process is determined by solving only one value, i.e. the size of the delivery.

This approach results in a large increase in the constraints propagation, as fixed variables significantly cut the variable domain. In addition, a reduction in backtracking processes takes place because backtracking occurs only between the following vectors and not, as in the classical approach, also between the variables within the vector.

Symbols necessary to understand both the representation of the problem and their descriptions are presented in tab. 3.

[Z_n,P,M,D,F,Tu,Tu,Oq,X,T]

Fig. 4a. Representation of the problem in the classical approach – definition

Rys. 4a. Reprezentacja problemu w podejściu klasycznym – definicja

[[z_1,p1,m1,_,_,_,_,10,_,8],
[z_2,p1,m2,_,_,_,_,20,_,6],…]

Fig. 4b. Representation of the problem in the classical approach – the process of finding a solution

Rys. 4b. Reprezentacja problemu w podejściu klasycznym – proces znajdowania rozwiązania

[[naz_1,f1,p1,c1,m1,s1,s1,5,12,100,_],
[naz_2,f1,p1,c1,m1,s1,s2,6,14,100,_],
[naz_3,f1,p1,c1,m1,s2,s1,6,22,100,_],…]

Fig. 5. Representation of the problem in the novel approach – set of feasible routes

Rys. 5. Reprezentacja problemu w nowym podejściu – zbiór dopuszczalnych tras

Tab. 3. Symbols used in the representation of the problem

Tab. 3. Symbole wykorzystane w reprezentacji problemu

Symbol	Description
Z_n	order number
P	products, P∈ $\{p_1,p_2, \ldots ,p_o\}$
M	customers, M∈$\{m_1,m_2, \ldots m_m\}$
D	distributors, D∈$\{c_1,c_2, \ldots c_e\}$
F	factories, F∈$\{f_1,f_2, \ldots f_n\}$
Tu	transport unit, Tu∈$\{s_1,s_2, \ldots s_l\}$
T	delivery time/period
Oq	order quantity
X	delivery quantity
Naz_	routes name-number

4. Computational examples

In order to verify and evaluate the proposed approach, several computational experiments were performed.
All the cases relate to the supply chain with two manufacturers ($i = 1..2$), three distributors ($s = 1..3$), four customers ($j = 1..4$), four mode of transport ($d = 1..4$), and five types of products ($k = 1..5$).

Numerical examples with different input data sets from *orders.ecl* were computed. The number of orders (Orders_N) in specific examples varied from 2 to 20.

The objective function value obtained for the classical approach (FCs), the novel approach (FCn) and computation time (in seconds) is shown in tab. 2.

Tab. 4. The results of numerical examples for both approaches
Tab. 4. Wyniki przykładów liczbowych dla obu podejść

Orders_N	Ts	Tn	FCs	FCn
2	0,01	0,02	3 424	3 424
4	0,02	0,03	8 555	85 55
6	0,06	0,04	14 881	14 881
8	18,72	0,05	36 858	36 363
10	6012	0,08	51 234	50 937
12	-*	0,09	-	57 285
16	-*	0,17	-	84 348
20	-*	0,23	-	88 440

*The calculation was discontinued after 50 000 seconds

Full data sets for these examples are shown in tab. 5. Symbols were taken as described in tab. 1. The results for the largest example of the 20 orders are presented in a tab. 6 and route schemes (fig. 6).

Tab. 5. The set of parts of data tables for the example with Orders_N = 20
Tab. 5. Fragmenty tabel z danymi dla przykładu z 20 zleceniami Orders_N = 20

k	P_k
p1	1
p2	1
p3	3
p4	2
P5	3

s	F_s	V_s
c1	300	400
c2	250	350
c3	250	500

s	k	Tp_{sk}
c1	p1	2
c1	p2	2
c1	p3	2
c1	p4	2
c2	p1	1
c2	p2	1
c2	p3	1
c2	p4	1
c3	p1	3
c3	p2	3
c3	p3	3
c3	p4	3

