Multi-agent Supply Planning Methods Allowing the Real-Time Performance of the Supply Network Management Systems

Alexander Tsarev(✉)

ICCS RAS, Samara, Russia
a@tsarev.info

Abstract. The paper deals with the questions of how to develop the automated planning systems that are fast enough to be used in real-time management of supply networks, considering the manual plan corrections by the users. Several practical situations and planning system use cases are considered. The paper proposes several methods that allow the increase of the data processing speed in practical cases. The methods include parallel data processing, dynamic control of the solutions space depth search, self-regulation of the system behavior based of the specifics of the data processed.

Keywords: Distributed planning · Self-regulation · Parallel processing · Coordination · Performance · Real-time scheduling · Real-time management · Multi-agent planning · Supply network · Supply chain

1 Introduction

Effective management always required a permanent re-evaluation of the available options, planning, modelling and making in-time decisions based on this analytical work. Finally, the success depends on how good the analysis is, and how fast it is.

Just making a good optimization is not enough anymore. Growing complexity and dynamics of modern business demand new paradigms in resource management [1, 4]. New approach to increase efficiency of business is associated today with the real-time economy, which requires adaptive reaction to events, ongoing decision making on resource planning, optimization and communication results with decision makers. This is especially important in supply network management, as it is a highly competitive environment with many participants trying to be fast and efficient all the time.

The main feature of real-time scheduling and optimization methods is the ability to produce a result fast enough to cope with the changes in the input data (events). It means that the model or the plan should be re-evaluated, re-built before the next significant change comes (new order, order cancellation, resource availability, etc.).

Multi-agent technology is considered to be a design methodology and framework to support distributed problem solving methods in real-time scheduling and optimization of resources [5, 6].

Figure 1 illustrates the difference in actuality of scheduling results (how well they reflect reality) in the changing environment. Having frequent data updates, it becomes

V. Mařík et al. (Eds.): HoloMAS 2019, LNAI 11710, pp. 81–95, 2019.
https://doi.org/10.1007/978-3-030-27878-6_7

more important to process them faster to get a valid result (green line). Otherwise, one can use a lengthy processing to get an optimal result (yellow/red line), but this result does not consider the last changes. Then, we are forced to always base your decisions on an optimal, but outdated picture.

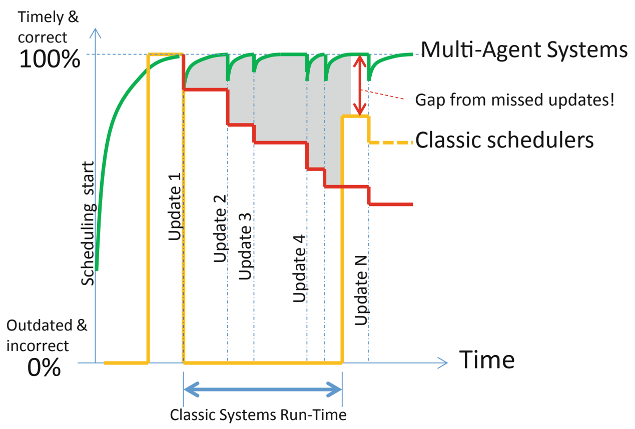

Fig. 1. Real-time adaptive scheduling effects. (Color figure online)

The quality and efficiency of decision making in resource scheduling and optimization process can be influenced by the number of factors: the intensity of events flow, the number and current state of resources, individual specifics of orders and resources, time interval between the events and processing time for events, productivity of resources and many others.

A big challenge is to ensure that certain quality of scheduling results is achieved in a short time after the event to make it possible to finish the processing before the next event and to always have a valid schedule needed for decision-making.

Although the multi-agent approach allows fast reaction to events, it still requires methods to ensure the processing time is short enough to produce results before the next event.

2 Task Definition and Practical Cases

In order to effectively manage supply networks, you need, first of all, to build a plan for the operation of the entire network and keep it up to date in a constantly changing environment. Efficiency at the same time depends on two main parameters: the quality of the plan (how high indicators are achieved when it is executed) and the speed of its re-building in response to events. These two parameters are in conflict, so the planning method should ensure the best quality of the plan, achievable in a characteristic time between successive events entering the system.

The quality of a plan when managing supply chains is usually determined by how much demand closes (which is directly related to income) and how much money is required to spend on executing this plan (expense). In practice, the expenditure part may include not only real costs, but also conditional, associated with administration, reputational risks, etc. But it is proposed to use a linear combination of revenue and all consumable parts with coefficients that regulate the importance of each component as the basic overall objective function of the network.

$$Q = k^R R - C$$

R - total satisfaction of needs $R = \sum k_i^r r_i$, r_i - satisfaction of an individual demand (ratio of the satisfied quantity to the need), k_i^r - need importance factor (assigned automatically, the more urgent the need, the higher the coefficient).

