THE ALGORITHM OF NEGOTIATION OF MULTI AGENTS FOR PLANNING IN GEOGRAPHICALLY DISTRIBUTED LOGISTIC SUPPLY CHAINS

Anatoly Levchenkov*, Mikhail Gorobetz**

Transport and Machinery Faculty*
Computer Science and Information Technology Faculty**
Riga Technical University
1, Kalku St.,
LV-1658 Riga, Latvia
levas@latnet.lv*
mgorobetz@gmail.com**

KEYWORDS
Multi-agent, schedule, negotiation, group decision making, logistics tasks

ABSTRACT

The purpose is to realize the negotiation algorithm for multi-agents[2] in stochastic condition using Monte-Carlo methods [3].
The negotiation is intended to support planning in geographically distributed supply chains and to create such schedule[1] that reduces usage of resources and coordinates actions of the logistics process participants.
It is proposed to test negotiation algorithm to create production schedule for the model of geographically distributed supply chain, where processors are plants, distribution centres, retailers and node points of the route, but jobs are transport units. The main condition is indefinite and stochastic character of some supply chain parameters such as delivery time and demand.
All the data for solution of the task have to be incorporated in a database, and appropriate programming languages and servers will be used for realization of algorithms using group decision support system.
The main advantage of using the negotiation in supply chain is organizing a centralized control system for all system, which follows large reduction of costs concerned with idle time and unsatisfied demand.

INTRODUCTION

This paper is based on our previous work [5]. This paper introduces a modification of negotiation algorithm in order to implement it in multi-agent supply chain systems.
Traditionally, marketing, distribution, planning, manufacturing, and the purchasing of organizations along the supply chain operate independently. Many manufacturing operations are designed to maximize throughput and lower costs with little consideration for the impact on inventory levels and distribution capabilities. Purchasing contracts are often negotiated with very little information beyond historical buying patterns. The result of these factors is that there is not a single, integrated plan for the organization. Clearly, there is a need for a mechanism through which these different functions can be integrated together. Supply chain management is a strategy through which such integration can be achieved. [4]
Usage multi-agents of high level architecture and negotiation in supply chain allow centralizing control of inventory, demand and orders.

There are two types of geographically distributed supply chain [4]:

- Supply chain has wide distribution network
- Big distances and long delivery time between locations

There are three main goals of each supply chain participant:

- Satisfy final demand
- Reduce costs
- Increase profit

Conflicts can be solved by negotiation are

- Time – impossible to accomplish multiple jobs simultaneously
- Transport - transport is not available
- Demand - not enough inventories or not enough time to complete order

MULTI-AGENT SUPPLY CHAIN STRUCTURE

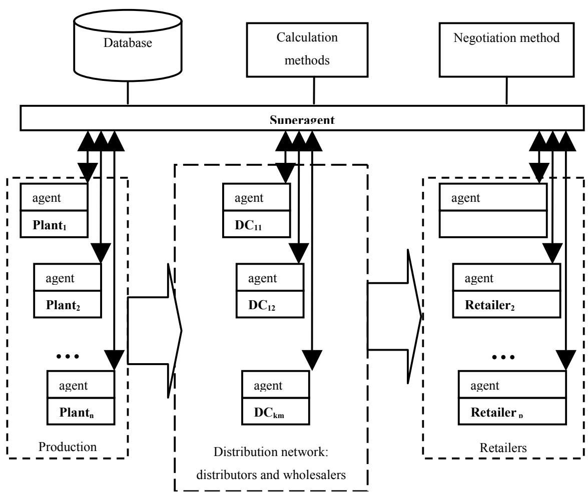


Figure 1. Structure.

There can be two kinds of objects in logistics supply process[1]. The processors are immovable objects – plants, distributors, shops etc. and the jobs that are

Proceedings 19th European Conference on Modelling and Simulation
Yuri Merkuryev, Richard Zobel, Eugène Kerckhoffs © ECMS, 2005
ISBN 1-84233-112-4 (Set) / ISBN 1-84233-113-2 (CD)

objects served by processors - for example, trucks. Each object in a supply chain has own multi-agent. Also there is a super agent – a multi-agent, which realizes negotiation and makes the schedule by using group decision support system.
A set of processors and a set of jobs are given as input data to solve the task. Each job consists of a number of operations and each operation is carried out by one processor. The duration of each operation and directive terms are randomly generated. General algorithm is proposed to solve the scheduling task to co-operate the work of multi-agents.
As it is seen from structure in fig.1 all information, calculations and negotiation process is concentrated in superagent.
The superagent organizes group decision making process. Other multi-agents are responsible only for their own objects, share operational information and accept or do not accept decisions and negotiation according to possible concessions.
The scheduling task is represented as a cooperative, non-coalition game, where all players are multi-agents and each is responsible only for its own object. They are interested in receiving maximal profit with minimal costs, not disturbing each other. The result of negotiation algorithm is the schedule that depends on stochastic processes.
All conflicts in supply chain are solved through negotiation process. Superagent plays a role of the arbitration judge, which nominates what players have to do proceeding from results of negotiation.