d	Pt_d	Zt_d	Od_d
s1	200	10	25
s2	300	10	30
s3	400	10	35

j
m1
m2
m3
m4

i
f1
f2

i	k	W_{ik}	C_{ik}
f1	p1	300	100
f1	p2	0	0
f1	p3	100	200
f1	p4	300	300
f1	p5	300	300
f2	p1	0	0
f2	p2	300	210
f2	p3	300	150
f2	p4	300	250
f2	p5	0	0

i	s	d	A_{isd}	Tf_{isd}
f1	c1	s1	10	2
f1	c1	s2	20	3
f1	c1	s3	40	4
f1	c2	s1	12	1
f1	c2	s2	24	2
f1	c2	s3	42	3
f1	c3	s1	5	1
f1	c3	s2	10	2
f1	c3	s3	15	3
f2	c1	s1	5	4
f2	c1	s2	10	6
f2	c2	s1	10	4
f2	c2	s2	20	6
f2	c2	s3	40	7
f2	c3	s1	15	4
f2	c3	s2	25	6

s	j	d	G_{sjd}	Tm_{sjd}	s	j	d	G_{sjd}	Tm_{sjd}
c1	m1	s1	2	1	c2	m2	s3	15	2
c1	m1	s2	4	2	c2	m3	s1	5	1
c1	m2	s1	2	1	c2	m3	s2	10	1
c1	m2	s2	5	1	c2	m4	s1	2	1
c1	m2	s3	12	2	c2	m4	s2	4	1
c1	m3	s1	14	1	c3	m1	s1	2	1
c1	m3	s2	12	1	c3	m1	s2	4	1
c1	m3	s3	20	2	c3	m2	s1	3	1
c1	m4	s1	15	1	c3	m2	s2	6	1
c1	m4	s2	13	1	c3	m2	s3	14	2
c1	m4	s3	30	2	c3	m3	s1	6	1
c2	m1	s1	4	1	c3	m3	s2	10	1
c2	m1	s2	8	1	c3	m3	s3	20	2
c2	m1	s3	16	2	c3	m4	s1	4	1
c2	m2	s1	3	1	c3	m4	s2	8	1
c2	m2	s2	6	1	c3	m4	s3	20	2

N	k	j	T_{jk}	Z_{jk}	N	k	j	T_{jk}	Z_{jk}
Z_1	p1	m1	8	10	Z_10	p3	m3	12	40
Z_2	p1	m2	6	20	Z_19	p3	m4	12	5
Z_3	p1	m3	12	30	Z_20	p4	m1	12	5
Z_18	p1	m4	12	5	Z_13	p4	m2	8	10
Z_4	p2	m1	12	10	Z_11	p4	m3	8	10
Z_5	p2	m2	8	10	Z_12	p4	m4	12	10
Z_6	p2	m3	12	20	Z_14	p5	m1	12	20
Z_7	p2	m4	8	45	Z_15	p5	m2	12	30
Z_8	p3	m1	12	60	Z_16	p5	m3	8	30
Z_9	p3	m2	6	40	Z_17	p5	m4	8	5

k	s	k	k	I	k
p1	c1	p2	p1	f1	p2
p1	c2	p2	p1	f2	p2
p1	c3	p2			

Tab. 6. Results for computational example Order_N = 20
Tab. 6. Wyniki dla przykładu z 20 zleceniami Order_N = 20

N	i	k	s	j	d_1	d_2	X
Z_1	f1	p1	c1	m1	s1	s2	10
Z_2	f1	p1	c1	m2	s1	s1	20
Z_3	f1	p1	c1	m3	s1	s1	30
Z_18	f1	p1	c1	m4	s1	s1	5
Z_4	f2	p2	c3	m1	s1	s1	10
Z_5	f2	p2	c3	m2	s1	s1	10
Z_6	f2	p2	c3	m3	s1	s1	20
Z_7	f2	p2	c3	m4	s1	s1	45
Z_8	f1	p3	c1	m1	s1	s1	45
Z_8	f1	p3	c1	m1	s2	s1	15
Z_9	f1	p3	c1	m2	s2	s1	40
Z_10	f2	p3	c1	m3	s2	s1	11

