ADVANCES IN CYBER-PHYSICAL SYSTEMS

Vol. 7, Num. 2, 2022

MULTI-AGENT COORDINATION WITH DEFERRED ASYNCHRONOUS MESSAGING IN A DISTRIBUTED COORDINATION SPACE

Alexey Botchkaryov

Lviv Polytechnic National University, 12, Bandera Str, Lviv, 79013, Ukraine.
Author’s e-mail: *oleksii.y.bochkarov@lpnu.ua*

https://doi.org/10.23939/acps2022.__.___
Submitted on 26.09.2022

***Abstract*: A method of multi-agent coordination with deferred asynchronous messaging in a distributed coordination space has been proposed. The method has been based on the concept of multi-agent conditional interaction. The method has used 1) a distributed coordination space in which agents move, 2) the rules of state transitions for the coordination space nodes depending on the movements of agents, 3) the rules of agents move and state transitions depending on the states of the coordination space nodes, 4) a multi-agent coordination game based on the coordination space and the rules. The coordination space has been implemented based on the distributed shared memory of agents. The rules have been applied by exchanging deferred asynchronous messages between agents through the distributed shared memory. The agent's decisions about movement in the coordination space and their consequences are interpreted according to the rules in asynchronous messages. Delivery of messages to other agents has been deferred until these agents visit the corresponding nodes of the coordination space. This has ensured 1) mutual exclusion when agents choose conflicting actions, and 2) resilience of multi-agent coordination to agent failures and loss of coordinating messages. Four multi-agent coordination games have been considered as examples. The issue of fault tolerance of the proposed coordination method has been considered. The simulation results show that the use of the method ensures the resilience of multi-agent coordination to agent failures in the considered coordination games.**

***Index Terms:* multi-agent system, multi-agent coordination, coordination space, coordination game.**

I. INTRODUCTION

Research in the field of multi-agent systems is becoming increasingly relevant as a result of technological breakthroughs in the field of computer technology, wireless communications, and robotics. New opportunities require new approaches to the development of technologies for multi-agent systems. One of the main directions here is the use of the resource of decentralized control and the principles of self-organization. It is assumed that a multi-agent system should independently solve complex problems in the absence of centralized control and limited local information interaction of agents [1]. To achieve this goal, various decision-making methods are being developed in multi-agent systems [2] and methods for coordinating the joint actions of agents solving a problem [3], including the usage of machine learning methods [4]. Multi-agent systems can be applied to solve important problems in many areas, such as business process automation, computer network management, distributed robotics, resource allocation and scheduling [5], smart city management [6], and many others.

A promising direction in the field of research of multi-agent coordination methods is the use of ideas and principles of game theory [7]. This approach allows to use the accumulated experience and solutions in the field of game theory to organize the interaction of agents and coordinate their joint actions [8]. Of great interest there are also approaches to solving the problem of multi-agent coordination at the intersection of different research areas, for example, combining the principles of game theory and reinforcement learning methods [9]. It is important to note that from the point of view of the general problematics of multi-agent coordination, models and solutions in the field of games of timing [10], including games of timing with many players [11], as well as applied solutions in this area [12] are of particular interest.

The article considers a new approach to the creation of a multi-agent coordination method based on the concept of multi-agent conditional interaction. The article proposes a method of multi-agent coordination with deferred asynchronous messaging in a distributed coordination space. The method implements the idea of preserving and using the history of inter-agent interactions in an explicit convenient form, which makes it possible to organize the spontaneous emergence of coordinated collective behavior. A set of requirements for multi-agent coordination methods is formulated and the compliance of the proposed method with these requirements is considered. As an example, the article considers four multi-agent coordination games that have broad prospects for practical application in multi-agent coordination problems within the framework of the proposed method. The issue of fault tolerance of the proposed coordination method is considered.

II. OBJECTIVES

Based on the analysis of existing solutions in the field of multi-agent coordination, the following set of requirements for a new coordination method can be formed.

1. Possibility with a high degree of freedom to take into account and control the dynamic structure of cause-and-effect relationships that arise in the course of the development of the process of inter-agent interactions.

2. The ability to flexibly regulate the level of awareness (the degree of lack of information) and the freedom of action of each agent participating in the process of inter-agent interactions.

3. Sufficient level of abstraction, allowing to offer a wide range of meaningful interpretations of the processes of inter-agent interactions in the course of coordinating joint actions.

4. The possibility of humans participating in multi-agent coordination games within the framework of the coordination method as an experimenter or as an agent whose behavior is coordinated with other intelligent agents.

5. A convenient representation of the history of the process of inter-agent interactions in the course of coordinating joint actions.

Thus, the following objectives can be formulated:

1. Propose a multi-agent coordination method that meets the above requirements.

2. Implement mutual exclusion of conflicting actions of agents in the coordination method.

3. Implement the resilience of multi-agent coordination to agent failures and loss of coordinating messages.

4. Consider multi-agent coordination games within the proposed method of multi-agent coordination.

5. Estimate the fault tolerance of the proposed coordination method. Limit the increase in the average solution time of a coordination game due to agent failures to 30 % for the intensity of the stationary Poisson process of agents' failures less than 0.02.