The cost (cost of the plan) may be different for each case of the supply network. For example, for the task of material support, taking into account several projects in the supply network:

$$C = k^{Co}C^o + k^{Cf}C^f + k^{Cp}C^p + k^{Ca}C^a + k^{Ct}C^t$$

C^o - conditional administrative costs of transfer between owners.
C^f - conditional administrative costs of transfer from free stocks.
C^p - conditional administrative costs for transfer between projects.
C^a - the cost of using analog.
C^t - the cost of transportation in the logistics network.
$k^R, k^{Co}, k^{Cf}, k^{Cp}, k^{Ca}, k^{Ct}$ are weight coefficients.

The analysis showed that for most tasks it is convenient to apply a linear function of dependence of satisfaction coefficients on their remoteness in time:

$$k_s^{d\,default}(\Delta t) = \begin{cases} k_s^{d\,max}, \Delta t \leq t_s^{min} \\ k_s^{dmax} - \frac{\left(k_s^{d\,max} - k_s^{d\,min}\right)\left(\Delta t - t_s^{min}\right)}{t_s^{max} - t_s^{min}}, t_s^{min} < \Delta t < t_s^{max} \\ k_s^{d\,min}, \Delta t \geq t_s^{max} \end{cases}$$

where t_s^{min} is the end of the "red" zone (the rest of the time, in which, as a rule, nothing can be fixed), t_s^{max} is the beginning of the "green" zone, where plans can change more than once and there is not much point in planning so far, $k_s^{d\,min}$ - the coefficient of importance of orders at the node s in the "green" zone, $k_s^{d\,max}$ - the coefficient of importance of orders at the node s in the "red" zone.

The method itself should not impose significant restrictions on the form of the objective function.

The solution in the system should be achieved through distributed agent negotiations, which are guided by individual objective functions. In order to better match the solution achieved as a balance of interests of individual agents, the overall objective function of the supply network, the network agent (headquarters agent) should be able to provide targeted feedback to the agents' needs and resources.

In various situations, in the process of the agents, and for the weighted evaluation of the planning results, other performance indicators may be used.

We used the described model to build the systems for mass production and distribution (LEGO, Coca-Cola), energy production and distribution, railway cars supply, construction supply.

For LEGO the main focus was to dynamically reschedule the replenishment of the products in the network of the brand retail stores in USA based on the sales forecast, current stocks and supply network limitations. The supply network included 50 retail stores, 3 distribution centers, about 1000 products.

For Coca-Cola in Germany we built the system to automatically schedule the order execution in real time as the orders are placed by the customers. The goal was to improve the order fulfillment and reduce the transport costs, especially in the peak seasons. The task included production and transportation scheduling in the distributed supply network of 300 DCs and 8 factories.

Energy production and distribution model included up to 100 energy production points including big coal plants as well as small solar and wind production and energy accumulation. It allowed dynamical rescheduling of the production at different plants and redistribution of flows in response to the changes in consumption forecasts, production power limitations and price.

Railway cars supply case was about redistribution of about 30000 cars of different types in the railway network of about 1000 production sites, ports, customers to ensure that they have enough cars to deliver their products.

Construction supply case deals with several interconnected networks related to big construction sites in oil industry. It focuses on in-time delivery of all components needed for construction in difficult weather conditions with many unpredicted transportation limitations, multi-modal delivery using trucks, railways, ships and helicopters. The supply scheduling includes the planning of transportation resources as well as loading-unloading equipment and personnel.

3 Proposed Methods

3.1 Distributed Parallel Processing

First of all, the most obvious approach we used to improve the performance is to do things in parallel. We see that for practical application of the planning software in open market involving several independent companies, it is important to let the companies have different policies, isolated data processing, and standardized API for interaction between different planning systems.

At the same time, it is crucial for planning scalability to perform the planning in a distributed, but coordinated way, to reduce processing time, but keep the plans synchronized.