Z_10	f2	p3	c3	m3	s1	s1	29
Z_19	f2	p3	c3	m4	s1	s1	5
Z_20	f1	p4	c1	m1	s2	s1	1
Z_20	f1	p4	c2	m1	s2	s2	4
Z_13	f1	p4	c2	m2	s2	s2	10
Z_11	f1	p4	c2	m3	s2	s2	10
Z_12	f1	p4	c2	m4	s2	s2	10
Z_14	f1	p5	c2	m1	s2	s2	20
Z_15	f1	p5	c2	m2	s2	s2	30
Z_16	f1	p5	c2	m3	s2	s2	30

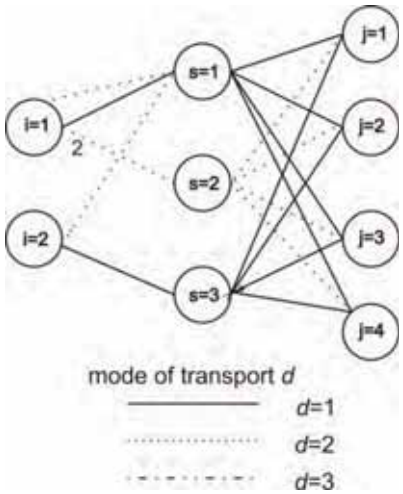

Fig. 6. Transport network of solution (FCn = 88 440) for example Orders_N = 20.

Rys. 6. Sieć transportowa dla przykładu przy Orders_N = 20 o wartości funkcji oceny (FCn = 88 440)

5. Conclusions

The experiments confirmed the correctness of the assumptions. We found that an increase in the propagation of constraints has a critical influence on the process of finding a solution. For larger examples, finding a feasible solution is a long and difficult process if the constraints propagation is insufficient.

Adopted solutions are innovative in nature. Changing the implementation and representation of the problem that in its nature has many decision variables subject to summing up allowed us to increase the range constraints propagation and reduce the backtracking process.

Therefore, the proposed solution is highly recommended for all types of decision problems in SCM or a similar structure. This structure is characterized by the constraints of many decision variables and their summing. The results from the implementation of the model allow specifying the cost of the order fulfillment process, determining the distribution of products and the use of modes of transport.

References

1. Apt K., Wallace M., *Constraint Logic Programming using Eclipse*, Cambridge University Press, 2006.
2. Williams, H.P., *Logic and Integer Programming*, Springer, Berlin 2009.
3. [www.eclipseclp.org] – EclipseHome (22.11.2012).
4. Sitek P., *Application of constraint logic programming to decision support for the supply chain management*, "Pomiary Automatyka Robotyka", 2/2013.
5. Niederliński A., *A Quick and Gentle Guide to Constraint Logic Programming via ECLiPSe*, [pkjs.com.pl], Gliwice 2011.

Implementacja modelu wspomagania decyzji zarzadzania łańcuchem dostaw w środowisku programowania w logice z ograniczeniami

Streszczenie: W artykule przedstawiono szczegóły implementacji koncepcji modelu wspomagania decyzji w łańcuchu dostaw. Do implementacji modelu wykorzystano środowisko CLP (Programowanie w logice z ograniczeniami) o nazwie Eclipse. Omówiono nowatorski sposób propagacji ograniczeń, który dla tej klasy problemów prowadzi do znacznej poprawy wydajności znajdowania rozwiązania. W artykule przedstawiono i scharakteryzowano najważniejsze predykaty, które służą do implementacji modelu. Jako ilustracje przyjętych założeń i rozwiązań zaprezentowano przykłady liczbowe.

Słowa kluczowe: zarządzanie łańcuchem dostaw, wspomaganie decyzji, programowanie w logice z ograniczeniami

Jarosław Wikarek, PhD

He graduated from the Faculty of Electrical Engineering and Automation Kielce University of Technology. He received a PhD at the Department of Automatic Control, Electronics and Computer Science, Silesian University of Technology. He is the author or co-author of over 90 articles. Main research interests include optimization and decision support for the processes of production, logistics and distribution using conventional MIP (Mixed Integer Programming) and declarative CLP (Constraint Logic Programming) programming environments.

e-mail: j.wikarek@tu.kielce.pl