Formal Models of Question-Answering Machine

Igor A. Chimir[a], and Anatolii F. Verlan[b]

[a] *Odessa State Environmental University, 15 Lvovskaya St., 65016, Odessa, Ukraine*
[b] *The Georgy Puchov Institute for Energy Modeling. National Academy of Science, 15 General Naumov St., 03164, Kiev, Ukraine*

Abstract

The article describes two models of question-answering dialogue machine: (1) model based on the idea of Mealy finite automata, and (2) model based on the idea of Petri net. Both models are problem-independent and describe question-answering dialogue process, which is independent of the subject area of the dialogue. The problem independence of the models is a consequence of a unified cognitive cycle of dialogue used in them. The unified cognitive cycle of dialogue is similar to Neisser's cyclical model of perception. Models are designed to specify a "dialogue machine" that simulates a goal-oriented behavior of the active dialogue agent when solving a problem by means of a question-answering dialogue. Implementation of the models presupposes data-driven approach when main components of the "dialogue machine" are not map in the program code but represented by data stored in the database.

Keywords 1

Question-answering dialogue, Dialogue agent, Dialogue script, Cognitive cycle of dialogue.

1. Introduction

Dialogue communication is one of the phenomena of human mental activity and is a subject of study in such sciences as artificial intelligence, cognitive psychology, erotetic logic and epistemology.

The Turing intelligence test is based on a dialogue process between a human and a technical system [1]. Modern presentation of artificial intelligence is based on the concept of intelligent agents [2]. Dialogue communication between the environment and the intelligent agent determine the behavior of the latter.

In cognitive psychology, one can find theories and models that contribute to understanding the essence of dialogue. Neisser's model of perception is a model of the dialogue process if the source of the flow of sensory events is one of the dialogue partners [3].

The vehicle for the exchange of knowledge between dialogue partners is a question-answering pair, therefore erotetic logic provides an apparatus for modeling the logical structure of a dialogue [4].

Epistemology is related to understanding the essence of the dialogue process, since it studies those types of knowledge that circulate within the question-answer pair. Dialogue plays a key role in Socratic epistemology [5].

Question-answering dialogue is one of the types of dialogue that can be operated by artificial dialogue agents. In a question-answering dialogue, the information messages of the active agent represent questions, and the information messages of the reactive agent represent the answers to these questions. The concepts of "question" and "answer" should be understood not in the narrow, linguistic sense of the word, but in the broad, behavioral sense. The question is, in the general case, the interrogative stimulus of the active agent, and the answer is the reaction of the reactive agent.

A feature of the question-answering dialogue is its ability to serve as a means of solving a number of ill-formalized problems. Newell was the first to draw attention to the need to develop methods for

CMIS-2021: The Fourth International Workshop on Computer Modeling and Intelligent Systems, April 27, 2021, Zaporizhzhia, Ukraine
EMAIL: chimir@gmail.com (I. Chimir); a.f.verlan@gmail.com (A. Verlan)
ORCID: 0000-0001-5326-768X (I. Chimir); 0000-0002-8565-2037 (A. Verlan)

solving ill-formalized problems [6]. Newell believed that ill-formalized problems are those that had one or more of the following characteristics: problems cannot be specified numerically; goals cannot be expressed in terms of well-defined objective functions; there is no algorithmic solution to the problem; an algorithmic solution exists, but it cannot be used due to limited computer resources. In applied systems of artificial intelligence, there are systems oriented on solving ill-formalized problems. One of the most extensive classes of such systems is intelligent tutoring systems [7]. The problem of teaching and knowledge transfer is ill-formalized and can be solved by dialogue methods.

The pragmatic dimension in the modeling of question-answering dialogue processes is aimed at designing systems focused on solving ill-formalized problems and systems called "chatbots" [8].

In the field of theory and practice of dialogue processes, focused on solving ill-formalized problems, an important question is how to create, store and repeatedly reproduce dialogue methods. In a goal-oriented dialogue process, in which the solution of an ill-formalized problem is carried out, the active agent plays a key role, since it is the active agent who is the carrier of the method for solving the problem. The article is devoted to the synthesis and description of cognitive and formal problem-independent dialogue models that can generate protocols of goal-oriented question-answering dialogues. Formal models have applied value and can be used as a basis for the development of artificial dialogue agents.

2. Cognitive cycle of question-answering dialogue

When developing a dialogue model, it is important that the model is adequate to the processes of perception and processing of information by a human. In the case when the basis of the formal model of dialogue is an adequate psychological model, then we can expect that the artificial dialogue agent inherits the anthropomorphic properties of the system of perception of a human.

Among the theories and models proposed by cognitive psychology and related to the dialogue process, let us dwell on the models that describe dialogue at the level of perception, which does not depend on the subject area of the dialogue and on the nature of the problem, which is solved in the dialogue process. In this sense, a psychological model of dialogue, useful in an applied aspect, should describe some unified cognitive "dialogue machine".

