Multithreading MAS for Real Time Scheduling

Yaroslav Shepilov
SEC "Smart Solutions" Ltd., Russia
shepilov@smartsolutions-123.ru

Daria Pavlova
SEC "Smart Solutions" Ltd., Russia
dpavlova@smartsolutions-123.ru

Daria Kazanskaia
SEC "Smart Solutions" Ltd., Russia
kazanskaya@smartsolutions-123.ru

Valerii Novikov
SEC "Smart Solutions" Ltd., Russia
vnovikov@smartsolutions-123.ru

***Abstract*— The scheduling is widely used for organizing the processes in time in different spheres ranging from the simple school timetables to production plans of the biggest factories with thousands of employees. In the most of cases this process has to be supported and managed by the complex tools, sometimes based on mathematical principles. In the paper the description of scheduling problem is given, the existing methods and tools for solving this problem are described, the approach based on multi-agent multithreading application is considered. Nowadays there exist numerous approaches to solving of the scheduling problem. The suggested method of multithreading multi-agent scheduling allows efficient and fast solution of complex problems featuring rapid dynamic changes and uncertainty that cannot be handled by the other methods and tools.**

***Keywords*— scheduling, multi-agent, multi-threading**

I. INTRODUCTION

Scheduling is an optimal resources allocation in order to achieve the set goals. It is an activity, associated with setting the goals and actions in the future.

At the same time, in mathematical terms the scheduling can be considered as a function, where one of the variables is time [1]. These approach is widely used in modern

Generally, scheduling implicates the performance of the following stages:

- Statement of problems and goals;
- Compilation of consequence of actions to reach the goals;
- Identification of required resources and sources;
- Identification of the executors and providing them with the schedule;
- Correction of the schedule and execution management [2].

As previously mentioned, scheduling is applied in almost every part of the human life, including the production, enterprise, public transport and transportation process organization.

Organization of production with hundreds and thousands of people is one of the intractable problems today. The enterprise has to put all of the organization employees' efforts towards the reaching common goals. Production scheduling in the modern conditions of severe competition in all the economy branches, is not a first step for management to ensure the company efficiency. A properly made production schedule can help to consider the best opportunities of the entire production process.

The main result of scheduling process is a plan, which quality can be measured by the following criteria:

- schedule flexibility and adaptability;
- orders delivered on time;
- resource utilization;
- response to the unexpected events in real-time;
- minimum down-time;
- identification and elimination of production "bottlenecks".

There are five common methods applied to the production systems scheduling and simulation:

1. Traditional optimization methods, based on the centralized algorithms such as mathematical methods, used in a compilation of integer and linear programming;
2. Genetic and neural network scheduling algorithms;
3. Heuristics algorithms that use business rules;
4. Multi-agent approaches using algorithms of solving optimization problem in distributed systems with constraints.
5. Multi-agent approaches using particle swarm optimization [3];
6. Multi-agent market methods using virtual currencies.

In the next chapters we will provide a brief overview of these methods, focusing on multi-agent approach, which is the basis of the scheduling system that is described in the paper.

II. TRADITIONAL OPTIMIZATION METHODS

Traditional approach to solving the optimization problems, based on the classical mathematic scheduling methods application, uses invariable and known in advance information about orders, resources and decision-making criteria. Common criteria system does not allow to consider individual preferences, constraints and features of all the resources. Application of this approach does not allow the system to response to the external data changes, which indicates a lack of efficiency of this method for solving complex real-life problems. Moreover, the algorithms of traditional mathematical approaches are rather complex and require large computational resources, which also prevents them from application for quick solution of complex problems under the conditions of daily enterprises operation [4]. Companies that apply these methods in their developments, create additional modules for real-life events processing that use hard-coded algorithms. However, these algorithms do not allow to follow the changes and rescheduling after the incoming events.