III. MULTI-AGENT COORDINATION METHOD

The proposed coordination method is based on the concept of multi-agent conditional interaction. Multi-agent conditional interaction can be generally expressed in the following terms. 1) The events take place in a common discrete time and a discrete structured space (represented by a set of nodes connected with edges). 2) Agents are in a state of continuous movement in the space (passing a single edge per move). 3) At the beginning all agents start at different positions. 4) While moving, an agent leaves a fixed-length trail that spans along the traversed path. 5) An agent cannot move along the trail of any other agent (i.e. an agent's trail cannot overlap with other agents' trails (even with itself), but crossing other agents' trails is allowed).

From the point of view of conditional interaction and the corresponding opportunities for coordination, two main aspects can be distinguished in this case.

1. Strong (or external) conditioning. Alternatives available to an agent (where to move next) depend on what path other agents have chosen before. The history of these choices is presented explicitly in the form of a collection of trails left by agents. This also represents their common interaction history. We can regulate the strength of conditioning by altering the length of trails. Trail-free space explicitly represents alternatives available to the agents to choose from.

2. Weak (or internal) conditioning. An agent's decision on where to move next can depend on its knowledge of choices other agents made at this node. That is, if a node has been visited by other agents before, this could provide additional decision-making information for an agent (e.g. directions of existing trails, their "strength", etc.). In addition, if there are several such trails, the agent can take into account the configuration of their mutual intersection.

The method of multi-agent coordination based on multi-agent conditional interaction is given by: 1) a tuple <M, A>, where M is a distributed coordination space in which agents A move (Fig.1); 2) a tuple <RM, RA>, where RM are the rules of node state transitions resulting from agents passing through the node, and RA are the rules of agents' movements and state transitions depending on the states of passed nodes; 3) a multi-agent coordination game implemented in the space M based on the rules <RM, RA>.

The coordination space M is implemented based on the distributed shared memory of agents or using any other distributed data structures with similar functionality, for example, blockchain technology. The <RM, RA> rules are applied by exchanging deferred asynchronous messages between agents through the distributed shared memory. That is, the agent's decisions about movement in the coordination space and their consequences are interpreted according to the <RM, RA> rules in asynchronous messages. Delivery of messages to other agents is deferred until these agents visit the corresponding nodes of the coordination space. This ensures 1) mutual exclusion when agents choose conflicting actions, and 2) resilience of multi-agent coordination to agent failures and loss of coordinating messages. In the proposed coordination method, three levels can be distinguished: 1) distributed coordination space M, which displays the structure of cause-and-effect relationships in the problem being solved by the multi-agent system and the corresponding domain; 2) rules <RM, RA> that implement the mechanism of conditional interaction of agents; 3) a multi-agent coordination game in the space M based on the rules <RM, RA>, which implements the logic of coordinating the joint actions of agents in the process of solving the task.

To estimate the fault tolerance of the multi-agent coordination method, we used the increase rate of the average time to solve a coordination game $W_T = (T_n / T_f) \times 100\ \%$, where T_n is the average time to solve a coordination game without agent failures, and T_f is the average time to solve a coordination game with agent failures. A failure model is in the form of a stationary Poisson process of agents' failures with the intensity λ in the range from 0.001 to 0.02.

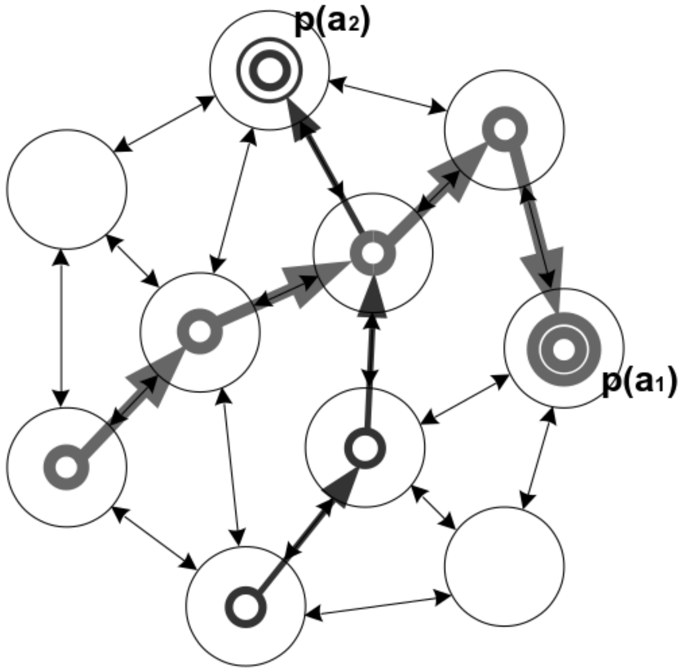

Fig. 1. An example of a section of the coordination space M with two agents (a_1, a_2) that move through its nodes, leaving behind a trail: $h(a_1) = 4$, $h(a_2) = 3$.