MATHEMATICAL MODEL FOR SUPERAGENT

Notation:
$\Re^+$ - nonnegative real number
Description – standardized verbal description for specified case.

Basic information in superagent's database

- Cost description
  $Description.Costs = \{$fixed order cost, running cost per hour, running cost per km, holding cost per unit, price Plant DC, price DC Retailer, price Retailer Client, Price Plant Retailer$\}$
- Transportation routes (from-to) – map of distribution network:
  - Graph – G(N,E), where
    - nodes – $N = \{n_1, n_2, ..., n_m\}; \quad n \in V$
    - edges – subset from Cartesian product of nodes
      $E = \{e_1, e_2, ..., e_n\};$
      $e_j = < n_a, n_b >, \quad j = \overline{1,n}, \quad a,b = \overline{1,m}, \quad a \neq b, \quad E \subseteq N \times N$
    - supply chain levels - $Lvls = \overline{0, LMax}, \quad LMax \in \aleph$
- A set of possible types of transport unit for all system: $TTypes = \{ttype_1, ttype_2, ..., ttype_t\}, \quad \text{ttype}_d \in Text$
  - speed - $speed[ttype_d] \in \Re^+, \quad d = \overline{1,t}$
  - costs - $\text{Costs}[\text{ttype}_1] = \{< Ctype_1, Cvalue_1 >, < Ctype_2, Cvalue_2 >, ...\},$
    $Ctype_s \in Description.Costs, \quad Cvalue_s \in \Re^+, \quad d = \overline{1,t}$
  - cargo maximum - $CapMax[ttype_d] \in \aleph, \quad d = \overline{1,t}$
- Information about all edges of graph:
  - distance - $dist[e_j] \quad j = \overline{1,n}$
  - possible transport type on current edge - $TTedge[e_j] = \{ttype_1, ttype_2, ...\}, \quad ttype_d \in TTypes$
  - probable delivery deviation for each transport type in percentage - $dev\%_j[ttype_d] \in \Re^+$

Necessary input information in supply chain for negotiation from agents:

- Information about all nodes in graph:
  - type of node - $type[n_i] \in Description, \quad i = \overline{1,\text{m}}$
  - level in supply chain (0 – production, max – retailer) - $level[n_i] \in Lvls$
  - supplier - $Sup[n_i] \in N, \quad i = \overline{1,\text{m}}$
  - capacity - $cap[n_i] \in \aleph, \quad i = \overline{1,\text{m}}$
  - costs – $\text{Costs}[\text{n}_i] = \{< Ctype_1, Cvalue_1 >, < Ctype_2, Cvalue_2 >, ...\},$
    $Ctype_s \in Description.Costs, \quad Cvalue_s \in \Re^+$
  - Inventory
    - current - $InvCur[n_i], \quad i = \overline{1,\text{m}}$
    - ordered - $InvOrd_l[n_i], \quad i = \overline{1,\text{m}}, \quad l \in \aleph$
    - directive terms - $dterm_l[n_i] = < beg, end >,$
      $i = \overline{1,\text{m}}, \quad l \in \aleph, \quad beg, end \in \Re^+$ (*date or time*)
  - order preparing time - $OprepTime[n_i] \in \Re^+$
  - compromises and additional charges – $C[n_i] = \{< type_1, values_1 >, < type_2, values_2 >, ...\},$
    $type_p \in Description,$
    $values_p = \{< \text{minvalue}_\text{p}, \text{costs}_{\text{p}1} >, < \text{maxvalue}_\text{p}, \text{costs}_{\text{p}1} >, ...\},$
    $p \in \aleph, \quad r = \overline{1,v}, \quad \text{minvalue}_\text{p}, \text{maxvalue}_\text{p} \in \Re, \quad \text{costs}_\text{p} \in \Re^+$
  - forecasted demand - $demand[n_i] = \{< time_1, amount_1 >, < time_2, amount_2 >, ..., < time_k, amount_k >, ...\} \quad i = \overline{1,m}$
  - schedule
    $sched[n_i] = \{< time_1, amount_1 >, < time_2, amount_2 >, ..., < time_k, amount_k >, ...\}$
    $time \in \Re, \quad amount \in \aleph, \quad k \in \aleph, \quad i = \overline{1,m}$
- Additional information from production nodes:
  - maximal production amount - $prodMax[n_i] \in \aleph, \quad i = \overline{1,\text{m}}$
  - current production amount - $prodCur[n_i] \in \aleph, \quad i = \overline{1,\text{m}}$
- Information about all movable objects (transport):
  - a set of transport units - $TR = \{tr_1, tr_2, ..., tr_v\}$
    - unit type - $type[tr_r] \in TTypes, \quad r = \overline{1,v}$
    - assigned to - $tassign[tr_r] = n_i, \quad r = \overline{1,v} \quad n_i \in N$
    - origin node - $origin[tr_r] \in N, \quad r = \overline{1,v}$