Basic constraints of the traditional centralized client-server architecture include centralized decision-making; sequential operation with no ability of concurrent problem solving;

978-1-4799-8676-7/15/$31.00 copyright 2015 IEEE
SNPD 2015, June 1-3 2015, Takamatsu, Japan

solution integrity, which prevents the part by part solution of the problem; low efficiency.

Systems, using the traditional optimization methods are usually ERP systems, such as Infor, SAP, BAAN, Manugistics, i-2, i-Log, QuintiQ, 1C, Galaxy, IT-Enterprise and others.

III. Genetic and neural network scheduling algorithms

Neural network is a mathematical model with software or hardware implementation, based on the organization and functioning of the biological neural networks – neuro cells of a living creature [5].

Neural networks became a common practice in the forecasting, classification or management problem solving [6, 7].

Neural networks allow forecasting the receiving of certain orders. This evaluation allows to schedule the future orders in a more optimal way. However, it is only approximate and is effective only when orders have distinctive features. Moreover, neural networks have a preliminary education phase and require data sets that include the resulting schedules to learn. Neural networks parameters are also defined without accuracy. Therefore, the number of neuros in the network is defined empirically and there are only recommendations about the network size, sometimes contradicting each other.

Systems using neural networks: Dendral, Mycin, Intelligent Inc.

IV. Multi-agent scheduling algorithms

The general concept of the agent-based algorithms is a decentralized decision-making, when the solution is obtained by the synchronous or asynchronous interaction between the agents representing the objects of the real world [8]. The implementations of this approach include the following:

- Distributed Constraint Optimization Problem (DCOP) is a mathematical discipline, describing the methods of agents application to the distributed constraint optimization problems. The general concept of these algorithms is a decentralized decision-making, dynamic decision forming and a tendency towards the balance, when a multi-agent system finds a new balance under the external influences [9].
- Asynchronous Distributed Constraint Optimization (ADOPT) uses the "top-down" approach when the problem is decomposed into simpler sub-tasks that can be solved by the agents. This method represents a depth-first search on the set of variables with numerous improvements of basic searching strategies. Its general structure is similar to the distributed version of branch and bound algorithm implemented by the means of multi-agent systems. An agent represents each node of the tree of constraints. Each agent contains maximum and minimum cost values for the sub-task considering the condition of the predecessor. The agent is connected to the predecessor node agent and descendant nodes agents. Node agent gives the agents of descendant nodes a task to find the solution. In this case, the descendants ignore the partial solutions, which cost is higher than the available value, as the descendant-agent already knows that it can obtain a better solution [10].
- Optimal Asynchronous Partial Overlay (OptAPO). In this method the solutions are built using "down-top" approach and then combined in a consistent manner [11].
- Asynchronous Backtracking (ABT). When a problem occurs during the message exchange, agents track the differences in constraints, so we return to the previous task state and correct it.

However, distributed algorithms still suffer from a batch mode operation and exponential increase of messages number and size during continuous interaction between agents. In this paper we describe the attempt to overcome the traditional disadvantages of the multi-agents system by creating the scheduling system that can react to the events in the real-time. The first version of the real-time scheduling system was developed two years ago and was used as the basis of production management system [12]. The system still had several constraints: it used synchronous one-threading interaction that does not allow to reach the full potential of agent-based algorithms. The second version that applies multi-threading asynchronous interaction is the focus of the next chapters.

V. Multi-threading multi-agent scheduling system

A. General principles

The developed multi-agent scheduling system is based on the main agent ideas and concepts. Let us consider these principles to provide the reader clearer understanding before describing the scheduling process.

The core of the system is the world of agents that consist of multiple agents that interact with each other, have their own goals and instructions and can react to the impact of the external world.

To create the world of agents, the system requires to retrieve the initial data from the legacy systems to populate the agents. Then the matching agent is created for each entity of the real world (for example, an employee agent for an employee etc.)

Each agent has its own goals that describe the results that should be achieved by solving the problems and resolving the conflicts. The system of agent goals can be complex while the goals themselves can be the result of the external factors or agent's own activity. The agent can perform specific actions to achieve its goals. However, to narrow the options of the possible actions, the agent should know the range of actions it can perform, required conditions and probable consequences.