IV. DISTRIBUTED COORDINATION SPACE

The distributed coordination space is defined by an undirected graph with every node's minimum local degree strictly greater than 2 (i.e. each node is connected to at least two other nodes). In the general case $M = (V, E)$, where $V = \{v\}$ is a set of nodes of the coordination space, and $E = \{e\}$ is a set of edges such that $e = \{v_i, v_j\}$, $i \neq j$.

The following tuple is defined for each node v

$(E(v), C(v), S_t(v), D_t(v), Q_t(v))$,

where

$E(v) = \{e_1,e_2,...,e_{n(v)}\}$ is a set of edges connected to node v, and n(v) is the number of those edges;

$C(v) = \{c_1,c_2,...,c_{m(v)}\}$ is a set of all possible pairs of edges (i.e. paths through the node) such as $c_k = (e_i,e_j)$, $i\neq j$, $e_i,e_j \in E(v)$, m(v) is a number of these pairs, and $C(v,e_i) \in C(v)$ is a set of all pairs of edges which include edge e_i;

$S_t(v) = \{s_t(c_1),s_t(c_2),...,s_t(c_{m(v)})\}$ is a set of states of all pairs of edges for node v at time step t with $s_t(c_k) \in \{0;1\}$ being a state of pair of edges c_k at time step t such that if $s_t(c_k) = 0$, then pair of edges c_k is closed (i.e. the agent cannot pass through the node in this direction), and if $s_t(c_k)=1$, then pair of edges c_k is open (i.e. the agent can move through this node in this direction);

$D_t(v) = \{d_t(e_1), d_t(e_2),..., d_t(e_{n(v)})\}$ is a set of alternatives' indicators such as

$d_t(e_i) = \Sigma s_t(c_j)$, for all $c_j \in C(v,e_i)$;

i.e. $d_t(e_i)$ is the sum of states of all pairs of edges which include edge e_i and it indicates the number of alternatives available to the agent if it enters node v from edge e_i: if $d_t(e_i) = 0$, then an agent cannot exit node v through edge e_i since all corresponding pairs of edges are closed; if $d_t(e_i) = 1$, there is no choice as such, agent has a single way to exit only (through the open pair of edges);

$Q_t(v) = \{q_t(c_1), q_t(c_2),..., q_t(c_{m(v)})\}$ is a set of time counters; if pair of edges c_k is closed, then corresponding counter $q_t(c_k)$ indicates the number of time steps until the pair of edges c_k become open; if $s_t(c_k) = 1$ (i.e. when a pair of edges c_k is open), then $q_t(c_k) = 0$.

The integral state of node v is defined as the current configuration of all values in $(S_t(v), D_t(v), Q_t(v))$.

The $A = \{a\}$ set consists of N agents, such as $N<K$, where K is the number of nodes in the coordination space M. The following tuple is defined for each agent:

$(x_0(a), x_t(a), s_t(a), h(a), f_a)$,

where

$x_0(a) \in V$ is a node of the space M, where the agent is placed at the beginning ($t = 0$),

$x_t(a) \in V$ is a node of the space M, where the agent currently is,

$s_t(a) \in \{0;1\}$ is a state of the agent at time step t, such as if the agent can move, then $s_t(a) = 1$, and if the agent is trapped (i.e. has no alternatives), then $s_t(a) = 0$,

h(a) is the agent's trail length,

f_a is the agent's movement function, such as

$x_{t+1} = f_a(x_t, y_t)$,

where

$x_{t+1} \in V$ is a node where the agent moves next,

$y_t = \{\delta_t(e_i)\}$, $e_i \in E(x_t, I(x_t))$ is a vector, indicating whether it is possible to move into an adjacent node through the corresponding edge e_i, here $I(x_t)$ is a set of nodes adjacent to node x_t (excluding x_{t-1}); $E(x_t, I(x_t))$ is a set of edges between node x_t and $I(x_t)$ nodes (i.e. if $e_i \in E(x_t)$ and $e_i \in E(I(x_t))$, then $e_i \in E(x_t, I(x_t))$); and $\delta_t(e_i)$ is a transition possibility indicator, such as

$\delta_t(e_i) = 0$, if $d_t(e_i) = 0$,

$\delta_t(e_i) = 1$, if $d_t(e_i) > 0$,

where $d_t(e_i)$ is an indicator of alternatives, that shows the number of paths available at time step t+1 if the agent enters node through edge e_i at time step t.

V. RULES OF NODE STATE TRANSITIONS

RM are the rules of node state transitions resulting from agents passing through the node. In general, RM consists of three rules: RM = {RM1, RM2, RM3}.