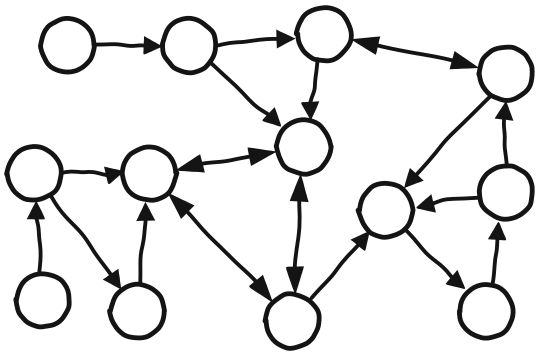

Fig. 2. Abstract example of the supply network.

Based on the multi-agent approach, it is possible to solve the problem in a distributed way, running software agents on the company-related hardware instead of the central server. This approach is very suitable for solving tasks in open environments. However, it is important to consider that the structure of the energy grid can change dynamically (customers and suppliers can connect and leave the network).

To illustrate the proposed approach to the distributed coordinated planning, let's consider the abstract network example shown in Fig. 2. Each node (site) in the network may supply or consume, and the channels are used to deliver the energy to other sites. Currently, the developed software prototype uses a separate agent for each site, and the agents communicate and negotiate in the single environment.

The internal constraints of each site in the network are hidden from external actors and affect the result via the restrictions and costs calculated and presented to neighbor sites in the network during negotiations.

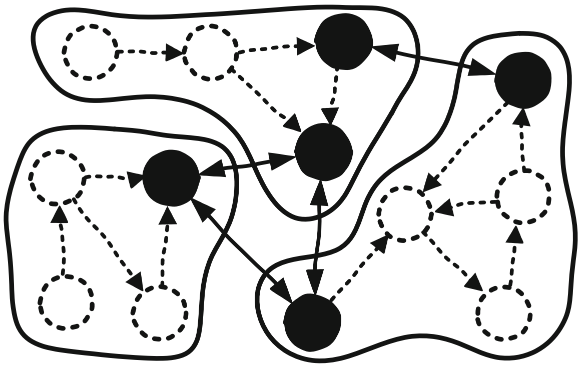

Fig. 3. Example of the network segmentation.

The further development of the planning software is seen in the establishment of subnetwork agents or company agents (depending on the situation) that are responsible for separate parts of the network that can work independently. Then, each subnetwork may be processed in a separate multi-agent environment, on separate remote hardware units interconnected via Internet in a P2P way. Then, the structure of the example changes to the following (Fig. 3). The black sites in this figure are the only visible outside of the subnetwork they belong to. The channels available between the subnetworks form the P2P communication channels between the multi-agent environments (swarms).

Thus, in this example we have three subnetwork agents connected via P2P service bus (Fig. 4). Each subnetwork agent "sees" the other connected subnetwork agents and the site agents that are directly connected via the channels. The sites that are visible from another subnetwork receive representative agents in this external network. These representative agents have limited functionality and only accept requests from other agents in the subnetwork to deliver them later to the real agent in another swarm. They also receive results (including planned limitations and costs) from the real agent and may respond fast to local requests inside the subnetwork swarms they are delegated to considering the previously detected limitations.

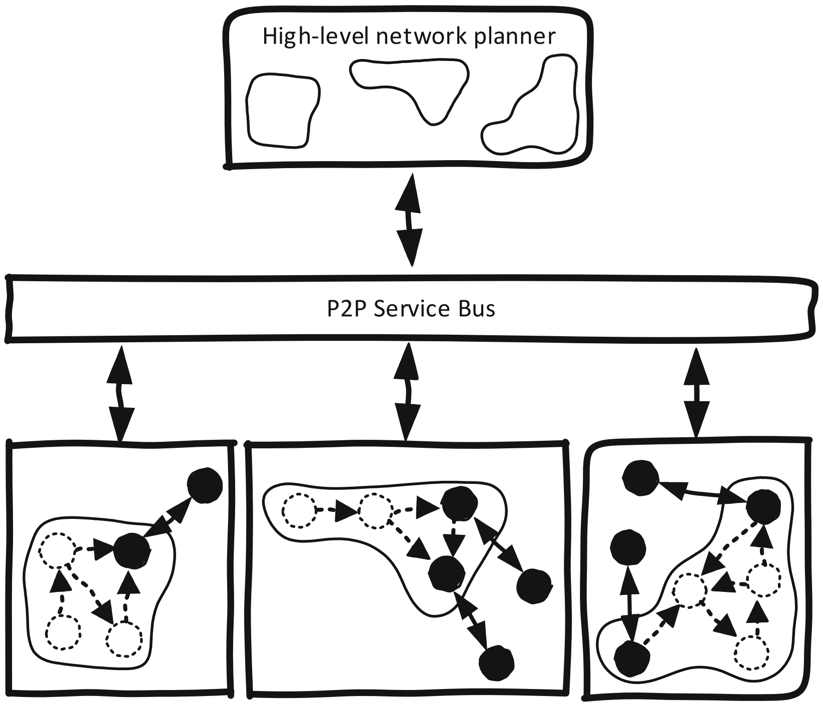

Fig. 4. Distributed architecture.