- destination node - $dest[tr_r] \in N, \quad r = \overline{1,v}$
- current location at the moment of negotiation - $loc[tr_r] = < place, dist_1, dist_2 >,$ $place \in E \cup N, \quad dist_1, dist_2 \in \Re^+$
- current cargo - $Cur[tr_r] \in \aleph, \quad r = \overline{1,v}$
- directive term - $dterm[tr_r] = < beg, end >,$ $r = \overline{1,v}, \quad beg, end \in \Re^+$ (*date or time*)
- due date/time – $dtime[tr_r] \in \Re^+$
- compromises and additional charges – $C[tr_r] = \{< type_1, values_1 >, < type_2, values_2 >, ...\},$ $type_p \in Description,$ $values_p = \{< \text{minvalue}_p, \text{costs}_{p1} >, < \text{maxvalue}_p, \text{costs}_{p1} >, ...\},$ $p \in \aleph, \quad r = \overline{1,v}, \quad \text{minvalue}_p, \text{maxvalue}_p \in \Re, \quad \text{costs}_p \in \Re^+$
- schedule - $sched[tr_r] = \{< time_1, amount_1 >, < time_2, amount_2 >, ..., < time_k, amount_k >, ...\}$ $time \in \Re, \quad amount \in \aleph, \quad k \in \aleph, \quad r = \overline{1,v}$

- Information about delivery routes in graph:
  - delivery route - $Routes(Sup[n_i], n_i) = \{route_1, ..., route_y\}$ $route_k(Sup[n_i], n_i) = \{edge_1, ..., edge_z\},$ $k = \overline{1,y} \quad z \in \overline{1,n} \quad edge_j \in E,$ $edge_1 = < Sup[n_i], n_a >, \quad edge_z = < n_a, n_i >$
    - distance - $dist[route_k] = \sum_{j=1}^{z} dist[edge_j],$ $\forall route_k \in Routes(Sup[n_i], n_i)$
    - possible transport types for each route $TTroute_k(Sup[n_i], n_i) = \bigcap_{j=1}^{z} TTedge(edge_j),$ $k = \overline{1,y}, \quad i = \overline{1,m}$

Necessary calculations for supply chain model

- running duration and costs for each edge transport:
  - running mean value - $dmean_j[ttype_d] = dist[e_j] / speed[ttype_d]$ $j = \overline{1,n} \quad \forall ttype_d \in TTedge[e_j]$
  - running standard deviation - $ddev_j[ttype_d] = dmean_j[ttype_d] * dev\%_j[ttype_d]$ $j = \overline{1,n} \quad \forall ttype_d \in TTedge[e_j]$
  - maximal running time $MaxDTime_d[e_j] = dmean_j[ttype_d] + ddev_j[ttype_d]$ $j = \overline{1,n} \quad \forall ttype_d \in TTedge[e_j]$
  - maximal running costs - $MaxRunCosts_d[e_j] = MaxDTime_j[ttype_d] \times \times RunCostsPerHour[ttype_d]$ $j = \overline{1,n} \quad \forall ttype_d \in TTedge[e_j]$
- running duration and costs for each route transport:
  - maximal running time $MaxDTime_d[route_k] = \sum_{j=1}^{z} MaxDTime_d[e_j]$ $\forall route_k \in Routes(Sup[n_i], n_i), \quad \forall ttype_d \in TTroute_k(Sup[n_i], n_i)$ $k = \overline{1,z} \quad i = \overline{1,m}$
  - maximal running costs - $MaxRunCosts_d[route_k] = MaxDTime_d[route_k] \times$ $\times \text{running cost per hour}[\text{ttype}_d]$ $\forall ttype_d \in TTroute_k(Sup[n_i], n_i) \quad k = \overline{1,z} \quad i = \overline{1,m}$

PRODUCTION SCHEDULING ALGORITHM

A production schedule is a result of multi agent collaboration which specify production amount for the specific date.