RM1 is a "node closing" rule, that defines a state transition for pairs of edges C(v) depending on the agents' moves:

$\forall$ $v \in V$: if edge $e_i \in E(v)$ is used by the agent to enter or exit node v at time step t,

then $\forall$ $c_k \in C(v,e_i)$

set 1) $s_t(c_k) = 0$; 2) $q_t(c_k) = h(a)$.

RM2 is a "trail fading" rule, that defines the temporal state transition for pairs of edges C(v). Each time a step counter $q_t(c_k)$ for each closed pair of edges c_k is decreased by one. If it reaches 0 for a particular pair, then the pair opens (i.e. the trail finally fades away):

$\forall$ $v \in V$, $c_k \in C(v)$:

if $q_t(c_k) > 0$, then $q_{t+1}(c_k) = q_t(c_k) – 1$.

if $q_{t+1}(c_k) = 0$, then $s_{t+1}(c_k) = 1$.

RM3 is an "edge dependence" rule that defines a state transition for pairs of edges caused by the state change of other pairs of edges. To reflect the different aspects of the domain represented by the coordination space we can define additional dependencies between independent pairs of edges of C(v). According to this dependence, if one pair of edges goes into a closed state, then the other one also closes. Employing the predicate $P^*(e_i,e_j)$, $e_i,e_j \in E(v)$ such a dependence would take on the following form:

$\forall$ $v \in V$: if (e_i, e_j), $i\neq j$, e_i, $e_j \in E(v)$ is a pair of edges used to enter and exit node v at time step t

and $P^*(e_i, e_j) = true$,

then $\forall$ $c_k \in C(v, e_l)$, $P^*(e_i, e_l) = true$, $l \neq j$

set 1) $s_t(c_k) = 0$; 2) $q_t(c_k) = h(a)$.

VI. RULES OF AGENT MOVE AND STATE TRANSITION

RA are the rules of agents' movements and state transitions depending on the states of passed nodes. They define how agents move through the coordination space and how their states change depending on the state of passed nodes. In general, RA consists of five rules: RA = {RA1, RA2, RA3, RA4, RA5}.

RA1 is a "scattered start" rule, that sets agents' unique starting positions:

$\forall(a_i, a_j)$, $a_i, a_j \in A$, $i\neq j$: $x_0(a_i) \neq x_0(a_j)$.

RA2 is a "move forward" rule, that forbids the agent to move back along the traversed path:

$x_{t+1} \neq x_{t-1}$, $x_{t-1} \notin I(x_t)$.

RA3 is a "keep moving" rule that forbids the agent to stay in the same node if there are alternatives available:

if $\exists$ $e_i \in E(x_t, I(x_t))$: $\delta_t(e_i) = 1$,

then $x_{t+1} \neq x_t$ & $x_{t+1} \in I(x_t)$

& $\delta_t(e) = 1$ & $e \in E(x_t), E(x_{t+1})$

RA4 is a "mutual exclusion" rule that forbids any two agents passing the same node at the same time to exit in the same direction:

if $\exists$ ai, aj $\in$ A, $i\neq j$, such as $x_t(a_i) = x_t(a_j)$,

then $x_{t+1}(a_i) \neq x_{t+1}(a_j)$.

RA5 is a "trapping" rule that defines conditions when agents are unable to move forward due to a lack of alternatives and conditions when such alternatives become available again. If at a time step t only one agent is at node v (N(v) = 1), then

if $s_t(a) = 1$ & ($\forall$ $e_i \in E(x_t, I(x_t))$: $\delta_t(e_i) = 0$),

then set $s_t(a) = 0$,

if $s_t(a) = 0$ & ($\exists$ $e_i \in E(x_t, I(x_t))$: $\delta_t(e_i) = 1$),

then set $s_t(a) = 1$,

else, if at the time step t more than one agent is at node v (N(v) > 1) and $\Lambda = \Sigma\delta_t(e_i)$ for all $e_i \in E(x_t, I(x_t))$, then

if $\Lambda < N(v)$,

then $\forall$ $a \in A(v)$ set $s_t(a) = 0$

if ($\exists$ $a \in A(v)$: $s_t(a) = 0$) & ($\Lambda \geq N(v)$),

then set $s_t(a) = 1$.

VII. MULTI-AGENT COORDINATION GAMES

Based on the coordination space M and the <RM, RA> rules, various multi-agent coordination games can be implemented that reflect the specifics of solving various problems by a multi-agent system. Here one needs to choose the required structure of the coordination space M, if necessary, refine the rules <RM, RA>, and specify the rules of the coordination game itself. In this case, agents are players playing an implemented coordination game. This ensures the coordination of their joint actions in solving a common task. To do this, the developer needs to map the problems of organizing collective behavior into the goals and rules of a multi-agent coordination game. At the same time, one can follow the path of interpretation. That is, to find necessary aspects of collective behavior in some known multi-agent coordination game.