The planning principle is that any event is processed first inside the directly influenced multi-agent environment (planning application). If the result of this processing includes new supply requests to external sites or new supply limitations and costs affecting the existing contracts, the subnetwork agent negotiates changes with other subnetwork agents in the network.

The subnetwork agent is also responsible for the intensity and reasonability of the communication between the subnetworks (or companies). It may introduce additional virtual costs on the external requests, and thus motivate the local agents to solve the problems locally. It also decides when to coordinate with other subnetworks depending on how many changes (events to be sent) are accumulated and how fast the other agents respond.

It is important to note that the distributed coordinated planning system may include not only automatic multi-agent software planners, but also legacy planning software and even manual planning software depending purely on users. In this case, these software components are also connected to the same service bus using the same protocols and API. Users can see the incoming events, requests and limitations from other subnetworks, and can propose decisions based on the current plans, available options and target KPIs.

3.2 Asynchronous Processing

We compared two different approaches to the organization of multi-agent interaction in relation to the supply scheduling. One approach is based on request and reply and follows the rejection presumption principle, which means that if no reply is given it is an equivalent of rejection (sender must wait for an answer). This approach is referred to as rejection assumed interaction in the paper. Another approach is based on the acceptance presumption principle, which means that without explicit rejection from the counterpart of communication the acceptance of request is assumed. This approach is referred to as acceptance assumed interaction in the paper. Of course, this relates to the requests that do not require an informational feedback, but only ask another site to do something, while the feedback is optional.

Following the rejection presumption principle, the sites cannot process next request until they get a response from other sites regarding the previous request. Thus, in our case the Storage A becomes a bottleneck because both shops ask it first (as potentially cheaper source), and it cannot answer them both until Storage B answers the request. For example, the request processing is blocked at the Storage A on the steps ‘c’, ‘d’, and ‘e’ on the following diagram (Fig. 5), which leads to the delay of the processing of the request from the Shop Y on step ‘g’.

Specifically, when Storage A gets a request from Shop Z at step ‘c’, it sends a request for this product to Storage B (as it does not have it in the stock). When the request from Shop Y comes (almost the same time as from Shop Z), it cannot be processed until the request to Storage B is accepted.

We consider only one product in the network in the paper, so the orders compete for the same stock. If you get requests for different products, they theoretically may be processed immediately one after another, but this is an abstract situation. In practical tasks there are much more interdependencies between resources and demands other than just product type. For example, the channel capacity between Storage A and

Storage B, or the dispatch capacity at Storage B can be limited, or the transportation cost may depend nonlinearly on the volume transported. This prevents Storage A from answering the second request even if it is for a different product, until the acceptance of the first delivery is received (or assumed).

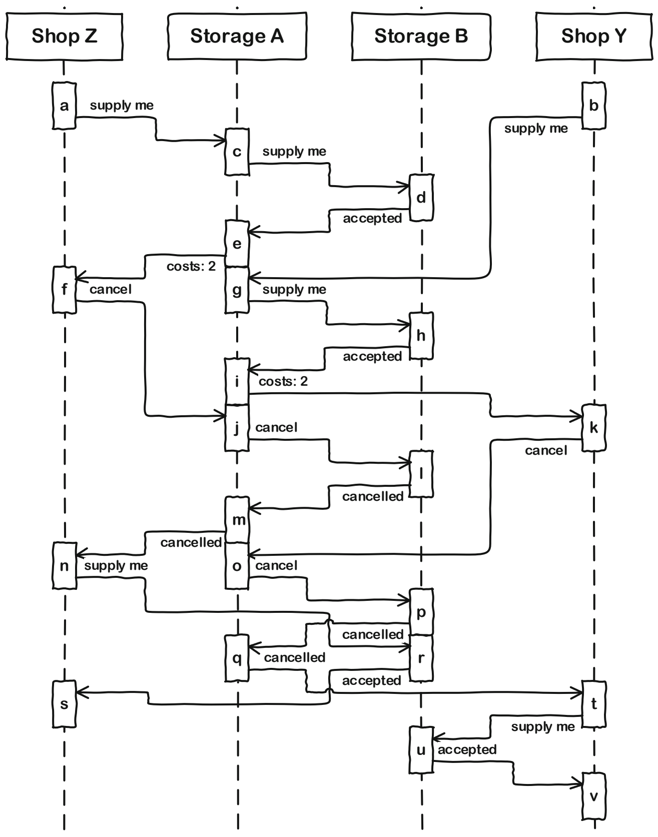

Fig. 5. Competitive interactions with rejection presumption.

