Recommendation of Collaboration Patterns for Human-Machine Collective Intelligence

Alexander Smirnov, Andrew Ponomarev
SPC RAS
St. Petersburg, Russian Federation
{smir, ponomarev}@iias.spb.su

Abstract—**The problem of supporting collaborative processes is especially acute in scenarios, where teams are formed dynamically and team members have little experience in working with each other. The paper proposes an approach to increase the efficiency of the collective activities in the context of a human-machine collective intelligence environment for decision support. The approach consists of two stages. In the first stage, non-productive situations of the collective work are identified by a set of rules, applied to the ontological representation of the status of the team activities, and a set of candidate recommendations of team behaviors (collaboration patterns) is formed. In the second stage, the candidate list of recommendations is ranked and filtered according to teams' feedback by a contextual bandits algorithm. Gradually, it allows to validate the set of rules and increase the relevance of the recommendations.**

I. Introduction

Many problems that arise in the management of complex systems (e.g., large organizations, environment) require a collective effort. This often happens either due to the fact that all the information needed to resolve the problem is spread across many participants or due to the limited processing capacity of a single participant. The necessity of collective effort supports the importance of various technologies supporting collective intelligence, understood as an ability of a group to solve various tasks.

Though the history of collective intelligence as a distinct research field can be traced back to the 1980s (and the problems of effective group interactions were discussed in various disciplines even before that), currently, this field faces new challenges. These challenges are connected with the emergence of mixed human-machine collectives, where software agents play not only a passive role of computational tools, but an active role, being able to initiate interaction and respond to it. This poses new questions about how to make such interactions effective, what are the requirements for software agents, what technologies are suitable for the organization of human-machine teams, and many others.

This paper presents results obtained during the continuing project aimed at the creation of human-machine collective intelligence (HMCI) environment for decision support. The overall design of the HMCI environment is described in earlier publications of the authors [1], [2].

The goal of the research is to create *an environment* supporting (and promoting) collective intelligence in the form of self-organization of teams of heterogeneous members in decision-making scenarios.

The distinguishing features of the HMCI environment are team self-organization, *ad hoc* nature of the teams, and ontology-based interoperability. It is shown, that for many complex problems especially in highly dynamic domains it is not viable to arrange a well-defined workflow in advance, because the situation changes very fast [3]. A natural technological answer to that is the creation of a new generation of human-machine systems, characterized by adaptiveness. Organizational decision-making is a dynamic and complex process because decision-making requirements are constantly changing over time and vary from person to person [4]. Decision support, therefore, is one of those kinds of activities that require adaptiveness of the system and flexible workflow, because decision-making very often is based on an interactive and iterative exploration of the problem.

The *ad hoc* nature of the considered teams makes the HMCI environment somewhat similar to crowdsourcing systems, where a problem is given to an undefined community. However, in most crowdsourcing scenarios, first, there are no interactions between participants, they don't have to establish connections, second, there is no seamless integration of software tools (other than problem-specific human input processors).

To sustain various coordination processes, as well as information flow during decision-making the multilevel interoperability has to be provided inside the collaborative environment. This is especially acute in the case of mixed teams, consisting of human and software agents. It is proposed to use ontologies as the main means of ensuring interoperability. The key role of the ontology model is in its ability to support semantic interoperability as the information represented by ontology can be interpreted both by humans and machines. Ontology-based information representation provides interoperability for all kinds of possible interactions (human-human, human-machine, machine-machine).

The HMCI environment supports the usage of artificial intelligence for supporting collective intelligence in two orthogonal ways. First, software agents can join hybrid teams and participate in decision-making processes by providing some data and/or specific processing capabilities. Second, foundational mechanisms of the environment support some aspects of collaborative work, making it more efficient. This paper deals with the latter way, as it presents the research aimed at helping the team to expedite the decision-making process by analyzing the current state of this process and recommending appropriate collaboration patterns and structures.

The rest of the paper has the following structure. Section II describes the related work in supporting team collaboration. Section III introduces the details of the discussions' representation in the HMCI environment, which largely defines the approach to collaboration patterns recommendation, described in Section IV.