The actions performed by the agent are the results of certain events. Event is an update of the entity due to changes in the external environment. A message is the notification on an event.

Events can be defined in advance or unexpected. The ability of the system to react to both classes of events in real time provides more flexibility in decision-making.

An agent subscribed for the updates of an entity receives the corresponding messages from the world of agents that are stored in the message queue. The agent starts processing them according to the importance of the event. The messages can be also sent from agent to agent.

The types of messages include the notifications about job start and end, employee unavailability, new orders, changes in orders priorities, equipment failure, and changes in supplies delivery time. For example, if certain supplies were received in advance, the system can reschedule the orders and start the execution of the important order earlier to minimize the risk of penalties. 15 minutes delay in operation execution can result in reallocation of the operations to other employees to ensure that the final assembly is done within the deadline. During the reallocation the system considers only related operations, not the complete schedule.

The recalculation only of the part of the schedule that is influenced by the event is the main feature of the adaptive system: the plan is not created from the scratch every time, but adjusted according to the events in real time. Such adjustment is the result of conflicts, negotiations and compromises between the agents. The chart in Fig.1 shows the number of entities changed as a result of agents activities. As can be noticed, the creation of the schedule or major events involve almost whole agents world, while minor events cause changes only in small number of agents.

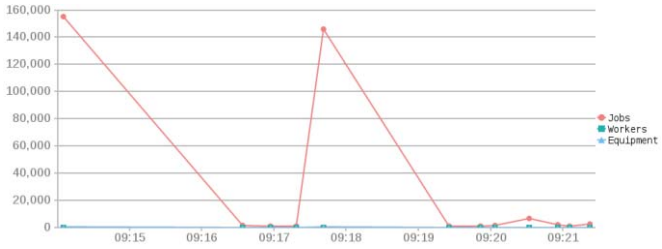

Fig. 1. The number of changed entities in time

The system calculates the plans in the real time allocating new jobs after the previous ones were completed. In the process of decision-making the system considers all events occurred by the current moment: the employee will never be allocated to the job that cannot be completed because of lack of components, equipment failure, etc.

The mechanism of the multi-agent scheduling can be implemented by two methods:

- Single-threading scheduling: all agent messages are processed in one thread. This method is simpler to implement, however it can lower the computation speed.
- Multi-threading scheduling: parallel execution of asynchronous processes result in increase in system productivity.

The latter option is the focus of the section.

System multi-threading is achieved due to the ability of the agents to process their messages in parallel in different CPU threads (see Fig.2). Let us consider the negotiation process presented in the figure in more details.

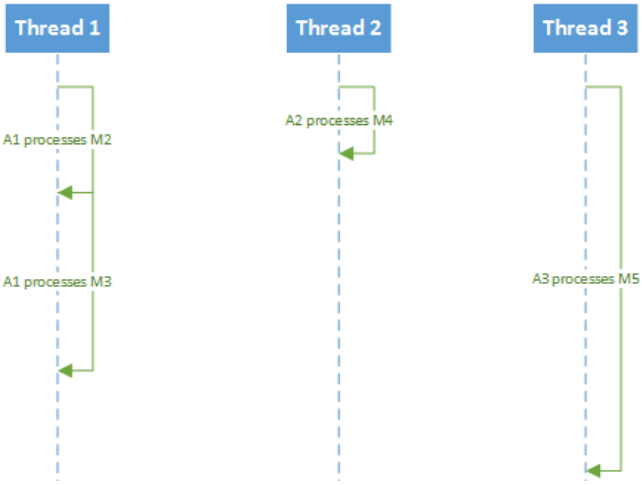

Fig. 2. Parallel processing of messages in different threads

Agent A1 processes message M2 in Thread 1. In parallel Agent A2 processes its message M4 in Thread 2, Agent A3 processes message M5 in Thread 3, etc. Operation of each agent does not influence the operation of other agents. After an agent processes its message, the thread becomes free and it is ready to receive another agent for operation (or the same agent with a new message). Processing of different messages can take different time: as one can see from the figure, A3 still processes its message while A1 has already completed its processing. However, it is does not result in additional delay since the agents work asynchronously and occupy different processor threads.