The complete processing of the two orders with this approach takes 22 steps. Considering that some of them are done in parallel and some of them are very quick, this exact sequence takes 12.4 tu (time units).

Actually, we do not consider here the fully synchronous interaction that requires all events to be processed separately. It means that the order from the Shop Z is completely processed first, and only then the processing of the order from the Shop Y starts. This forces the whole sequence to go in one thread and take 16.6 tu.

The next diagram (Fig. 6) shows the interactions using acceptance presumption protocols.

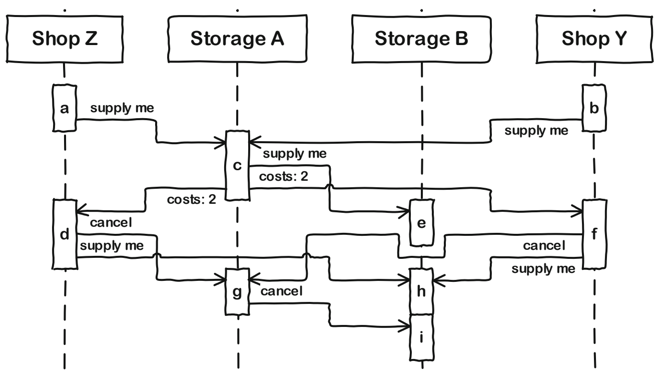

Fig. 6. Competitive interactions with acceptance presumption.

We can see that, in this example, the structure of the interactions is the same as in the case where we have only one order. The significant difference from the rejection presumption case is that at some steps several requests are processed by the site simultaneously. From one point of view, such steps should take more time, but from the other point of view, the processing of several requests at once never takes more time than separate processing of the same requests. What is more important, having several requests at once allows avoiding blind decisions that should be re-considered when the next request comes.

3.3 Self-regulation of the Processing Time

To ensure high speed of event processing, a method has been proposed that allows automatic adjustment of the duration of event processing by agents.

When choosing a planning option, each agent considers several possibilities and assesses them in terms of potential benefits. Normally, the agent, when he decides with whom to try to reach an agreement, chooses the most promising. But usually at this stage (before, in fact, negotiations) it is impossible to determine exactly what the negotiations will lead to. Therefore, there is always a chance that time will be spent, and the result will not improve.

In a situation where events are rare, you can consider more risky options, if they can potentially give a better result. If there is not enough time for processing, it is more reasonable to choose the option with the least risk (as a rule, with less negotiations) in order to get the result quickly.

For example, an agent may need a choice of how to deliver a product: request that it be added to a half-empty delivery or offer another need (later) to make room in a delivery that is full.

In the first case, a preliminary assessment will give more shipping cost, because half-empty delivery is less effective and more expensive. But at the same time there are almost no risks that it will not be possible to agree with the delivery agent.

In the second case, the cost of delivery is potentially lower (that is, more profitable), but there is a risk that a later need will not be able to find a resource for itself or the resource will be even more expensive. This can only be clarified after a request for another need and the completion of the entire chain of negotiations.

The essence of the proposed method is that, depending on the intensity of the flow of events, agents raise or lower the "bar" of acceptable risk. If in previous iterations of planning it (planning) could not be completed before the arrival of a new event, the level of acceptable risk decreases, agents choose those planning options in which the risk does not exceed the current level.

Risk is determined by the spread of values in the preliminary assessment of planning options. Zero risk means that only variants with exactly known results will be considered, i.e. not requiring negotiations.

If planning can be completed before the arrival of a new event, the level of acceptable risk increases, more negotiations appear, more options are considered.

3.4 The Method of "Negative Experience"

To improve the quality of planning without increasing its duration, it is proposed to introduce into the planning method an element of self-study, which consists in the following.

After reaching equilibrium and completing the planning iteration, the network agent (headquarters agent) evaluates the received plan for the overall network objective function. If the plan is improved, the staff agent initiates the iteration of the improvement in accordance with the "weak link" method. If the plan has worsened, the changes are not only canceled, but each agent in the chain of negotiations that led to deterioration, associates the magnitude of the deterioration with those agents with whom he entered into negotiations.