The dialogue process is similar to the process of perceptual human interaction with the environment, modeled by the Neisser cycle [3]. The difference lies in the fact that in the process of dialogue, the main components of the human sensory system - sight and hearing - are connected not to the "natural" environment, but to the "artificial" environment formed by streams of visual and sound sensory events generated by the opposite agent of the dialogue. Thus, in the dialogue process, the real environment is replaced by an artificial one. However, it is obvious that the perception and subsequent processing of both sensory events generated by the artificial environment (dialogue agent) and sensory events generated by the natural environment are carried out according to the same "rules and laws". A diagram illustrating the cognitive cycle of a question-answering dialogue is shown in Figure 1.

The cognitive cycle of the dialogue process, shown in Figure 1, simulates the behavior of an active agent in a dialogue with fixed roles. Fixing agent roles means prohibiting changing agent roles during a conversation. One of the agents always asks questions, and the other always answers them. For convenience, we will assume that the dialogue cycle begins from the moment when the active agent operates with a relatively small set of answers, called the set of expected answers, and the real answer of the reactive agent coincides with one of the answers from the set of expected answers of the active agent. Thus, one of the main cognitive structures that the active agent operates is the schema-answer, or mental representation of the answer. The set of expected answers is embedded in a broader cognitive framework called the active agent script. It is assumed that the script of the active agent includes all the schema-answers necessary for the given dialogue, and that the set of expected answers is some subset of answers from the script that is relevant to the current cycle of the dialogue process. After the perception and recognition of the real answer received from the reactive agent, the current set of expected answers is modified, the purpose of which is to prepare the set of expected answers for the next cycle. As a result of the modification, a new set of expected answers is formed, including the answer expected on the next cycle. During the modification process, the current set of expected answers can be replaced in whole or in part.

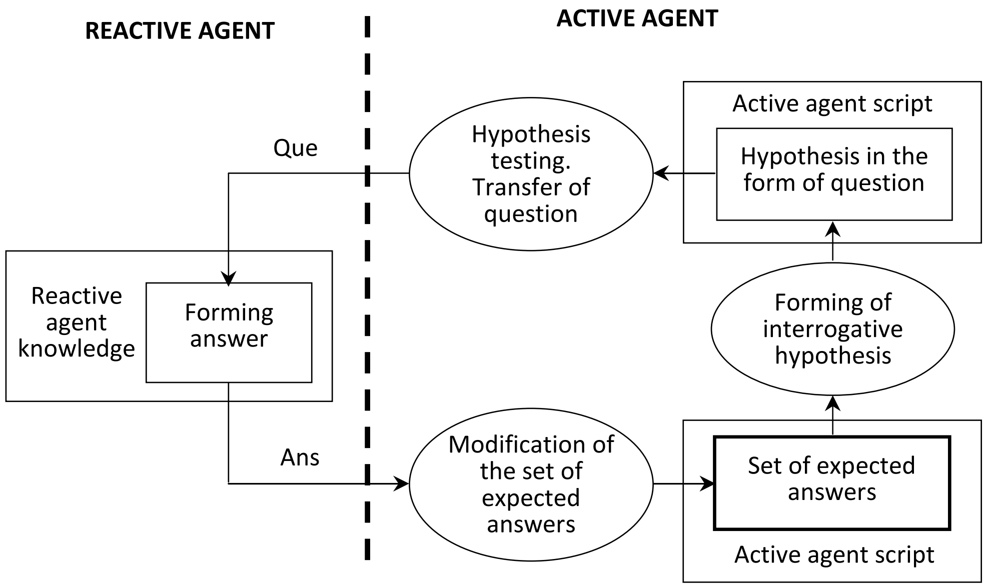
REACTIVE AGENT
ACTIVE AGENT
Active agent script
Hypothesis in the form of question
Que
Hypothesis testing. Transfer of question
Reactive agent knowledge
Forming answer
Forming of interrogative hypothesis
Ans
Modification of the set of expected answers
Set of expected answers
Active agent script

Figure 1: Cognitive cycle of a question-answering dialogue process

The concept of a dialogue script means that the active agent does not generate new question using some "super-algorithm", but searches for it in the memory of questions, using the method of achieving the goal of the dialogue as a method of accessing the memory of questions. Thus, the method of accessing the memory of questions is a repository of the method for achieving the goal of a dialogue, or a dialogue method for solving a problem. The concepts "memory of questions" and "dialogue method of solving a problem" are included in the scope of the concept of "script" of an active agent and detail its structure.

Introspection and analysis of real protocols of question-answering dialogue processes reveals that there is an ambiguous relationship between the current answer of the reactive agent and the subsequent question generated by the active agent. In other words, in different dialogue transactions for the same answer received from the reactive agent, the active agent can generate different questions. Therefore, when developing dialogue methods for solving problems, for modeling the noted ambiguity, it is necessary to take into account the following three principles.