Step 0. Level selection for analysis

- select level of supply chain $L = LMax$

Step 1. Demand analysis using Monte-Carlo methods

1.1. superagent gets subset of retailers $N' \subset N, \quad where\ level[n_i] = L$

1.2. superagent gets demand of each retailer $demand[n_i], \quad \forall n_i \in N'$

1.3. calculating average demand per day for each retailer: $\mu[n_i] = \frac{\sum_{a=1}^{k} demand[n_i].amount_a}{k}$

1.4. calculating standard deviation of demand for each retailer:

$$\sigma[n_i] = \sqrt{\frac{\sum_{a=1}^{k-1} (demand[n_i].amount_a - demand[n_i].amount_{a+1})^2}{k}}$$

1.5. calculating safety stock for each retailer: $Sstock[n_i] = \sigma[n_i] * z, \quad z = 1{,}65 \ (for\ service\ level\ 95\%)$

Step 2. Optimal order quantity planning for each retailer

2.1. superagent gets supplier for each retailer: $Sup[n_i], \quad \forall n_i \in N'$

2.2. get possible transport types for each supply: $TSup[n_i] = \{TTroute_1(Sup[n_i], n_i), ..., TTroute_k(Sup[n_i], n_i)\},$ $k = \overline{1,y} \quad \forall n_i \in RET$

2.3. calculate EOQ for each retailer and transport:

$$EOQ_d[n_i] = \sqrt{\frac{2 \cdot (\mu[n_i] + \sigma[n_i]) \cdot \text{fixed order cost}[\text{ttype}_d]}{\text{holding cost per unit}}}$$

$ttype_d \in TSup[n_i], \quad n_i \in N'$

2.4. EOQ correction according to transport maximal capacity:

$$EOQ_d[n_i] = \begin{cases} EOQ_d[n_i], & \text{if } EOQ_d[n_i] < CapMax[ttype_d] \\ CapMax[ttype_d], & \text{if } EOQ_d[n_i] > CapMax[ttype_d] \end{cases}$$

$ttype_d \in TSup[n_i], \quad n_i \in N'$

2.5. Average interval between order calculation: $Int_d[n_i] = EOQ_d[n_i] / \mu[n_i]$

Step 3. Order time and transport type calculation

3.0. Start with current date: $date = 0$

3.1. Getting current inventory and capacity.

$Inv_i[date].start = InvCur[n_i] + Order_i[date].amount,$

$cap[n_i] \quad n_i \in N'$

$If\ Inv_i[date].start > cap[n_i] \quad then \quad initiate\ negotiation$

3.2. Calculating inventory status

$Inv_i[date].end = Inv_i[date].start + Order_i[date].amount -$
$- demand[n_i].amount_k$

$Inv_i[date].start = Inv_i[date-1].end + Order_i[date].amount$

$\forall n_i \in N'$

3.3. Calculating order delivery due date critical delivery date:

$if\ Inv_i[date] - Sstock[n_i] < 0 \quad then \quad ODate_{nr} = date$
$\qquad else \quad date = date + 1, repeat\ 3.2.$

$if\ Inv_i[date] < 0 \quad then \quad OCritDate_{nr} = date$
$\qquad else \quad date = date + 1, repeat\ 3.2.$

3.4. Getting maximal running time and running costs

$MaxDTime_d[route_k], MaxRunCosts_d[route_k],$

$\forall route_k \in Routes(Sup[n_i], n_i), \quad \forall n_i \in N'$

3.5. Evaluate transport type by order cost:

$TC_d[n_i] = \text{fixed order cost}[\text{ttype}_d] + 2MaxRunCosts_d[route_k]$

$eval_d[n_i] = TC_d[n_i] / Int_d[n_i]$

$\forall route_k \in Routes(Sup[n_i], n_i), \quad \forall n_i \in N'$

3.6. Select the best transport type

$DTType(Sup[n_i], n_i) = d, \quad if\ eval_d[n_i] = \min(\forall eval[n_i])$

$ttype_d \in TSup[n_i], \quad \forall n_i \in N'$

3.7. Calculating order amount:

$Order_i[date].amount = EOQ_{DTType(Sup[n_i],n_i)}[n_i] + neg.amount[n_i].corr$

3.8. Create schedule records (amount, order preparing, departure, delivery, return)

$record[n_i] =< amount, prep, depart, due, return >$

$amount = Order_i[date].amount + negotiation[n_i].demand.corr,$

$due = ODate_{nr}[n_i] + negotiation[n_i].sched.corr$

$depart = due - MaxDTime_d(Sup[n_i], n_i)$

$prep = depart - OprepTime[n_i]$

$return = due + MaxDTime_d(Sup[n_i], n_i)$

3.9. Check records.
If $record[n_i].prep \geq 0$ then 3.9.
else - ***initiate negotiation procedure***

3.10. Check availability for transport unit

$$tr[n_i] \in TR$$

3.11. If transport is available then assign records to schedules

$sched_{nr}(Sup[n_i]) =< Job_{nr}, n_i, tr[n_i], record[n_i].prep,$
$record[n_i].departure, record[n_i].amount >$

$sched_{nr}[tr] =< Job_{nr}, Sup[n_i], n_1, record[n_i].depart, record[n_i].due,$
$OCritDate_{nr}[n_i], record[n_i].return, record[n_i].amount >$

else – ***initiate negotiation procedure***

3.12. Return 3.1.

3.13. L=L-1. Return Step 1.

NEGOTIATION PROCEDURE

Step 1. Define conflict types in a problem:

$\text{conflicts} \subseteq \{\text{time, demand, transport}\}$ time,

Step 2. Offer possible strategies to solve a problem according to conflicts

$$strategies[conflict_i] \subseteq \{\text{schedule correction, another transport unit selection, another transport type selection}\}$$

$conflict_i \in \text{conflict}$

Step 3. Imitate implementation of

$\exists strategy_j \in strategies$ using solution methods [5].