The next time when an agent considers a planning option, it not only assesses how much this option will improve its objective function, but also reduces this estimate if this option requires negotiations with those agents with whom there were previously negative results.

Thus, first of all, variants without "negative experience" are considered, which increases the probability of finding a better solution sooner.

4 Analysis of the Results

To determine the features of the developed methods, studies were conducted on the nature of the dependence of the processing results on various parameters of input data, incl. on their quantity. In a series of experiments, a model supply network with 30 suppliers, 100 distribution nodes and 1000 consumption points was used.

The dependence of the planning time on the number of orders was checked. At the same time, it is necessary to distinguish the number of orders and the volume of demand, because, unlike the volume of demand, each order can have its own preferences, purchase cost, delivery time and other parameters. In the series, the number of orders varied from 5 to 100 per node in the network, and for each value 8 separate experiments were carried out to eliminate the influence of random factors on the processing time. The measurement results show that the dependence of the planning time on the number of orders is close to linear (Figs. 7 and 8).

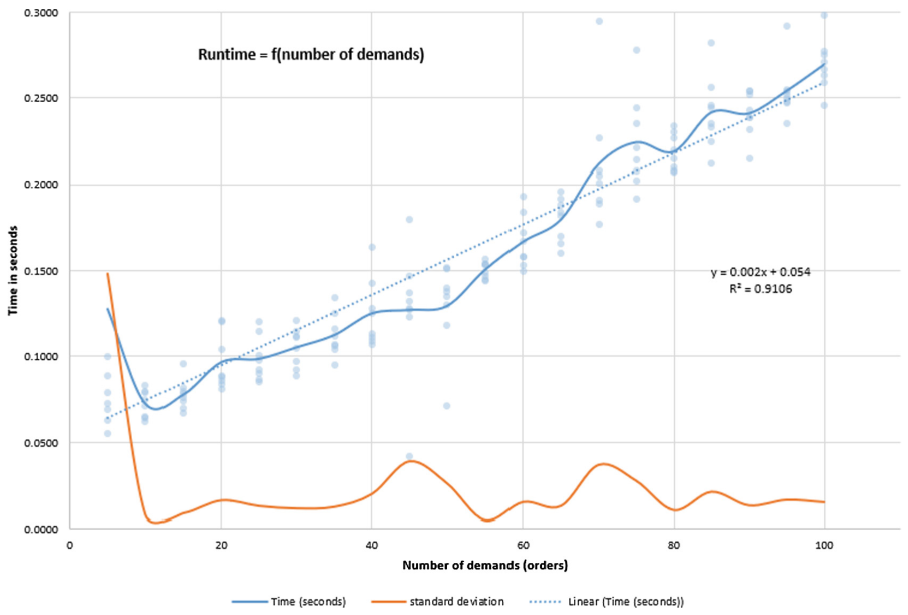

Fig. 7. The dependence of the duration of planning on the number of orders

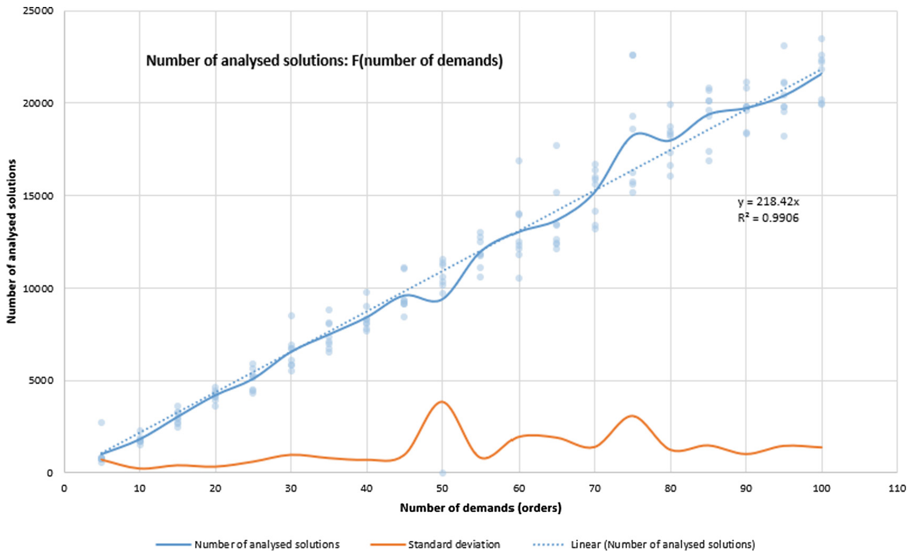

Fig. 8. Dependence of the number of analyzed solutions on the number of orders

In another series of experiments, the dependence of the planning time on the number of competitive orders is considered. At the same time, all orders are given the same price, high enough for any of them to be profitable (taking into account the cost of production and the cost of transportation along any route in the network). From the results of the experiments, it can be seen that the dependence of the planning time on the number of competing orders has a quadratic character (Fig. 9).