The principle of "depth of dialogue". The principle of "depth of dialogue" means that the active agent, when forming the next question, must take into account both the perceived answer and the index of the dialogue cycle. Here, the term "index" is used synonymously with the term name, or identifier. In different dialogue cycles, different questions can be generated for the same answer received from the reactive agent.

The principle of "history of answers". The principle of "history of answers" means that the active agent, when forming the next question, must take into account both the perceived answer and the history of the received answers. In the same dialogue cycle, different questions can be generated for the same answer received from the reactive agent, depending on what answers were received in the previous cycles.

The principle of "history of questions". The principle of “history of questions” means that the active agent, when forming the next question, must take into account both the perceived answer and the history of previously formed questions. In the same dialogue cycle, different questions can be formed for the same answer, depending on what questions were formed in the previous cycles.

The dialogue cycle shown in Fig. 1 describes well the “harmonious dialogue” corresponding to the Neisser cycle for the case of routine perception. By harmonious dialogue we mean such a dialogue when both agents are satisfied with their roles and do not want to change them. However, a harmonious dialogue is not always possible. The initiator of the role change is usually the reactive agent, and the sign of the moment of the role change is the generation and transmission to the active agent of an information message with the status of a question. Therefore, one of the ways to take into account the possibility of changing roles can be to include a question detector in the list of expected answers. A question detector can be included in each set of expected answers (then agents will be able to change roles in any dialog cycle) or only in some sets of expected answers (then the roles of agents can be changed only in some predefined dialogue cycles).

Considered, further, formal problem-independent models of dialogue are models of the dialogue method of accessing the memory of questions, considered also as a dialogue method for solving a problem. These models model the behavior of an active agent of a question-answering dialogue, as the functioning of some "dialogue machine". The purpose of modeling is to identify the components of the "dialogue machine" that is invariant to the problem being solved and to the subject area of the dialogue process.

3. Finite-automaton model

The finite-automaton model of question-answering dialogue simulates a harmonious dialogue and is based on three assumptions: (1) the problem solved during the question-answering dialogue can be solved in a finite number of dialogue cycles (dialogue steps); (2) by the beginning of solving the problem, all questions and answers necessary for the formation of dialogue steps are determined; (3) modification of the current set of expected answers always generates a new set that is different from the current one and has a unique index.

The finite-automaton model is a variant of the Mealy automaton [9] and is described by the following formula

$$\text{DiAM} = (\text{QUE}, \text{ANS}, \text{S}, \varphi) \tag{1}$$

where DiAM denotes a dialogue access method to the memory of questions.

$$\text{QUE} = \{\text{Que}_i\};\ i = 1,\ldots,k \tag{2}$$

QUE – a set of indexes of questions used to solve a problem.

$$\text{ANS} = \{\text{Ans}_i\};\ i = 1,\ldots,l \tag{3}$$

ANS – a set of answers used to solve a problem.

$$\text{S} = \{\text{S}_i\};\ i = 1,\ldots,m \tag{4}$$

S – a set of states of waiting for an answer. The number of elements of the set S is equal to the total number of dialogue steps.

$$\varphi : \text{S X ANS} \rightarrow \text{S X QUE} \tag{5}$$

φ – step function that determines the index of the new question and the index of the next step, depending on the current answer and the index of the current step.

When performing a step, an automaton that simulates a question-answering dialogue process performs iterations consisting of the following sequence of actions: (1) perceived the current answer of the reactive agent; (2) the current answer of the reactive agent is recognized; (3) the index of the next question of the active agent is determined.

If the set of answers ANS is not structured in any way, this means that in order to recognize the current answer of the reactive agent, the dialog access method must, at each step, operate with the entire set of answers ANS. For large scripts, the cardinality of this set can be significant. In order to reduce the cardinality of the set of answers that DiAM operates with at each step, and to make the model practically realizable, we introduce into the model a set of answers recognizable at the i-th step. The set of recognizable answers contains only those answers that are expected at the i-th step, are necessary for the implementation of the dialogue method and, therefore, must be recognized. All other answers will be referred to the class of answers that are not recognizable at the i-th step. We will use the following notation: R^i – is the set of answers recognizable at the i-th step; NR^i – is the set of answers not recognizable at the i-th step.

The set of answers that are not recognizable at the i-th step includes all answers belonging to the set ANS and not belonging to the set R^i, i.e.

$$NR^i = ANS - R^i \tag{6}$$

However, the set of answers that are not recognizable at the i-th step must be considered wider and understood as the set of any conceivable answers of a reactive agent that do not belong to the set R^i, i.e.

$$NR^i = U - R^i \tag{7}$$

where U – is the set of any responses that the reactive dialogue agent can ever generate.