Step 4. Evaluate charges of using strategy.

Step 5. Choose strategy with minimal costs

NUMERICAL EXAMPLE

For numerical example, abstract supply chain is taken. It is simplified just to show principles of negotiation between multi agents. So, there is one type of product, one plant and one level distribution network with 2 DC and 4 retailers.

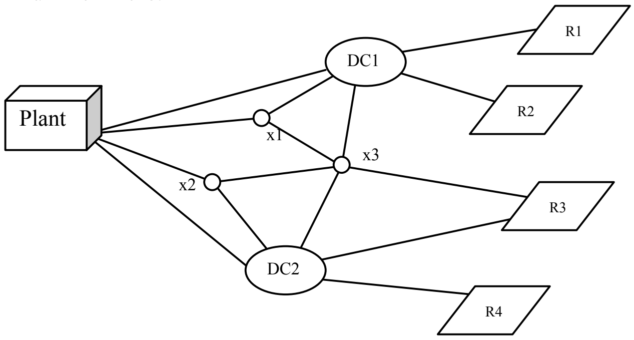


Figure 2. Numerical example supply network

INPUT DATA

A set of nodes N (table 1), a set of possible transport types TTypes (table 2), a set of edges E and prenegotiation calculation (table 3), a set of Routes (table 4), transport units TR (table 5) are given.

Table 1: Nodes

Node	Type	Level
Plant	Plant	0
DC1	DC	1
DC2	DC	1
R1	Retailer	2
R2	Retailer	2
R3	Retailer	2
R4	Retailer	2
x1	Crossroad	NULL
x2	Crossroad	NULL
x3	Crossroad	NULL

Table 2: Transport types

Type	Speed (km/h)	FC	Running cost (CU/h)	Order preparing	Capacity
Ship	30	2300	6	24	500
Truck	50	800	2	24	50
Train	60	3000	9	24	500
Aircraft	400	7200	24	24	150

Table 3: Edges

Edge	From	To	Distance	Transport type	Probable delivery deviation	dmean (h)	ddev (h)	MaxDtime (h)	MaxRun Cost (CU)
e1	Plant	x1	2500	Truck	0,1	50	5	55	110
e1	Plant	x1	2500	Train	0,1	42	5	47	423
e2	Plant	DC1	3500	Ship	0,1	117	12	129	774
e3	Plant	x2	3000	Train	0,1	50	5	55	495
e4	Plant	DC2	3900	Aircraft	0,1	10	1	11	264
e5	x1	DC1	2000	Truck	0,1	40	4	44	88
e5	x1	DC1	2000	Train	0,1	34	4	38	342
e6	x1	x3	1800	Train	0,1	30	3	33	297
e7	x2	x3	2000	Train	0,1	34	4	38	342
e8	x2	DC2	1800	Train	0,1	30	3	33	297
e9	DC1	R1	1600	Truck	0,1	32	4	36	72
e9	DC1	R1	1600	Train	0,1	27	3	30	270
e10	DC1	R2	1500	Truck	0,1	30	3	33	66
e10	DC1	R2	1500	Train	0,1	25	3	28	252
e11	DC1	x3	700	Truck	0,1	14	2	16	32
e12	DC2	x3	800	Truck	0,1	16	2	18	36
e13	DC2	R3	1300	Truck	0,1	26	3	29	58
e14	DC2	R4	1200	Truck	0,1	24	3	27	54
e15	x3	R3	2000	Train	0,1	34	4	38	342

Table 4: Routes

From	To	Route	Edges	Distance	Transport type	MaxDtime (h)	MaxRun Cost (CU)
Plant	DC1	route1	e1,e5	4500	Truck	99	198
					Train	83	747
		route2	e3	3500	Ship	129	774
Plant	DC2	route1	e3,e8	4800	Train	88	792
		route2	e4	3900	Aircraft	11	264
DC1	R1	route1	e9	1600	Truck	36	72
					Train	30	270
DC1	R2	route1	e10	1500	Truck	33	66
					Train	28	252
DC2	R3	route1	e13	1300	Truck	29	58
DC2	R4	route1	e14	1200	Truck	27	54
DC1	DC2	route1	e11,e12	1500	Truck	33	66
Plant	R3	route1	e1,e6,e15	6800	Train	126	1134
		route2	e3,e7,e15	7000	Train	129	1161

Table 5: Transport units

Unit	Type	Current location		
		Edge/Node	Dist1	Dist2
Plane1	Aircraft	Plant	0	0
Ship1	Ship	Plant	0	0
Truck1	Truck	Plant	0	0
Truck2	Truck	Plant	0	0
Truck3	Truck	DC1	0	0
Truck4	Truck	DC1	0	0
Truck5	Truck	DC2	0	0
Truck6	Truck	DC2	0	0
Truck7	Truck	DC2	0	0
Train1	Train	Plant	0	0
Train2	Train	Plant	0	0
Train3	Train	DC1	0	0

GETTING PRODUCTION SCHEDULE USING MULTI-AGENTS AND NEGOTIATION

Step 0. L = 2

Step 1.