Multi-agent coordination games can be used to study various aspects of the collective behavior of intelligent agents. The goals of the corresponding experiments can be: 1) evaluation of the efficiency of the architectures of intelligent agents participating in the coordination game; 2) determination of the advantages and disadvantages of the developed algorithms of collective behavior; 3) study of the joint collective behavior of humans and intelligent agents in a coordination game.

To demonstrate the capabilities of the proposed coordination method, we consider four multi-agent coordination games: the weaving game, the pursuit game, the expansion game, and the waiting game. The first three games are considered at the level of a general description and discussion of options. The waiting game is considered in more detail. In all the games considered, the coordination space M is a regular square grid closed in one of the coordinates into a ring. It is also assumed that 1) the player can see the state of only neighboring nodes of the coordination space, 2) the player can make only one transition between nodes in one time step, 3) the player does not know the number of players participating in the game, 4) the player does not know the size of the coordination space.

VIII.THE WEAVING GAME

At the beginning of the game, agents are randomly placed at the nodes of the coordination space M. After that, the agents begin to move from node to node, leaving behind a trail of closed and half-closed nodes. At each time step, each agent must make a move (according to the RA3 rule). If the agent cannot make a move, i.e., gets into a situation where all neighboring nodes are closed, he loses. The agent who falls into the trap last wins. There may be several such agents. If the length of the trail is not limited, then the game is guaranteed to end in a countable number of time steps. On average, the game will last longer, the smaller the ratio of the number of agents to the number of nodes in the coordination space, that is, the more "free space" there is in the coordination space. The fault tolerance of the proposed coordination method for the case of this game was studied. The simulation results showed that the maximum increase in the average time to solve the game due to agent failures for different combinations of values K={100,…,2000}, N = {10, …, 100}, λ={0.001,…,0.02}) was W_T=26.4 %.

IX. THE PURSUIT GAME

In this game, the idea of a changing trail length is used. At the beginning of the game, the agents are randomly placed at the nodes of the coordination space M. After that, they begin to move through the nodes, leaving behind a trail of limited length $h_0(a)$. At the end of each trail there is the agent's pursuer. The pursuer follows the trail, his goal is to catch the agent. If the agent, in the course of his movements, gets into a trap, then the pursuer approaches him by two transitions in one-time step, that is, the length of the trail decreases by two: $h_t(a)=h_{t-1}(a)-2$. If the agent in the course of his movements is forced to cross the trail of another agent, then the pursuer approaches him by one transition, that is, the length of the track decreases by one: $h_t(a)=h_{t-1}(a)-1$. The length of the trail of the agent who owns the crossed trail is increased by one, that is, his pursuer moves away from him by one transition: $h_t(a)=h_{t-1}(a)+1$. The agent caught by his pursuer loses. The winner is the agent whose trail length reaches the given value $h_w(a)$, $h_w(a) > h_0(a)$ earlier than other agents. Some other options for determining the winner: the last agent remaining not caught wins, or the agent with the longest trail after a given number of time steps wins. The fault tolerance of the proposed coordination method for the case of this game was studied. The simulation results showed that the maximum increase in the average time to solve the game due to agent failures for different combinations of values K = {100, …, 2000}, N = {10, …, 100}, λ={0.001,…,0.02}) was W_T=18.2 %.

X. THE EXPANSION GAME

The game is played by two rival teams of agents. The number of agents in each team is the same. At the beginning of the game, agents are randomly placed at the nodes of the outer level of the coordination space M. After that, they begin to move through the nodes, leaving behind a trail of gradually increasing length. The goal of each team is to maximize the number of nodes covered by the trails of the team's agents. The nodes in which the trails of the agents of the rival teams intersect are not taken into account. The team whose trails occupy the largest area of the coordination space wins. The different ways to end the game are as follows: 1) when there are no open nodes left, 2) when all agents fall into a trap, or 3) after a given number of time steps. When an agent falls into a trap, his trail decreases by one at each subsequent time step, which reduces his team's payoff. Thus, each team needs its members to get trapped as little as possible, and members of the opposing team to get trapped as often as possible. The fault tolerance of the proposed coordination method for the case of this game was studied. The simulation results showed that the maximum increase in the average time to solve the game due to agent failures for different combinations of values K = {100,…,2000}, N_1=N_2={10,…,100}, λ={0.001,…,0.02}) was W_T=27.2 %.

XI. THE WAITING GAME

The waiting game resembles games of timing [10, 11] while having the main features of classical maze problems. At the beginning of the game, N agents are randomly placed at the nodes of the outer level of the coordination space M. At each subsequent time step, an agent must make a move, that is, go to one of the neighboring open nodes. Moving in the coordination space, an agent leaves a trail of closed and half-closed nodes. The length of the agent's trail is not limited and grows with each time step by one. If during the movement the agent falls into a trap, then he loses. The agent's goal here is to keep moving for as long as possible in the coordination space, that is, to make as many transitions between nodes as possible and not get trapped.