It is natural to assume that the appearance at the i-th step of any answer belonging to NR^i leads to the fact that DiAM generates the same index of the next question. Otherwise, it would mean that the answers from the NR^i belong to the class of recognizable answers. Therefore, in the step function φ for each step of the dialogue, the entire class of unrecognizable answers will be modeled with one unrecognizable answer.

The above reasoning allows us to return to the cycle in Figure. 1, and clarify the structure of the set of expected responses. At each step of the dialogue, the set of answers that the active agent expects can include responses from the following three classes: (1) the class of recognizable answers; (2) the class of unrecognizable answers (modeled by one answer); (3) the class of question detectors (to determine the need for a role reversal).

Let us consider an example that will allow, firstly, to assess how useful the finite-automaton model of the question-answer dialogue is when constructing a dialogue access method to the memory of questions, and, secondly, will provide a transition to the subsequent model in the form of a Petri net. The example includes several steps.

Step 1. The active agent generates a question with index Que_1 and expects to receive the following answers:

Ans_1 – request for a hard copy of a question;

Ans_3 – answer followed by a question with an index Que_3;

Ans_4 – answer followed by a question with an index Que_4;

Ans_5 – unrecognizable answer (any answer other than Ans_1, Ans_3, Ans_4). Answer Ans_5 means that it is necessary to generate a question with Que_2 index.

Step 2. The active agent generates a question with index Que_2, which is a reformulated question with index Que_1. In this step, the active agent expects to receive the same answers as in the first step, as well as an Ans_2 answer, which is a request to return to step 1.

DiAM is described using the following formulas

$$ANS = \{Ans_1, Ans_2, Ans_3, Ans_4, Ans_5\} \tag{8}$$

$$QUE = \{Que_1, Que_2, Que_3, Que_4\} \tag{9}$$

$$S = \{S_1, S_2, S_3, S_4\} \tag{10}$$

$$\left.\begin{aligned} \varphi : (S_1, Ans_1) &\rightarrow (S_1, Que_1), \\ (S_1, Ans_3) &\rightarrow (S_3, Que_3), \\ (S_1, Ans_4) &\rightarrow (S_4, Que_4), \\ (S_1, Ans_5) &\rightarrow (S_1, Que_2), \\ (S_2, Ans_1) &\rightarrow (S_2, Que_2), \\ (S_2, Ans_2) &\rightarrow (S_1, Que_1), \\ (S_2, Ans_3) &\rightarrow (S_3, Que_3), \\ (S_2, Ans_4) &\rightarrow (S_4, Que_4), \\ (S_1, Ans_5) &\rightarrow (S_2, Que_2), \end{aligned}\right\} \tag{11}$$

The step function φ, represented by mappings (11), on the one hand, describes the “logic” of access to the memory of questions, and on the other, the “logic” of the dialogue method for solving a certain problem. The step function can be presented in tabular form. The tabular presentation of the step function is convenient in that it allows you to easily switch to a graphical representation of the question-answering dialogue in the form of a state diagram. An example of a graphical representation of a question-answering dialogue in the form of a state diagram is shown in figure 2.

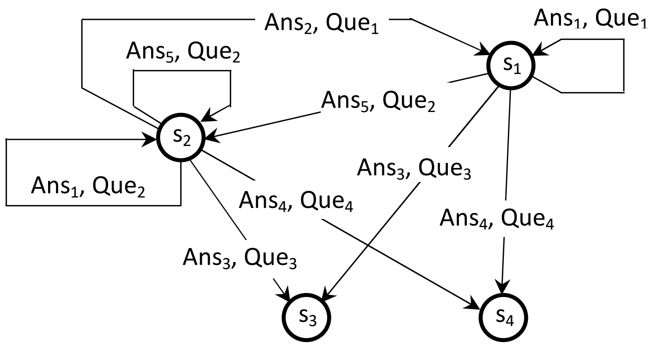

Figure 2: Question-answering dialogue state diagram for the example described in the text

In the state diagram shown in Figure 2, the set of vertices corresponds to the set of steps of the question-answering dialogue (or to the set of stable states), and each arc corresponds to one of the possible options for the development of the dialogue, determined by the answer of the reactive agent. Arcs are marked by pairs Ans_i, Que_i. Where Ans_i – one of the expected answers; Que_i – question index.

As noted earlier, the step function can be considered as a way of describing the "logic" of the dialogue problem solving method, and in this sense, a state diagram, an example of which is shown in Figure 2 can be viewed as a diagram describing the behavior of an active agent in the process of solving a problem.

4. Model in the form of a Petri net

The finite-automaton model of question-answering dialogue has its own sphere of applicability and can be used to specify a number of applied dialogue systems. However, the finite-automaton model has disadvantages that appear at the stage of its computer implementation.

An effective approach to the computer implementation of question-answer dialog systems is the datalogical approach, which, if applied to the design of the dialog access method DiAM, involves the implementation of the step function not in the form of a program code, but in the form of a mapping into a certain database. A decisive advantage of the datalogical approach is the ability to easily modify the dialogue method using full-screen editing tools. Thus, the datalogical approach excludes the process of compiling the source code when modifying the system and allows the dialogue method to be modified by its author.