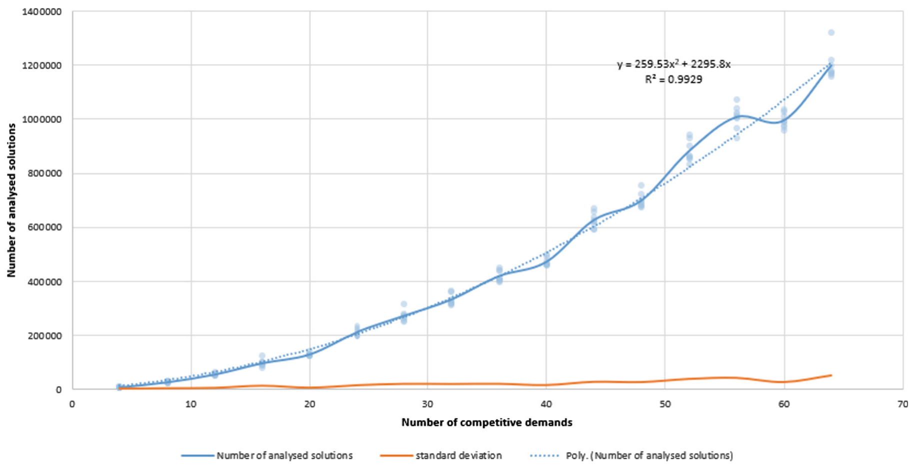

Fig. 9. Dependence of the number of analyzed solutions on the number of competitive orders

The developed methods and algorithms are intended for use in multiprocessor and distributed environments and are implemented in such a way as to effectively use the available capacity. If measurements are made with a different number of available computational cores, the planning time will differ significantly with the same number of plan variants considered by the system. A series of experiments shows the effectiveness of the use of available computing power. Also, experiments show the effectiveness of the proposed methods themselves (Fig. 10).

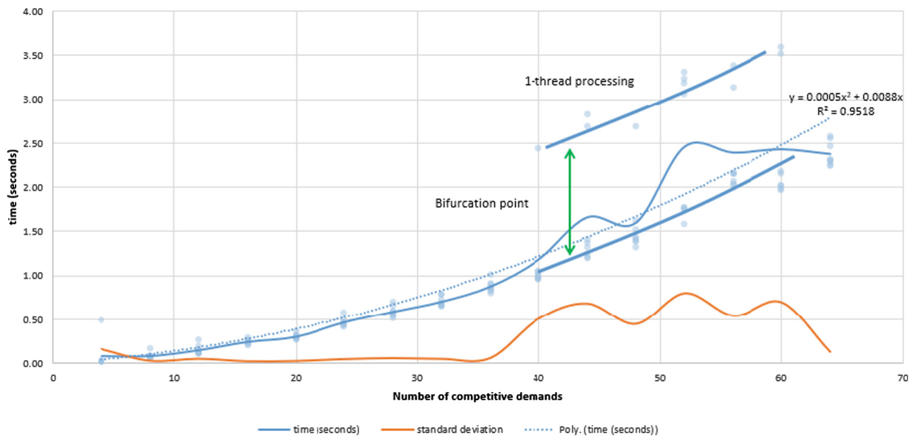

Fig. 10. Effect of parallel async processing

The choice of the approaches described in 3.2 depends very much on the specific practical case characterized by structure of the network and competitiveness of the demands. In general, more distributed networks are better processed using acceptance presumption approach. We used a real case data including more than 300 sites in the network (part of which is fully interconnected) and about 10 000 orders to model different interaction protocols. The network to be scheduled includes several factories, their storages that can interexchange materials and final products, and customer distribution centers that should be supplied. The model also includes production scheduling and some other features that affect the processing time in different situations. The modelling has been done using 16-core processor. The table below shows the results of the comparison (Fig. 11).

	Processing time (ms)	Messages between sites	Achieved quality ($)
Fully synchronous processing	737236	3200	1813499
Rejection presumption	191334	3140	1813359
Acceptance presumption	50275	2333	1812240

Fig. 11. Agent negotiation approaches comparison

The slight difference in quality between the synchronous processing and the rejection presumption most probably happens because of asynchronous stock competition between different orders.