At any time, an agent may decide to leave the coordination space and move into its inner area. The agent who makes this decision becomes a contender for victory. As soon as any other agent is trapped, the contender moves closer to the center of the inner area. Thus, the more other agents are trapped, the closer the contender approaches the center and his victory. However, if a new contender, that has moved longer in the coordination space, appears in the inner area, then all the previous contenders lose. The game is won by the last contender, after which no other agent could enter the inner area.

The game ends when no moving agents remain at the nodes of the coordination space, that is when all agents either become contenders or fall into a trap. At this moment, the last contender or contenders, if there are several of them, reaches the center of the inner area and wins the game (Fig.2, Fig.3).

The waiting game has two components. At the level of tactics, the agent is busy finding his way through the maze of trails of other agents. At the level of strategy, the agent is busy choosing the moment in time when to become a contender. By becoming a contender too early, the agent risks losing new contenders. Staying in the coordination space for too long the agent risks being trapped as the maze of trails becomes more complex with each time step. Note that the interaction of these two components of the game is of particular interest for the development of collective behavior algorithms.

To demonstrate basic player logic in the waiting game let us examine a simple artificial player algorithm. The algorithm is based on a simple principle: if with each next move it becomes more difficult for the player to find his way in the maze of trails, then this stimulates him to become a contender, that is, to move into the inner area. To implement this principle, let's introduce a player's confidence level c_t. The higher the c_t, the less the player tends to become a contender, and the lower the c_t, the more the player is afraid of being trapped and tends to enter the inner area by becoming a contender. Due to this, two levels of decision-making interact: tactical (maze task) and strategic (game of timing).

Given the confidence level and the fact that the player only sees the state of neighboring nodes, the decision-making rules of the simple artificial player are as follows.

1. Do not select closed nodes.

2. Choose open nodes with greater probability than half-closed ones.

3. Choose with greater probability open nodes in which there are no other players.

4. If the number of alternatives has decreased compared to the previous time step, then lower the confidence level is.

5. The lower the level of confidence, the more likely it is to choose those nodes that lead the player to the inner area.

In this case, one needs to choose 1) a method for calculating the probabilities of choosing a neighboring node for a transition (a question of tactics), 2) a method for changing the player’s confidence level c_t based on the information available to him about the game situation (a question of strategy).

In the proposed algorithm, the player's tactics are implemented based on the idea of a stochastic learning automaton. Accordingly, the probabilities of choosing neighboring nodes change as follows

$p_{i,t+1} = p_{i,t} - \Delta$,

$p_{j,t+1} = p_{j,t} + \Delta/(n - 1), i\neq j$,

where p_i is the probability of choosing the i-th unfavorable node (a half-closed node or a node in which another agent is located), $\{p_j\}$ are the probabilities of choosing all other neighboring nodes, Δ is the step of changing the probabilities, n is the number of neighboring nodes available for transition.

As a simple strategy, a linear decrease in the confidence level depending on the decrease in the number of neighboring nodes available for transition is chosen. At each time step, the level of confidence is calculated according to the following rules:

if $d_t(e_i) < d_{t-1}(e_i)$, then $c_t = c_{t-1} - (d_{t-1}(e_i) - d_t(e_i))$,

if $d_t(e_i) = d_{t-1}(e_i)$ and $d_t(e_i) < 3$, then $c_t = c_{t-1} - 1$,

$\Delta_{3,t} = \Delta_{3,t-1} + 1 / c_t$,

where $d_{t-1}(e_i)$ is the number of alternatives available to the player at the previous time step, $d_t(e_i)$ is the number of alternatives available to the player at the current time step, c_t – is the player's current level of confidence, Δ_3 is the step of decreasing the probabilities of choosing those nodes that do not bring the player closer to the inner area.

According to these rules, the level of confidence decreases in proportion to the decrease in the number of available alternatives. Otherwise, the player checks how many alternatives are available to him. If the number of alternatives is less than three and this situation repeats, then the level of confidence decreases by one.

It should be emphasized that the method of changing the level of confidence is the key to develop a strong artificial player. This method can be more complex and take into account more aspects of the current game situation. For example, there may be an increase in the level of confidence when the player enters a trail-free region of the coordination space.

Thus, a simple artificial player algorithm is as follows. At the beginning of the game, the player is assigned a starting confidence level c_0. Then, at each time step, the player does the following:

1. Determine available alternatives based on the state of neighboring nodes.

2. Remove closed nodes from the list of alternatives.

3. If the list of alternatives is empty, then admit defeat, otherwise, go to step 4.

4. Determine the nodes from the list, where other players are located, and reduce the probabilities of choosing these nodes by Δ_1.

5. Determine which nodes from the list of alternatives are half-closed and reduce the probabilities of choosing these nodes by Δ_2.