From the standpoint of the datalogical approach, the formalism adopted to describe the dialogue method must be identified with the conceptual model of some database. As noted in [10], the problem of the conceptual database model is associated with such a representation of the model, which, on the one hand, most naturally reflects the subject area (in our case, the dialogue method of accessing the memory of questions), and, on the other hand, can be supported by computer means. In other words, a formalism is needed that can be easily transformed into a data schema. The experience of designing dialogue applications allows us to conclude that the noted property is inherent not in the state diagram of the dialogue access method (an example of which is shown in Figure 2), but in the graph modeling the dialogue access method in the form of a Petri net [11].

The Petri model of the dialogue method, like the automaton model, is based on three assumptions: (1) the problem solved in the dialogue process can be solved in a finite number of dialogue steps; (2) by the beginning of the solution of the problem, all questions and answers necessary for its solution are determined; (3) modification of the current set of expected answers of the reactive agent, in any case, generates a set of answers that is different from the current one and, therefore, has a unique index. We describe the Petri-model of the dialogue method with the following four:

$$\mathrm{DiAM} = (\mathrm{QUE}, \mathrm{ANS}, \mathrm{NextQue}, \mathrm{NextAns}) \qquad (12)$$

$$\mathrm{QUE} = \{\mathrm{Que}_i\};\ i = 1,\ldots,n \qquad (13)$$

where QUE – a collection of question indices made up of elements of the set (2), which allows the presence of several instances of the same element [11].

$$\mathrm{ANS} = \{\mathrm{Ans}_i\};\ i = 1,\ldots,p \qquad (14)$$

ANS – a collection of answers made up of elements of the set (3).

$$\mathrm{NextQue} : \mathrm{ANS} \rightarrow \mathrm{QUE} \qquad (15)$$

NextQue – function of subsequent question indices. Since the active agent generates only one question at each step, NextQue determines the index of a single question for each answer.

$$\mathrm{NextAns} : \mathrm{QUE} \rightarrow \mathrm{ANS} \qquad (16)$$

NextAns – function of subsequent answers, which map the collection of question indices into the collection of answers.

To illustrate the Petri-model, let us present the example, described earlier, by formulas (12) – (16).

$$\mathrm{ANS} = \{\mathrm{Ans}_1, \mathrm{Ans}_2, \mathrm{Ans}_3, \mathrm{Ans}_4, \mathrm{Ans}_5\} \qquad (17)$$

$$\mathrm{QUE} = \{\mathrm{Que}_1, \mathrm{Que}_2, \mathrm{Que}_3, \mathrm{Que}_4\} \qquad (18)$$

$$\left.\begin{aligned}
&\mathrm{NextAns}(\mathrm{Que}_1) = \{\mathrm{Ans}_1, \mathrm{Ans}_3, \mathrm{Ans}_4, \mathrm{Ans}_5\}\\
&\mathrm{NextAns}(\mathrm{Que}_2) = \{\mathrm{Ans}_1, \mathrm{Ans}_2, \mathrm{Ans}_3, \mathrm{Ans}_4, \mathrm{Ans}_5\}\\
&\mathrm{NextAns}(\mathrm{Que}_3) = \{\}\\
&\mathrm{NextAns}(\mathrm{Que}_4) = \{\}
\end{aligned}\right\} \qquad (19)$$

$$\left.\begin{aligned}
&\mathrm{NextQue}(\mathrm{Ans}_1) = \{\mathrm{Que}_1\}\\
&\mathrm{NextQue}(\mathrm{Ans}_2) = \{\mathrm{Que}_1\}\\
&\mathrm{NextQue}(\mathrm{Ans}_3) = \{\mathrm{Que}_3\}\\
&\mathrm{NextQue}(\mathrm{Ans}_4) = \{\mathrm{Que}_4\}\\
&\mathrm{NextQue}(\mathrm{Ans}_5) = \{\mathrm{Que}_2\}
\end{aligned}\right\} \qquad (20)$$

Formulas (19) and (20) can be used to define NextAns and NextQue functions in the form of tables. The right-hand side of the i-th line of the NextAns function specifies the set of reactive agent answers expected at the i-th step.

$$\mathrm{NextAns}(\mathrm{Que}_i) = R^i \cup NR^i \qquad (21)$$

The right-hand side of the i-th line of the NextQue function consists of one element of the collection Que, which is a consequence of the limitation inherent in the question-answering dialogue, which is that for each answer from the set of expected answers, the function of indexes of subsequent questions determines the index of one single question.

In the graphical interpretation of the Petri-model of the dialogue method, the collection of indexes of the questions Que and the collections of answers Ans are represented by sets of places (circles) and transitions (rectangles), respectively. The set of answers $R^i \cup NR^i$ expected at the i-th step is represented by a set of transitions that are incident to the i-th place and are connected to it by outgoing arcs.