1.1. N'={R1, R2, R3, R4, R5, R6}

1.2. Getting demand from each retailer.

Day	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
Begin h	*24*	*48*	*72*	*96*	*120*	*144*	*168*	*192*
End h	*48*	*72*	*96*	*120*	*144*	*168*	*192*	*216*
R1	15	12	10	8	19	14	16	11
R2	13	14	10	11	17	19	22	12
R3	10	12	11	8	15	13	14	11
R4	8	7	6	5	9	4	3	6
Day	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*
Begin h	*216*	*240*	*264*	*288*	*312*	*336*	*360*	*384*
End h	*240*	*264*	*288*	*312*	*336*	*360*	*384*	*408*
R1	21	18	15	10	17	18	16	15
R2	18	16	11	14	15	25	23	13
R3	12	13	10	9	10	11	9	10
R4	7	10	9	8	7	11	13	8

1.3.,1.4.,1.5. Mean value, deviation of demand and safety stock calculation (table 7)

Customer	μ	σ	Safety stock
R1	14,69	3,65	7
R2	15,81	4,53	8
R3	11,13	1,93	4
R4	7,56	2,56	5

Step 2.

2.1., 2.2. Getting suppliers and transport types

Customer	Supplier	Transport
R1	DC1	train, truck
R2	DC1	train, truck
R3	DC2	truck
R4	DC2	truck

2.3., 2.4., 2.5. EOQ calculation, correction and frequency

Customer	Supplier	Transport	Max capacity	EOQ	EOQ corr.	interval
R1	DC1	truck	50	55	50	3
		train	500	105	105	7
R2	DC1	truck	50	58	50	3
		train	500	111	111	7
R3	DC2	truck	50	46	46	4
R4	DC2	truck	50	41	41	5

Step 3.

3.0. Day = 1.

Iteration 1

3.1. Getting inventory

Customer	Current	Maximal	Check
R1	45	120	OK
R2	50	150	OK
R3	30	90	OK
R4	25	75	OK

3.2.,3.3. Calculating inventory status due date and critical due date:

	Day	**1**	**2**	**3**
	Start h	**24**	**48**	**72**
	End h	**48**	**72**	**96**
R1	Start	45	30	18
	End	30	18	8
	Order date	FALSE	FALSE	FALSE
	Critical date	FALSE	FALSE	FALSE
R2	Start	50	37	23
	End	37	23	13
	Order date	FALSE	FALSE	FALSE
	Critical date	FALSE	FALSE	FALSE
R3	Start	30	20	8
	End	20	8	-3
	Order date	**FALSE**	**FALSE**	**72**
	Critical date	**FALSE**	**FALSE**	**72**
R4	Start	25	17	10
	End	17	10	4
	Order date	**FALSE**	**FALSE**	**72**
	Critical date	FALSE	FALSE	FALSE

3.4., 3.5, 3.6. Getting maximal running time and running costs, evaluate transport type and selecting the best for each retailer:

From	**To**	**Transport type**	**Fixed order cost (CU)**	**Max Dtime (h)**	**Max Run Cost (CU)**	**TC (CU)**	**interval**	**Eval**	**Select**
DC2	**R3**	**Truck**	**800**	**29**	**58**	**916**	**4**	**229**	**OK**
DC2	**R4**	**Truck**	**800**	**27**	**54**	**908**	**5**	**182**	**OK**

3.7. Order amounts:

From	**To**	**Negotiated correction**	**Order amount**
DC2	R3	NULL	46
DC2	R4	NULL	41

3.8., 3.9. Creating and checking schedule records

Job	To	Amount	Order preparing	departure	due	return	negotiation correction	Record check
Job11	R3	46	19	43	72	101	0	OK
Job12	R4	41	21	45	72	99	0	OK

3.10. Select available transport:

Supplier	Name	Type	Available	Return	Select
DC2	Truck5	Truck	TRUE	NULL	OK
	Truck6	Truck	TRUE	NULL	OK
	Truck7	Truck	TRUE	NULL	

3.11. Assigning record to schedules:

Table 6: DC2 schedule

Job	Dest	Unit	Prep	Depart	Amount
Job1	R3	Truck5	19(0)	43(1)	46
Job2	R4	Truck5	21(0)	45(1)	41