The message exchange is implemented with the special mechanism called message whiteboard. The whiteboard itself is a high-level agent that coordinates the tasks between the agent that requires the resources (demand agents) and the agents that can provide the resources (resource agents). The agents can indicate their demands and resources by making the matching rules. Each agent sends its rule with the demands and resources to the whiteboard. According to these rules, the whiteboard selects the most optimal matches.

A simpler mechanism of message processing is negotiating between the agents. Since the agent is an intelligent entity, it can decide which negotiation method is the best according to the data received from the world of agents. The type of the received message allows the agent to specify the complexity of the processing.

As any intelligent entity, the agent has a certain lifecycle in the system: it is born, it lives and it dies. Agent lifecycle is included in the planning cycle that consists of the following steps:

a) Start. At this stage the world of agents is created. In the world there exist and interact the instances of applications implementing the required agents functions including the basic agent interaction mechanisms.

b) Data load. The loading of the initial and updated data for the system operation*;*

c) Creation of agents. At the first stage the world of agents sends the creation and activation messages to all agents that were created ("wake-up" message);

d) Agents initialization. The agents define their goals, priorities, criteria according to the data they received from the agents world. Each agent decides to which updates from the specific agents it will be subscribed. At this stage the threads are started and the parallel operation of the agents can be started;

e) Agents validation. Agents specify how accurate their goals, priorities and criteria are defined;

f) Agents operation. The agents start operating according to their instructions to achieve their goal in parallel asynchronous mode;

g) Achieving the compromise. The agent finds the best solution by negotiating with other agents or on its own, after that the agent operation is stopped;

h) Saving the results. Solution achieved by the agents is saved;

i) Receiving the events. Notification on the events from the real world is received by the world of agents;

j) Cycle repeated. The data is uploaded or updated according to the received event (stage b) and the cycle is repeated.

After the completion of the cycle, the agents transmit to the pending state when they do not perform any actions until they receive a specific message from other agents.

B. Scheduling mechanism

The scheduling starts when the system receives a data set that will be used for creating a plan for a given time period. The data is loaded from the factory legacy systems or is input manually.

The calculation of the plan is done by negotiation of agents. The schedule is not saved until all agents finish their activities. Activity is an action of an agent in the world of agents: message processing, waiting for the response etc. The world of agents checks the completion of all activities. As soon as the agent transmits to the pending state, it should indicate that its activities are completed by sending a message to the world of agents.

There are two types of agents in the implemented multithread scheduling system: agent of job and agent of resource. The resources are represented by the employees, equipment and workshops:

- Agent of employee represents an employee that can perform a certain type (or types) of jobs, has specific skills, can use the equipment and is ready to perform any relevant job;
- Agent of equipment represents a unit of equipment that has the specific model that can be used by the employee to perform specific type of jobs;
- Agent of workshop is looking for jobs and services from other workshops to perform them in its own facilities.

Agent of job is representing a technological operation that is looking for its allocation in the schedule according to the given criteria (employee, equipment). To satisfy the requirements of the agent of job, the agent of employee must be able to perform the job of this type, have required skills and be able to work on the specific equipment model required for the job.

Agent of job is the most active agent: it reacts to the allocation request from the employee, can be initiated by the agent of the related job or just take part in conflicts resolution. In order to be allocated to the specific slot in the schedule, the agent must satisfy all above mentioned criteria. Employee agent must be relevant to the given parameters to satisfy the demands of the job agent. This can result in long interactions between the agents.

The long negotiations on allocation can be avoided by using the message whiteboard described in the previous section. Agent of job leaves the required demands in the rules while employee agent leaves the resources it can provide in the rules. Then the whiteboard analyses the rules and informs the agent on the matches found.