II. Related Work

The problem of supporting the collaboration processes and increasing their efficiency is very important. It has received much attention both in the scope of decision and management science in general and in the scope of group decision support systems.

A traditional approach to this problem relies on a participant of a collaborative process, playing a special role of facilitator [5]. The ultimate goal of the facilitator is to recognize the status of the discussion process, detect the non-productive behavior of the participants and apply the protocols/guidelines to fetter this non-productive behavior, focusing the attention of the group on the most important parts of the problem. Typically, the facilitator is not an expert in the domain where lies the problem being discussed by the group, instead, he/she is an expert in one (or more) methodologies of the group work (e.g., design thinking [6], [7]).

While the presence of an experienced human facilitator usually has a very positive effect on the group work, obviously, this approach has very limited scalability. There is a shortage of facilitation expertise, which is an especially limiting factor in the case of large-scale collective intelligence systems, based on free participation (e.g., crowd-based) [8].

These limitations of personal human-driven facilitation have led to two methodological and technological branches of research: collaboration engineering and automated facilitation. Here, we briefly discuss the most important and relevant achievements in both branches.

The emergence of collaboration engineering [9], [10] as a research field was motivated by a necessity to formalize collaboration procedures and develop a limited set of well-defined practices that could be executed by practitioners without guidance from dedicated facilitators. It is assumed that collaboration practices (of some organization) are first developed by a "collaboration engineer" (equipped with both collaboration engineering theory and the details of the particular organization) and then transferred to the practitioners (e.g., in the form of workflows) to be executed by them without supervision.

In particular, the research in collaboration engineering has resulted in the identification of 6 collaboration patterns [9]: generate, reduce, clarify, organize, evaluate, build consensus. Each of these patterns is characterized by a certain direction of the group's work (e.g., generate pattern is characterized by moving from fewer to more concepts shared by the group). Some general patterns can further be specialized. At the same time, collaboration engineering offers concrete protocols for implementing these patterns, also called ThinkLets. The idea is, these ThinkLets can be used by collaboration engineers as building blocks to construct complex collaboration procedures.

While collaboration engineering is mostly focused on designing collaboration procedures so that they could be implemented without facilitators, automatic facilitation is focused on developing methods and algorithms of (at least partial) automating facilitators' functions, making them more scalable.

Automatic facilitation requires the interaction of a software entity (facilitator component) with the group, understanding the current situation, and identifying steps to make collective action more efficient.

As for the interaction, it is typical, that messages (recommendations), generated by the automated facilitator mimic messages from group members. The most popular form of group discussions, considered in publications on automatic facilitation is text, therefore, the automated facilitator acts just as one of the members of the forum (chatbot, conversational agent) [11], [12].

There are two main ways of understanding the current situation for an automated facilitator: a structured representation of a discussion or a structured representation augmented with natural language processing. For example, the authors of [8] rely on the fact that information in discussion systems (issue-based information systems, IBIS) is often organized with the help of interconnected "problem", "idea", "argument", etc. Therefore, the status of the discussion can be expressed as some quantitative characteristics of this discussion graph. The paper [12] extends this approach by performing natural language processing to classify each of the messages into the same categories ("problem", "idea", "argument", etc.). However, some approaches sidestep the problem of understanding the current discussion situation and provide automatic facilitation basically as an interactive knowledge base on the particular group work method [7].

To identify the recommendations, there are two main approaches: machine learning and rule-based. For example, in [8] case-based reasoning is applied, connecting the structured representation of a discussion status with the facilitation decisions made in a similar situation by a human facilitator. The automatic facilitator proposed in [7] is based on a knowledge base, manually collected by the authors of the respective paper.

The papers on automatic facilitation discussed above show that these approaches allow improving collaborative processes, helping teams to focus on more important parts of a problem and avoid non-productive effects (like *groupthink* [13]).