For the Petri-model, a step is a natural "building block" of a dialogue access method and for the i-th step can be defined as a set

$$Que_i, (R^i \cup NR^i) \tag{22}$$

Figure 3 shows the graph of the Petri-model of the dialogue method, corresponding to the example described in the text.

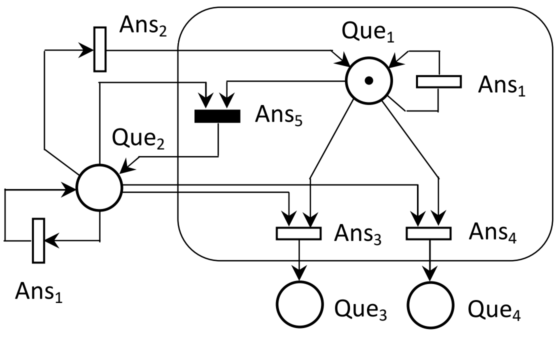

Figure 3: The Petri-model graph for the example described in the text. The black transition simulates all unrecognizable answers

When interpreting the Petri-model graphically, a dialogue step is a place and a set of transitions connected to it by outgoing arcs. In Figure 3 highlights the elements included in the scope of the concept of "dialogue step".

The Petri-model graph is marked with a single token marking the current active step. The initial marking, in this case, should be understood as indicating the first step from which the implementation of the dialogue method begins. The condition for firing a transition is the coincidence of the answer encoded by this transition and the answer received from the reactive agent. In other words, a transition is fired and "skips" a token if an answer is received from the reactive agent, which coincides with the answer encoded by this transition.

The theory of Petri nets provides a convenient apparatus for tracking the dynamics of a simulated process, consisting of marking places and conditions for firing transitions. Since, in the Petri-model of the question-answering dialogue, there is a single token, the marking can be represented by a vector, the number of components of which is equal to the number of elements of the collection of question indices – the number n.

$$\mu = (\mu_1, \mu_2, \ldots \mu_n) \tag{23}$$

Each component of the vector μ takes values on the two-element set {0,1}. $\mu_i = 0$ if the token is absent in the i-th place corresponding to the i-th element of the collection Que. $\mu_i = 1$ if the token is in the i-th place corresponding to the i-th element of the collection Que.

The question-answering dialogue process is accompanied by the movement of the token across the Petri-model without destroying or multiplying it. Therefore, at any moment of the dialogue, $\Sigma\ \mu_i = 1$ takes place. Thus, in the process of a question-answering dialogue, the dialogue method is characterized by a continuously changing vector μ, which indicates the current, active step of the dialogue. For the Petri-model shown in Figure 3, the vector μ indicates step number one and has a value $\mu = (1,0,0,0)$. To move a token to the next place, it is necessary that one of the transitions of the step fires.

Representation of the dialogue method in the form of a Petri net (formulas 12 – 16) models its spatial structure and does not reflect the dynamics of the dialogue process, which is set, in the general case, by the cognitive cycle of the question-answering dialogue process shown in Figure 1. Let us supplement the structural description of the dialogue method with components that simulate the dynamics of the dialogue process. For this purpose, let us consider how, within the framework of the proposed Petri-model, the cycle of the question-answer dialogue process is implemented, based on the assumption that the NextAns and NextQue functions are presented in tables.

It can be assumed that a separate row in the NextAns function table corresponds to one step of the dialogue. Each row starts with the index of the question generated at the given step, followed by a list of answers associated with that step. In this case, the NextAns function table consists of n rows, where n is the number of elements in the collection of question indices. The number of columns in the NextAns function table is determined by the maximum possible number of answers expected from the reactive agent during the dialogue. Since the set of answers associated with a step is individual for each step and is determined by the number of answers expected at a given step, some of the cells in the NextAns table remain empty.

The NextQue function table has a simpler structure. It consists of p rows, where p is the number of elements in the collection of answers. Each row in the NextQue function table contains two elements: the expected answer and the associated follow-up question.

Let's add a unified DiMC process to the model, which simulates the cyclically repeating dynamics of the dialogue

$$\text{DiMC}(\mu, \text{NextAns}, \text{NextQue}) \tag{24}$$

The unified DiMC process controls the transition from the previous step of the dialogue to the next, which is equivalent to firing one of the transitions of the step and moving the token. The unification of the DiMC process means that for the implementation of any step of the dialogue it is necessary to perform the same sequence of actions, and that this sequence of actions does not depend on the subject area of the dialogue process. We describe a separate iteration of the DiMC cycle as follows.