6. Change the confidence level c_t according to the given rules.

7. Change the value of $\Delta_{3,t}$ according to the new confidence level c_t.

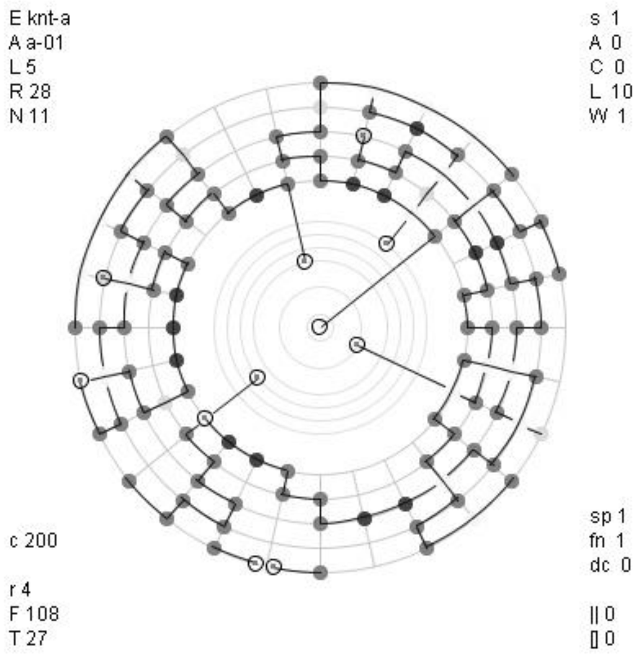

Fig. 2. An example of the end of a waiting game with one winner (N=11, K=140).

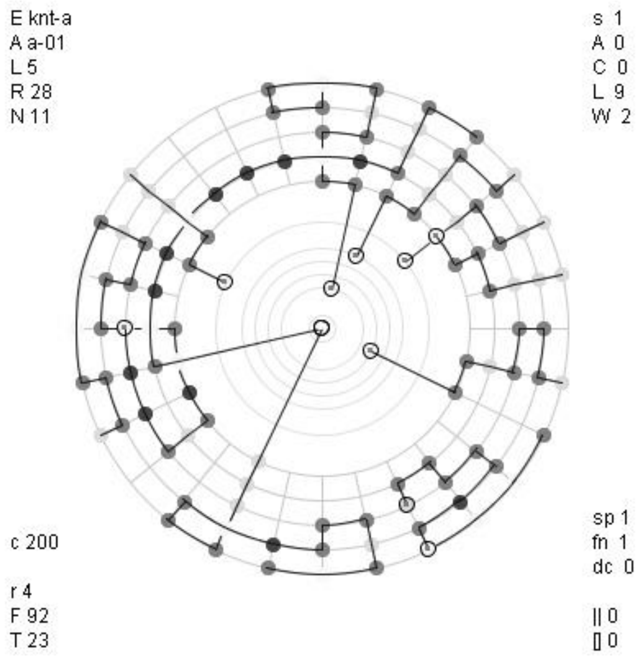

Fig. 3. An example of the end of a waiting game with two winners (N=11, K=140).

8. Determine which nodes do not bring the player closer to the inner area and reduce the probabilities of choosing these nodes by $\Delta_{3,t}$.

9. Using the resulting probability distribution, randomly select a node for the transition.

10. Move to the selected node.

The fault tolerance of the proposed coordination method for the case of this game was studied. The simulation results showed that the maximum increase in the average time to solve the game due to agent failures for different combinations of values K={100,…,2000}, N={10,…,100}, λ={0.001,…,0.02}) was W_T=21.9 %.

XII.CONCLUSION

The article proposed a method of multi-agent coordination with deferred asynchronous messaging in a distributed coordination space. The method implemented the idea of preserving and using the history of inter-agent interactions in an explicit convenient form, which made it possible to organize the spontaneous emergence of coordinated collective behavior. The method was based on the concept of multi-agent conditional interaction. The method used 1) a distributed coordination space in which agents move, 2) the rules of state transitions for the coordination space nodes depending on the movements of agents, 3) the rules of agents move and state transitions depending on the states of the coordination space nodes, 4) a multi-agent coordination game based on the coordination space and the rules.

The coordination space was implemented based on the distributed shared memory of agents. The rules were applied by exchanging deferred asynchronous messages between agents through the distributed shared memory. The agent's decisions about movement in the coordination space and their consequences were interpreted according to the rules in asynchronous messages. Delivery of messages to other agents had been deferred until these agents visited the corresponding nodes of the coordination space. This ensured 1) mutual exclusion when agents choose conflicting actions, and 2) resilience of multi-agent coordination to agent failures and loss of coordinating messages.

The proposed coordination method met the above requirements in the following way.

1. The emerging structure of cause-and-effect relationships is mapped into a dynamic system of agents' trails in the coordination space. The actions of some agents cause general restrictions on the actions that other agents may choose in the future. By changing the length of the trail, it becomes possible to take into account and control the cause-and-effect relationships that arise in the course of inter-agent interactions.

2. The agent's level of awareness about the general situation is regulated by the number of nodes in the coordination space, the state of which is known to the agent. That is, the agent is more informed, the more neighboring nodes it can see around him. The agent's freedom of action is regulated by the number of nodes that it can pass in one-time step.