However, the setup considered in this paper differs from these papers – the HMCI environment relies on a specific ontology-based representation of the problem information (rooted in decision-support methodologies), while the existing papers research either rely on a generic IBIS discussion structure [8], [12] or describe just an interactive knowledge repository without attempting to analyze the current situation of the collaborative process [7] and adapt recommendations to this situation.

III. Representation of the Collaboration Status

Similar to the approach, described in [8], the approach proposed in this paper relies on the structured representation of the problem status. This section describes how the status of the collective work is represented in the HMCI environment, resulting in a formal definition of the collaborative process recommendation problem.

The HMCI environment for the decision support [1] provides a set of mechanisms, allowing team members (human participants and software agents) to collaborate to analyze the problem offered by the end-user (e.g., corporate decision-maker) to collect all the factual information and (expert) estimations to help in making a decision.

Each problem is treated in the HMCI environment as a kind of independent "project" and there is a team collected to address a particular problem. The team may change dynamically, according to the requirements – if during the problem analysis it turns out that current team members do not have all the necessary competencies, the team may be extended by hiring new members.

All the interactions of the team are done via an ontology-based smart-space [14], [15], holding a representation of the problem, that becomes richer and richer as the team progresses with the problem (initially it contains only problem definition received from the end-user, then all the details that were specified during the problem clarification, identified alternatives, their evaluations, etc.). This ontology-based smart space solves two design problems. First, it provides communication between heterogeneous members of a team. The information in this smart space can be read by software agents via querying and/or publish/subscribe methods, besides, the information encoded in this space can also be presented to human participants in a structured way (not necessarily involving underlying ontological expressions). Second (and the most important in the context of this paper), this structured ontological representation allows analyzing the current progress on the problem, enabling various recommendations, regarding process and collaboration techniques.

The status is represented using several ontologies: decision-making ontology, argumentation ontology, data provenance ontology, and competencies ontology.

The decision-making ontology defines main concepts and relationships that are typically used during decision-making. This ontology was designed based on the analysis of existing decision-making methodologies to be compatible with the majority of them. The ontology is described in detail in [16], here we provide only the main concepts of it crucial for the collaboration recommendation problem, discussed in this paper (Fig. 1).

In particular, this ontology defines the role of each information piece in the overall process – is it related to the problem statement, alternative specification, or evaluation of an alternative with respect to a certain criterion.

In some cases, the specific values of some alternative evaluations may be disputable. Team members may offer their evaluations, support already existing evaluations with new

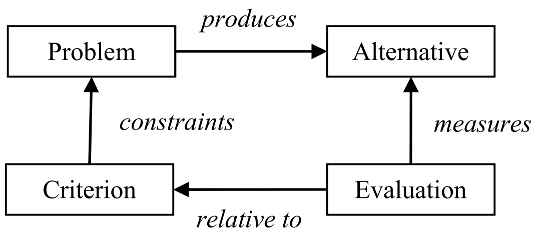

Fig. 1. Decision support ontology (fragment)

evidence, or contradict evaluations. This is modeled with a help of simple arguments ontology consisting of only two kinds of statements: supporting statement and contradicting statement.

The source of a statement can have a significant effect on its importance, and also on the effort required for verification, for example. To track the sources of the information pieces the HMCI environment adopts PROV-O model [17], which allows representing provenance information using ontological statements. The core of the PROV-O ontology is formed by three concepts: Entity (usually, some artifact), Activity (some process that may generate and/or use Entities), and Agent (an active party that can be associated with Activities, and to which Entities may be attributed). In particular, argumentation statements (supporting and contradicting) are typical Entities, which are attributed to team members who produced them. Another example of Entities in the context of the HMCI environment are various artifacts created by team members. Moreover, relationship `prov:wasInformedBy` (between the activities) and `prov:used` (between an activity and an entity) may represent some artifacts used as a basis for some statement.

The competency ontology helps to account for the skills and experience of the authors of the statements. So, the chain is the following. Each supporting or contradicting statement is assigned authorship (provenance chain) via PROV-O statements, and each team member, mentioned in his provenance chain has a profile, where his/her competencies are described with the help of the competence ontology. It allows getting an idea about the statement's strength of support.