1. The marking vector μ defines the currently active step of the dialogue process, which corresponds to the row number in the NextAns function table.
2. The index of the current question is determined and transferred to the memory of questions. Using the question specifications in the memory of questions, the active agent generates and passes the current question to the reactive agent. The active agent enters the waiting state for the answer of the reactive agent.
3. The active agent perceives the current answer of the reactive agent.
4. The current answer of the reactive agent is processed. The perceived answer is sequentially compared with all the answers expected at this step, stored in the cells of the selected row of the NextAns function table. For a matched answer, a NextQue function table row is determined.
5. Using the table of the NextQue function, the component of the vector μ is searched for, which determines the next step of the dialogue. The value of this component changes from zero to one, and the process returns to item 1.

Among the actions iteratively performed by the unified DiMC process, there are two actions that require more detailed consideration and clarification: (1) generating the current question; (2) processing the current answer. Since DiMC is a unified process that processes any step “uniformly”, both question generation and response processing are handled by DiMC using the same rules for any dialogue step in any dialogue. These rules define a certain standard model for the functioning of an active dialogue agent. In other words, DiMC generates a question and processes the answer in a certain standard way. It is clear that even with a fairly extensive standard, there may be cases where standard tools are not enough. For example, when generating a question, in the case when it is presented non-verbally, it is impossible to take into account all the diversity of non-verbal presentation of information, and when processing the answer, specific numerical processing may be required. Therefore, in addition to the basic DiMC process, it is rational to include in the model “external” processes in relation to DiMC. These processes must be called from DiMC and return control to DiMC when finished. Let's call such processes process-daemons and add two classes of process-daemons to the model

QueDemon and AnsDemon

Process-daemons carry out "additional processing" of a question or answer at any step of the dialogue process in the case when standard DiMC tools are not enough [12].

Let's consider the sphere of applicability of the Petri-model of the question-answering dialogue. By the sphere of applicability we mean the division of problems solved in the dialogue process into two classes: (1) easily implemented within the Petri-model and (2) difficult to implement within the Petri-model.

Problems that are difficult to be implemented within the framework of the Petri-model include, for example, problems that are solved when conducting psycho-diagnostic tests. As a rule, at the first stage of psycho-diagnostic testing (in the process of dialogue interaction with the subject), a primary data set is formed, which is then used to form a conclusion about the intellectual abilities or the structure of the subject's intelligence. An example of a psycho-diagnostic test is the Raven's progressive matrix technique used to test intellectual abilities [13]. The dialogue process that implements the Raven's progressive matrix technique has two specific features. Firstly, at each step of the dialogue, the reactive agent (subject of testing) is presented with stimuli-questions in a non-verbal form. Secondly, the sequence of stimuli-questions transmitted to the reactive agent is formed by the active agent without taking into account the answers of the reactive agent.

We will describe a simple psycho-diagnostic test as follows. The first step generates a Que_1 question that expects one of the answers Ans_1 or Ans_2. In the second step, regardless of the answer received, a Que_2 question is generated, for which one of the answers Ans_3 or Ans_4 is expecting. The conclusion is determined depending on the combination of the answers received. The graph of the Petri-model of the dialog method that implements the described test is shown in Figure 4.

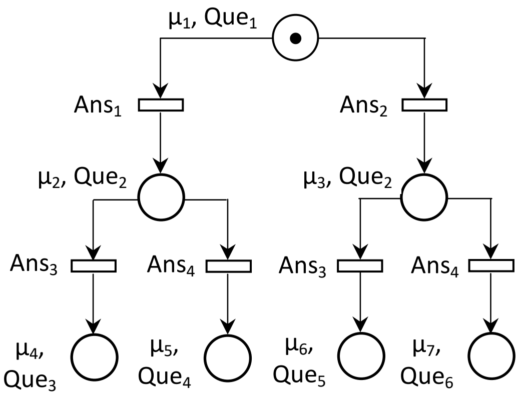

Figure 4: Petri-model of a "difficult" dialogue method

At the bottom of Fig. 4 are target places corresponding to the end of the dialogue and the formation of a conclusion. The difficulty of implementing the Petri-model is manifested in the rapid increase in the number of steps. If a answers are expected at each step, and subsequent questions do not depend on previous answers, then the total number of steps is determined by the formula

$$n = a^0 + a^1 + a^3 + \ldots + a^b \tag{25}$$

where b is the number of questions. The Petri-model for Raven's progressive matrix technique consists of 2^{61} steps. The tree-like shape of the Petri-model and a large number of steps are determined by the fact that the model takes into account the combination of answers by forming a unique trajectory of the token in the network. A way to combat this deficiency is to include a "history of answers" memory into the model. The presence of such memory means taking into account the principle of the history of answers formulated in Section 1, which means that the dialogue method, when forming the next question, should take into account both the answer just received and the previously perceived answers. To simplify the presentation, we will assume that the model includes the memory Mem, which stores only one answer obtained at the previous step of the dialogue. Thus, the contents of Mem: (1) are automatically updated when dialogue go to the next step; (2) DiMC takes into account the contents of Mem when determining the index of the next question.