Table 7: Truck5 schedule

Job	Orig	Dest	Amount	Depart	Due	Return
Job1	DC2	R3	46	43(1)	72(3)	101(4)

Table 8: Truck6 schedule:

Job	Orig	Dest	Amount	Depart	Due	Return
Job2	DC2	R4	41	45(1)	72(3)	99(4)

3.12. Return 3.1.

Iteration 2

3.1. Inventory (day 3)

Customer	Current	Maximal	Check
R1	45	120	OK
R2	50	150	OK
R3	54	90	OK
R4	51	75	OK

3.2. etc…

3.11. Result of iteration

Table 9: DC1 schedule:

Job	Dest	Unit	Prep	Depart	Amount
Job1	R1	Truck3	36(1)	60(2)	50
Job2	R2	Truck4	39(1)	63(2)	50

Table 10: Truck3 schedule:

Job	Orig	Dest	Amount	Depart	Due	Return
Job1	DC1	R1	50	60(2)	96(4)	132(5)

Table 11: Truck4 schedule:

Job	Orig	Dest	Amount	Depart	Due	Return
Job2	DC1	R4	50	63(2)	96(4)	129(5)

Iterations 3 - 7

Repeat previous steps without conflicts. Result schedules are following.

Table 12: DC1 schedule

Job	Dest	Unit	Prep	Depart	Amount
Job11	R1	Truck3	36	60	50
Job12	R2	Truck4	39	63	50
Job13	R1	Truck4	108	132	50
Job14	R2	Truck3	111	135	50
Job15	R1	Truck3	180	204	50
Job16	R2	Truck4	183	207	50
Job17	R1	Truck4	276	300	50
Job18	R2	Truck3	279	303	50

Table 13: DC2 schedule

Job	Dest	Unit	Prep	Depart	Amount
Job21	R3	Truck5	19	43	46
Job22	R4	Truck6	21	45	41
Job23	R3	Truck6	115	139	46
Job24	R3	Truck5	187	211	46
Job25	R4	Truck6	189	213	41
Job26	R3	Truck6	307	331	46
Job27	R4	Truck5	309	333	41

Table 14: Truck3 schedule

job	orig	dest	amount	departure	due	return
job11	DC1	R1	50	60	96	132
job14	DC1	R2	50	135	168	201
job15	DC1	R1	50	204	240	276
job18	DC1	R2	50	303	336	369

Table 15: Truck4 schedule

job	orig	dest	amount	departure	due	return
job12	DC1	R2	50	63	96	129
job13	DC1	R1	50	132	168	204
job16	DC1	R2	50	207	240	273
job17	DC1	R1	50	300	336	372

Table 16: Truck5 schedule

job	orig	dest	amount	departure	due	return
job21	DC2	R3	46	43	72	101
job24	DC2	R3	46	211	240	269
job27	DC2	R4	41	333	360	387

Table 17: Truck6 schedule

job	orig	dest	amount	departure	due	return
job22	DC2	R4	41	45	72	99
job23	DC2	R3	46	139	168	197
job25	DC2	R4	41	213	240	267
job26	DC2	R3	46	331	360	389

Iteration 7

There are fragments from iteration

3.2., 3.3. Finding next due date:

	Day	**16**
	Start h	**384**
	End h	**408**
R1	…	
R2	Start	10
	End	-3
	Order date	**384**
	Critical date	**384**
R3	…	
R4	…	

3.4., 3.5, 3.6. Transport type evaluation

From	To	Transport type	Fixed order cost (CU)	Max Dtime (h)	Max Run Cost (CU)	TC (CU)	interval	Eval	Select
DC1	**R2**	**Truck**	**800**	**33**	**72**	**944**	**3**	**290**	**OK**
DC1	R2	Train	3000	28	270	3540	7	467	

3.7., 3.8, 3.9. Defining amount of order, creating record and check.

Job	To	Amount	Order preparing	departure	due	return	negotiation correction	Record check
Job19	R2	50	327	351	384	417	0	OK

3.10. Select available transport:

Supplier	Name	Type	Available	Return	Select
DC1	Truck3	Truck	FALSE	369(15)	
	Truck4	Truck	FALSE	372(15)	

3.11. There is no transport available – it is necessary to initiate negotiation process.

NEGOTIATION PROCEDURE

1. conflict detection
   - Not available transport of the best type
   - Time conflict
2. possible solutions for this conflicts
   - Check availability for all possible transport type
   - Existing schedule correction
3. evaluate solutions

3.1. Another transport strategy

- Checking all transport units of DC1

Supplier	Name	Type	Available	Return	Select
DC1	Truck3	Truck	FALSE	369(15)	
	Truck4	Truck	FALSE	372(15)	
	Train3	Train	TRUE	NULL	

- Evaluate costs of using this strategy

From	To	Transport type	Fixed order cost (CU)	Max Dtime (h)	Max Run Cost (CU)	TC (CU)
DC1	R2	Train	3000	28	270	3540