A partial example of a problem representation is shown in Fig. 2. The ontology graph is shown in a "collapsed" form (some attributes are shown in the same rectangle as the entity they describe). ID is the unique identifier of the nodes (which typically are anonymous), prefix "dm" used for decision support ontology concepts and relations, prefix "prov" – for PROV-O ontology, unprefixed roles are built-in, except for "a", which means that an individual belongs to some class.

Such description is generally hard to obtain from human participants (and expressing the judgments via ontological structures is not typically convenient for people). Therefore, the structured representation is formed in an interactive way, with a

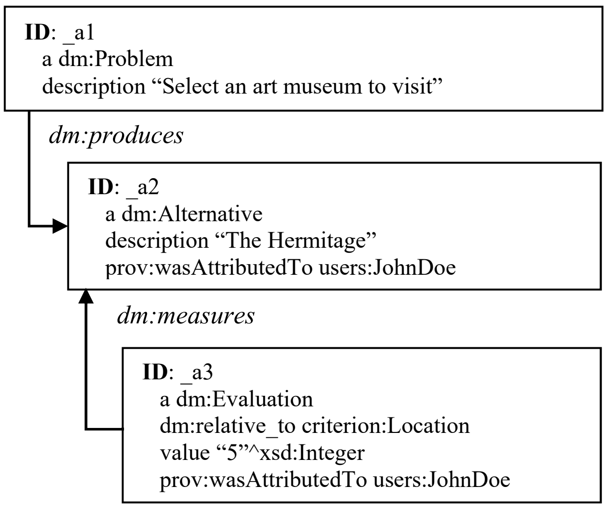
ID: _a1
a dm:Problem
description "Select an art museum to visit"
dm:produces
ID: _a2
a dm:Alternative
description "The Hermitage"
prov:wasAttributedTo users:JohnDoe
dm:measures
ID: _a3
a dm:Evaluation
dm:relative_to criterion:Location
value "5"^xsd:Integer
prov:wasAttributedTo users:JohnDoe

Fig. 2. Partial example of problem representation

help of natural language processing models identifying the role of a particular piece of information (similar to [12]), and GUI solutions helping to disambiguate the classification done by prediction models (confirming that the links identified by the models are correct). The whole procedure of obtaining this representation is out of the scope of the paper.

There are two possible forms of recommendations that could be offered to a team. First, it could just be a textual recommendation, offering a certain plan of actions to the team, and the team may interpret this recommendation in some way. An example of such recommendation is "A severe disagreement is detected on issue <EVALUATION>. Possible options are: a) hire new members on <COMPETENCY> to get additional expertise, b) activate a Delphi protocol to reconcile the evaluations." The second form is to offer this recommendation in a more actionable manner. The environment maintains a general plan of team activities, and the team may be offered to integrate the proposed actions ("hire new members" or "activate a Delphi protocol" in the example above) to this plan. A protocol typically consists of several steps that have to be executed in a certain order with certain criteria. In this case, all these steps are added to the team's plan. In both cases, the possible actions should be identified. This paper focuses exactly on that – identification and ranking of the possible ways of action. The presentation of the recommendations to the team and/or their integration to the team's plan is a technical task, not discussed here.

IV. Recommendation Method

The input of the recommendation method is the ontological representation (graph) of current problem information, contained in the smart space. The method analyzes this graph and selects possible actions to improve the team's work if some non-productive situation is detected.

The proposed method is based on the following consideration. There is a huge variety of situations in collaborative work, and only a few of them require intervention. The approaches based on pure machine learning would require a very large record of team activities. Besides, there are no public datasets available, where collaboration situations are described by the same set of relationships as in the HMCI environment being developed. Therefore, all the data that can be used has to be collected by the HMCI environment, which means a typical system's "cold start" situation. Therefore, it is proposed to use a hybrid approach, based on a knowledge base and collected feedback.