Fig. 5 shows the graph of the Petri-model with memory Mem storing the answer obtained in the previous step. The inclusion in the model the memory of the previous answer does not require the formation of a unique trajectory for each conclusion. The target positions remain unique, and the trajectories leading to the target positions have a common part.

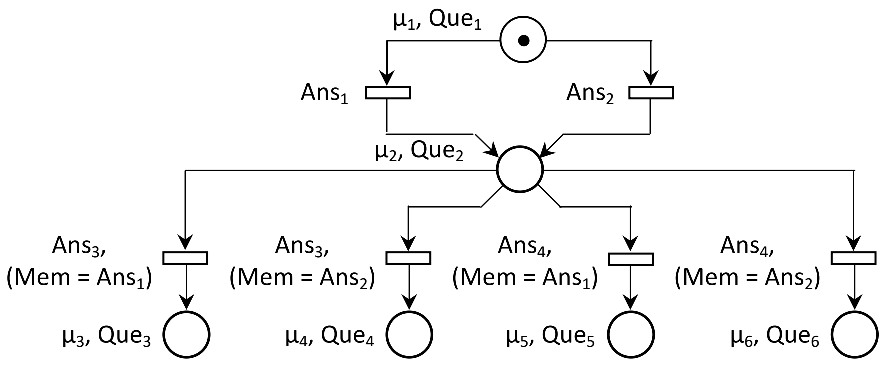

Figure 5: Petri-model in the presence of memory of the previous answer Mem

5. Conclusion

The article describes two formal models of a question-answer dialogue: (1) a finite-automaton model and (2) a model in the form of a Petri net. The models can be used to design a "dialogue machine" that simulates the purposeful behavior of an artificial active dialogue agent, which does not depend on the subject area of the dialogue. The functioning of the dialogue machine is based on the implementation of a unified cognitive cycle of dialogue. The question-answering dialogue model in the form of a Petri net has a wider sphere of applicability than the finite-automaton model. The advantage of this model is that it simulates a typical question-answering process loop using the unified DiMC process, and also allows custom processing of questions and answers using the unique QueDemon and AnsDemon processes. The model in the form of a Petri net also allows taking into account the history of the answers of the reactive agent when forming the next question.

It is supposed to direct further research to develop the main ideas of the described models, but to start not from mathematical structures in the form of a Mealy machine or a Petri net, but from the idea of representing the dialogue method in the form of a network structure *consisting of nodes of different types*. A priori, it is assumed that the network structure is mapped to a relational database. In such a datalogical and network model, the following should be essential: (1) two of the three principles for organizing a question-answering dialogue formulated in Section 1 should be taken into account: the principle of the "depth of dialogue" and the principle of the "history of answers" of a reactive agent; (2) a datalogical approach should be developed and, therefore, a dialogue method for solving a problem, as well as questions of an active agent, should be represented by data stored in a database.

6. References

[1] A. M. Turing, Computing machinery and intelligence, Mind, 49, (1950) 433-460.

[2] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, 2nd ed., Prentice Hall, New Jersey, 2003.

[3] U. Neisser, Cognition and Reality, W.H. Freeman and Company, San Francisco, 1976.

[4] N. Belnap, T. Steel, The Logic of Questions and Answers, Yale University Press, New Haven and London, 1976.

[5] J. Hintikka, Socratic Epistemology: Explorations of Knowledge Seeking by Questions, Cambridge University Press, Cambridge, 2007.

[6] A. Newell, Remarks on the relationship between artificial intelligence and cognitive psychology, in: R. Banerji and J.D. Mesarovich (Eds.), Theoretical Approaches to Non-Numerical Problem Solving, Springer-Verlag, New York, NY, 1970, pp. 363-399.

[7] E. Wenger, Artificial Intelligence and Tutoring Systems. Computational and Cognitive Approaches to the Communication of Knowledge, Morgan Kaufman, Los Altos, California, 1987.

[8] Zhou, L., Gao, J., Li, D., and Shum, H.-Y., The design and implementation of XiaoIce, an empathetic social chatbot, Computational Linguistics 46(1) (2020), 53-93.

[9] Lawson, Mark V, Finite automata, CRC Press, Taylor & Francis Group, 2004.

[10] Graham A. A., Framework for Enterprise Data Architecture, 2nd edition, Koios Associates Ltd., 2012.

[11] J. Peterson, Petri Net Theory and the Modeling of Systems, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[12] I. Chmyr, Dialogue of Partners as a Method of Non-Formal Problem Solving, in Maddy D. Brouwer-Janse and Thomas Harrington (Eds.), Human-Machine Communication for Educational Systems Design, NATO ASI Series F129, Springer-Verlag, Berlin, 1994, pp. 222 – 228.

[13] J. Raven, Mental tests used in genetic studies: The performance of related individuals on tests mainly educative and mainly reproductive, Master's thesis, University of London, London, 1936.