Comparing the last two rows we can see that the use of acceptance presumption approach gives us 3.8 times faster processing and decreases the quality by about 0.1%, which seems to be a fair price in most cases.

A series of experiments was conducted in which the methods developed were compared with the simplex method. The number of consumers in the series increased from 50 to 300. Every consumer could receive products from any supplier. In this case, the total size of the network (the complexity of the problem) is determined by the number of possible delivery methods from the supplier to the consumer (the number of delivery channels). The total production capacity, its distribution over the network, the cost of production and transportation were set randomly for each individual experiment in the series.

According to the experiments, the processing time of an individual event remains approximately the same for any network size. It averages 14 ms, which is about 10 times faster than full rescheduling for a network with more than 250 customers.

The paper is prepared based on the results of the state financed research of ICCS RAS AAAA-A19-119030190053-2 “Development and analysis of the methods of analytical construction, digital knowledge representation, computing algorithms and multi-agent technologies used for optimisation of the control processes of complex systems”.

5 Conclusion

Based on the practical applications of the developed methods and the analysis we did it can be said that they allow achieving better results in terms of scheduling speed. Thus, they are especially useful in the tasks where real-time processing is needed.

Distributed parallel processing methods improve equipment utilization and can be applied efficiently in almost any practical case. Asynchronous processing also provides improvement in processing speed in multi-thread and distributed environment, but the specific approach should be chosen based on the specific practical case. Self-regulation and "negative experience" methods improve the real-time performance of the supply network scheduling systems by adapting the agent negotiations to the specific demand patterns and network structure, and make the dependence of the processing time on the number of demands more linear.

The future research will include the extension of the self-regulation methods with more parameters taken into consideration and automatically switching negotiation approaches based on the scheduling results evaluation.

References

1. Matsune, T., Fujita, K.: Designing a flexible supply chain network with autonomous agents. In: ICAART 2019, vol. 1, pp. 194–201 (2019)
2. Toader, F.A.: Production scheduling in flexible manufacturing systems: a state of the art survey. J. Electr. Eng. Electron. Control Comput. Sci. **1**, 1–6 (2017)
3. Rzevski, G., Skobelev, P.: Managing Complexity. WIT Press, Boston (2014)
4. Park, A., Nayyar, G., Low, P.: Supply Chain Perspectives and Issues: A Literature Review. Fung Global Institute and World Trade Organization, Geneva (2014)
5. Mohammadi, A., Akl, S.: Scheduling algorithms for real-time systems. Technical report, no. 2005–499, School of Computing Queen's University, Kingston (2005)
6. Joseph, M.: Real-Time Systems: Specification. Prentice Hall, Verification and Analysis (2001)
7. Pinedo, M.: Scheduling: Theory, Algorithms, and Systems. Springer, Heidelberg (2008)
8. Leung, J.: Handbook of Scheduling: Algorithms, Models and Performance Analysis. CRC Computer and Information Science Series. Chapman & Hall, London (2004)
9. Binitha, S., Sathya, S.: A survey of bio inspired optimization algorithms. Int. J. Soft Comput. Eng. **2**(2), 2231–2307 (2012)
10. Sun, S., Li, J.: A two-swarm cooperative particle swarms optimization. Swarm Evol. Comput. **15**, 1–18 (2014)
11. Tasgetiren, M., Sevkli, M., Liang, Y., Yenisey, M.: Particle swarm optimization and differential evolution algorithms for job shop scheduling problem. Int. J. Oper. Res. **3**(2), 120–135 (2008)
12. Vittikh, V., Skobelev, P.: Multiagent interaction models for constructing the demand-resource networks in open systems. Autom. Remote Control **64**(1), 162–169 (2003)
13. Chevaleyre, Y., et al.: Issues in multiagent resource allocation. https://staff.science.uva.nl/u.endriss/MARA/mara-survey.pdf. Accessed March 2015

14. Barbuceanu, M., Fox, M.S.: Coordinating multiple agents in the supply chain. In: Proceedings of the fifth Workshops on Enabling Technology for Collaborative Enterprises, WET ICE 1996, pp. 134–141. IEEE Computer Society Press (1996)
15. Stockheim, T., Schwind, M., Wendt, O., Grolik, S.: Coordination of supply webs based on dispositive protocols. In: 10th European Conference on Information Systems (ECIS), Gdañsk, 6–8 June 2002
16. Oliinyk, A.: The multiagent optimization method with adaptive parameters. Artif. Intell. J. **1**, 83–90 (2011)