3. Wide range of meaningful interpretations of the processes of inter-agent interactions is based on different ways of interpreting the elementary choices that agents make in the course of coordination. Depending on how the nodes of the coordination space and the edges between them are interpreted, different ways for using the

proposed multi-agent coordination method in practical problems can be taken.

4. The proposed approach makes it possible for humans to participate in multi-agent coordination. Of great interest here is the competition between a team of humans and a team of intelligent agents in multi-agent coordination games. A human can act 1) as an experimenter, exploring the abilities of intelligent agents; 2) as an agent in scenarios for coordinating the joint actions of humans and intelligent agents.

5. The history of the process of inter-agent interactions is represented as a collection of agents' trails in the coordination space and the states of coordination space nodes. This allows to see a complete picture of multi-agent coordination unfolded in time with both the history and the current state of the process of inter-agent interaction directly presented.

As examples, four multi-agent coordination games (the weaving game, the pursuit game, the expansion game, and the waiting game) were considered. The waiting game was considered in more detail. A simple artificial player algorithm was proposed for this game.

The issue of fault tolerance of the proposed coordination method was considered. The simulation results showed that the use of the method ensured the resilience of multi-agent coordination to agent failures in the considered coordination games. In particular, the use of the proposed coordination method limited the increase in the average solution time of a coordination game due to agent failures to 30 % for the intensity of the stationary Poisson process of agents' failures in the range from 0.001 to 0.02.

References

[1] Dorri, A., Kanhere, S., Jurdak, R. (2018). Multi-Agent Systems: A Survey, in IEEE Access, vol. 6. – pp. 28573-28593, DOI: 10.1109/ACCESS.2018.2831228.

[2] Rizk, Y., Awad, M., Tunstel, E. (2018). Decision Making in Multi-Agent Systems: A Survey, in IEEE Transactions on Cognitive and Developmental Systems, vol. 10, no. 3. – pp. 514-529, DOI: 10.1109/TCDS.2018.2840971.

[3] Sun, Z. (2018). Cooperative Coordination and Formation Control for Multi-agent Systems. Springer Cham. – 179 p. DOI: 10.1007/978-3-319-74265-6.

[4] Arup Kumar Sadhu, Amit Konar (2020). Multi-Agent Coordination: A Reinforcement Learning Approach, Wiley. – 320 p.

[5] . Binyamin, S., Ben Slama, S. (2022). Multi-Agent Systems for Resource Allocation and Scheduling in a Smart Grid. Sensors, 22, 8099. – pp. 1-40. DOI: 10.3390/s22218099.

[6] Nezamoddini, N., Gholami, A. (2022). A Survey of Adaptive Multi-Agent Networks and Their Applications in Smart Cities. Smart Cities, 5(1). – pp. 318-346. DOI: 10.3390/smartcities5010019.

[7] Zhou, L., Zheng, Y., Zhao, Q., Xiao, F., Zhang, Y. (2022). Game-based coordination control of multi-agent systems. Systems & Control Letters. 169. – pp. 1-24. 105376. DOI: 10.1016/j.sysconle.2022.105376.

[8] Paccagnan, D., Chandan, R., Marden, J. (2022). Utility and mechanism design in multi-agent systems: An overview. Annual Reviews in Control. 53. – pp. 315-328. DOI: 10.1016/j.arcontrol.2022.02.002.

[9] Jin, B., Cao, M. (2022). Control using Q-learning for networked coordination games, 2022 13th Asian Control Conference (ASCC). – pp. 941-946. DOI: 10.23919/ASCC56756.2022.9828324.

[10] Merlevede, J., Johnson, B., Grossklags, J., Holvoet, T. (2019). Time-Dependent Strategies in Games of Timing. In: Decision and Game Theory for Security. GameSec 2019. Lecture Notes in Computer Science, vol. 11836. Springer, Cham. – pp. 310-330. DOI: 10.1007/978-3-030-32430-8_19.

[11] Smirnov, V., Wait, A. (2022). General Timing Games with Multiple Players, (April 22, 2022). – 34 p. Available at SSRN: https://ssrn.com/abstract=4090339, DOI: 10.2139/ssrn.4090339.

[12] Merlevede, J., Johnson, B., Grossklags, J., Holvoet, T. (2021). Exponential discounting in security games of timing. Journal of Cybersecurity, Volume 7, Issue 1. – pp. 1-20. DOI: 10.1093/cybsec/tyaa008.

Alexey Botchkaryov was born in 1975 in Lviv, Ukraine. He received B.S. and M.S. degrees in Computer Engineering at Lviv Polytechnic National University, in 1998 and a Ph.D. degree in Computer Systems and Components at Lviv Polytechnic National University in 2019. He has been doing scientific and research work since 1994. Currently, he is an associate professor at the Computer Engineering Department, ~~at~~ Lviv Polytechnic National University. His research interests include self-organization in complex systems, structural adaptation, intelligent information-gathering agents, and multi-agent systems.