Negotiations take considerably long time since there is a huge number of agents of jobs that want to be allocated to the best slot in the schedule, while at the same time many agents of employees and equipment can match many jobs. The number of agents considered during allocating can be decreased by several criteria: priority, availability, response time, etc.

The main scheduling process is done during the stage of achieving the compromise of the agents lifecycle. Thee agent finds the best allocation option by the negotiations with other agents or by its own means. Then agent activity is stopped and check of the event scheduling accuracy starts. This check consists of correct event processing and the schedule consistency checks. Event processing check is required to ensure that all changes triggered by the scheduling were effected (for example, the fired employee has no operations in his schedule, new order is completely scheduled). Only after the check for plan accuracy, the schedule is stored and available to the user.

In Fig. 3 the chart that displays the number of active agents is shown.

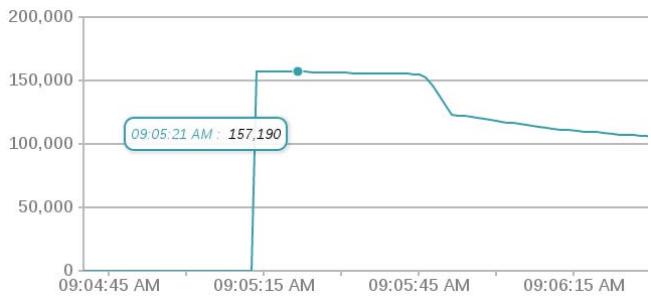

Fig. 3. Active agents chart

The chart shows that the equilibrium was not achieved yet, since the number of active agents is still significantly large. This chart can also provide information on the stability of the achieved equilibrium and to which extent the events influence the achieved result (the increase in chart means that agents were moved from final or optimal state).

The mandatory condition of agents existence is the existence of the world of agents. The world is considered as active if at least one agent is active. During its operation, the world runs the parallel operation of the agents by running the CPU threads. All CPU threads can be run simultaneously and work in parallel. For example, if the CPU has eight cores, eight agents maximum can process their messages at the same time. After the message processing is completed, the thread is disengaged and will be occupied by the agent the scheduler chooses to activate. The threads can be free during a certain time, but a thread can be occupied only by one agent at a time.

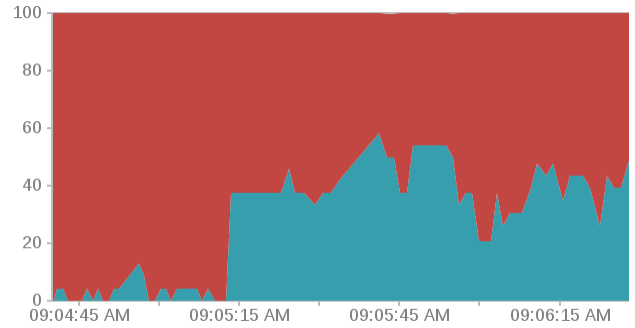

Fig. 4. Processes workload chart

The chart in Fig.4 shows the load of the processors. This chart allows one to analyze the efficiency of computational capacity utilization, degree of paralleling and asynchronous agents operation, estimate the required computational resources to solve the current task with the given amount of data.

In the result, the agents that have only finite set of instructions can interact with each other by the given rules and generate complete schedule considering different interests and constraints.

C. *Comparison of single-threading and multi-threading solutions*

The results for single-threading and multi-threading schedulers are presented in the table below.

TABLE I. COMPARISON OS TWO SCHEDULERS

Indicator	Single-threading	Multi-threading (8 cores)
Number of the existing agents in 1 sec	1 – 6000	1 – 50000
Number of messages sent for each agent type	3 - 10	3 – 20
Number of messages sent in 1 sec	1000000	500000
Thread idle time	1 msec	30 mcsec
Schedule density: % of delayed orders; % of equipment idle time; % of employees idle time.	10 of 100; 20 of 50; 20 of 200	5 of 100; 10 of 50; 10 of 200
Scheduling speed (number of orders per time t);	150000 for 40 min	150000 for 5 min

The results have shown, that in the case of multi-threading scheduling, the schedule density increases significantly, which means that the workload is uniform, there is no idle time for workers and equipment.