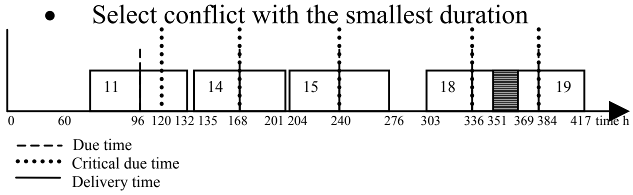
- Strategy 3.1. charges: **TC = 3540**

3.2. Schedule correction strategy – using negotiation algorithm for time conflict solution [5].

- Creating negotiation set

Conflict	Time crossing	Duration
Truck3, job18, job19	**351-369**	**18**
Truck4, job17, job19	351-372	21

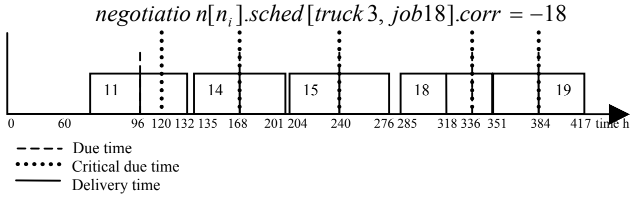
- Select conflict with the smallest duration



- Free space for job18, job19 t1 and t2

Truck	Interval	Free time
job19	384 – 384	0
job18	303 – 274	29

- Interval for moving – 369 - 351=18
  Free time = 29
- Result:
  $negotiatio\,n[n_i].sched[truck3, job18].corr = -18$



- Strategy 3.2. charges: **additional holding cost = time*unit holding cost /2= 18*100/2 = 900**

Negotiation procedure result – strategy 3.2:
$negotiatio\,n[n_i].sched[truck3, job18].corr = -18$

Result changed schedules are for DC1(Table 18) and for Truck3 (table 19).

Table 18: DC1 schedule

Job	Dest	Unit	Prep	Depart	Amount
Job11	R1	Truck3	36	60	50
Job12	R2	Truck4	39	63	50
Job13	R1	Truck4	108	132	50
Job14	R2	Truck3	111	135	50
Job15	R1	Truck3	180	204	50
Job16	R2	Truck4	183	207	50
Job17	R1	Truck4	276	300	50
Job18	R2	Truck3	261	285	50
Job19	R2	Truck3	327	351	50

Table 19: Truck3 schedule

job	orig	dest	amount	departure	due	return
job11	DC1	R1	50	60	96	132
job14	DC1	R2	50	135	168	201
job15	DC1	R1	50	204	240	276
job18	DC1	R2	50	285	318	351
job19	DC1	R2	50	351	384	417

Calculated schedules for DC1 and DC2 are demand for next level of supply chain. This way, algorithm will repeat until the result – convenient schedule for production, distribution network participants and all transport units.

CONCLUSIONS AND FURTHER RESEARCH

Numerical example shows functionality of the negotiation algorithm. In proposed mathematical model all types of all parameters are given. That is why developed algorithm can be easy realized in any programming language and DBMS.

Our aim for future is an expert system for supply chain management based on multi-agent negotiation algorithm intended to fast tasks solutions in conditions approached to real. For near future it is planned to focus attention on negotiation strategies and evaluation of these strategies.

REFERENCES

1. Conway R. Theory of Scheduling. Dover Publications. 2003.
2. Ferber J. Multi-agent Systems: An Introduction to Distributed Artificial Intelligence. Addison Wesley. 1999
3. Jeremy F. Shapiro. Modeling the Supply Chain. Wadsworth group, Thomson Learning inc., 2001
4. F. Frank Chen. Process Modeling, Performance Analysis and Configuration Simulation in Integrated Supply Chain Network Design, Blacksburg, Virginia, 2001
5. Anatoly Levchenkov, Mikhail Gorobetz. The Algorithm of Negotiation for Software Agents for the Open Conveyor Schedule in Logistics Tasks. In: HMS2003, Riga, Latvia, 142-148.

BIOGRAPHY

ANATOLY LEVCHENKOV is Associate Professor of the Logistics and Telematics at Riga Technical University from 2001. Since 1997 he has been Associate Professor at the Transport Railways Institute and from 1992 to 1997 was Head of the Riga Technical University's Department of Decision Support system. He studied Electrical Engineering in Automation and Telemechanics from 1964 to 1969 and from 1969 to 1992 he was a reader and associate professor at the Department of Automated Control and Decision Support Systems at Riga Technical University.

MIKHAIL GOROBETZ is graduated Bachelor of Computer Science and Computer Management at Riga Technical University Computer Science and Information Technology Faculty Information Technology Institute in 2003. He started studies at Riga Technical University in 2000. At present he studies for Master of Computer Science grade at Riga Technical University. He participated in Riga Technical University Annual Student Scientific Conference 2003.