Moreover, the multi-threading schedule shows significant increase in the scheduling speed at the same time providing the ability to handle more instances of agents.

The given indicators allow us to conclude that the multi-threading scheduler provides more adaptivity in response to the unexpected details, i.e. even if the event results in significant change in the schedule, it will be done in reasonable time, since the scheduler can handle larger number of events in shorter time.

VI. CONCLUSION

The developed approach of multi-threading scheduling shows significant advantage in comparison with the existing scheduling systems and single-threading scheduling. The developed system allows rescheduling in real time, introducing new criteria and complexity degrees, adaptively react to the events in real time, increase scheduling speed and density due to asynchronous interaction of agents.

Industrial version of the system is deployed in two factories providing the ability to manage workshop schedules, reduce idle time and order delays. However, the scheduling system is universal and can be used for any specified domain. Currently it is applied for ferries management and scheduling of hospital staff and facilities.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007- 2013 under grant agreement n° 314056.

REFERENCES

[1] Ravindran, Binoy, E. Douglas Jensen, and Peng Li. "On recent advances in time/utility function real-time scheduling and resource management." Object-Oriented Real-Time Distributed Computing, 2005. ISORC 2005. Eighth IEEE International Symposium on. IEEE, 2005.

[2] G.Rzevski, P.Skobelev. “Managing complexity”. WIT Press, London-Boston, 2014.

[3] Shi, Jonathan Jingsheng, and Daniel W. Halpin. "Enterprise resource planning for construction business management." Journal of Construction Engineering and Management 129.2 (2003): 214-221.

[4] Pěchouček, Michal, and Vladimír Mařík. "Industrial deployment of multi-agent technologies: review and selected case studies." Autonomous Agents and Multi-Agent Systems 17.3 (2008): 397-431.

[5] Hagan, Martin T., Howard B. Demuth, and Mark H. Beale. “Neural network design”. Boston: Pws Pub., 1996.

[6] Zhang, H-C., and S. H. Huang. "Applications of neural networks in manufacturing: a state-of-the-art survey." THE INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH 33.3 (1995): 705-728.

[7] Hippert, Henrique Steinherz, Carlos Eduardo Pedreira, and Reinaldo Castro Souza. "Neural networks for short-term load forecasting: A review and evaluation." Power Systems, IEEE Transactions on 16.1 (2001): 44-55.

[8] P.Skobelev. “Multi-Agent Systems for Real Time Adaptive Resource Management. In Industrial Agents: Emerging Applications of Software Agents in Industry (Invited Chapter)”. Elsevier, 2014.

[9] Liu, Jyi-Shane, and Katia P. Sycara. "Exploiting Problem Structure for Distributed Constraint Optimization." ICMAS. Vol. 95. 1995.

[10] Modi, Pragnesh Jay, et al. "ADOPT: Asynchronous distributed constraint optimization with quality guarantees." Artificial Intelligence 161.1 (2005): 149-180.

[11] Petcu, Adrian, Boi Faltings, and Roger Mailler. "PC-DPOP: A New Partial Centralization Algorithm for Distributed Optimization." IJCAI. Vol. 7. 2007.

[12] V. Shpilevoy, A. Shishov, P. Skobelev, E. Kolbova, D. Kazanskaia, Ya. Shepilov, A. Tsarev. Multi-agent system “Smart Factory” for real-time workshop management in aircraft jet engines production // Proceedings of the 11th IFAC Workshop on Intelligent Manufacturing Systems (IMS’13), May 22-24, 2013, São Paulo, Brazil. 2013. – P. 204-209. ISBN 978-3-902823-33-5.