The knowledge base is used to detect situations when recommendations can be offered. The source of the knowledge base is the massive literature on facilitation principles, collaboration engineering, as well as collaboration protocols and procedures. Technically, the knowledge base consists of rules. Each rule has a (potentially non-productive) situation description as an antecedent and possible resolution (recommendation) as a consequent.

Rule antecedents are evaluated on the team's smart space, augmented with additional features, quantitatively describing the situation on a high level. These features allow to generalize situation description and lower the complexity of graph pattern matching during the rule evaluation. The list of the features grouped by the respective entity is shown in Table. I.

TABLE I. Aggregate Features

Feature Name	Brief Description
Problem	
P_Time	Time (in hours) the problem is active
P_NAlternatives	The number of alternatives
Alternative	
A_Time	Time (in hours) since the alternative has been introduced
A_NSupport	The number of supporting statements
A_NContr	The number of contradicting statements
A_MeanSupport	Mean competency of the supporting statement authors (including all the entities, reachable via PROV-O links)
A_MeanContr	Mean competency of the contradicting statement authors (including all the entities, reachable via PROV-O links)
Evaluation	
E_Time	Time (in hours) since the evaluation has been added
E_NSupportS	The number of supporting statements
E_NContrS	The number of contradicting statements
E_MeanSupport	Mean competency of the supporting statement authors (including all the entities, reachable via PROV-O links)
E_MeanContr	Mean competency of the contradicting statement authors (including all the entities, reachable via PROV-O links)
E_NSupportE	The number of entities, referenced by supporting statements (via PROV-O links)
E_NContrE	The number of entities, referenced by contradicting statements (via PROV-O links)

Each rule can also be characterized by an "anchor entity", which is an entity, being the root of the collaboration problem, described by the rule, and the resolution of the collaboration problem can usually be done considering only this entity and more "fine-grained" entities connected to it. For example, if a rule deals with a situation, when there are too few alternatives considered, then the "anchor entity" is a Problem, because the situation affects the whole Problem and has to be resolved at the level of the Problem. At the same time, if a rule deals with a

RULE ID: Alternative_Evaluation_Conflict_1

ANCHOR ENTITY: Alternative

ANTECEDENT:
```
SELECT ?alt, ?crit, ?ev1, ?ev2
WHERE {
    ?alt a dm:Alternative.
    ?crit a dm:Criterion.
    ?ev1 [dm:measures ?alt;
          dm:relative_to ?crit].
    ?ev2 [dm:measures ?alt;
          dm:relative_to ?crit].
    FILTER (?ev1 != ?ev2).
}
```

CONSEQUENT:
```
BUILD_CONSENSUS(?alt, ?crit)
```

DESCRIPTION: An alternative has conflicting evaluations on one criterion. The team has to disambiguate the evaluation (*build consensus* pattern).

RULE ID: Too_Few_Alternatives

ANCHOR ENTITY: Problem

ANTECEDENT:
```
SELECT ?problem
WHERE {
    ?problem a dm:Problem.
    ?problem fts:P_Time ?t.
    ?problem fts:P_NAlternatives ?na.
    FILTER (?t > 24 and ?na < 2).
}
```

CONSEQUENT:
```
GENERATE_ALTERNATIVES(?problem)
```

DESCRIPTION: Too few alternatives for the problem that exists already more than 24 hours. This may result in premature focus on the suboptimal solution. *Generate* pattern should be applied (e.g. brainstorming).

Fig. 3. Examples of the rules

situation, when there are too many conflicting evaluations of the same alternative, the "anchor entity" is the Alternative, because the resolution of this situation requires mostly information on the Alternative level and below (more specific).

Technically, the antecedent of a rule is defined as a SPARQL query, executed on the augmented smart space. Examples of the rules are shown in Fig. 3. The first rule detects the lack of agreement in the team, and, based on collaboration engineering guidelines, recommends to apply a *build consensus* pattern (one of its forms) [9]. The second rule shown detects the lack of diversity in teams' proposals, and recommends to correct that with *generate* pattern (e.g., brainstorming).

The collaboration recommendations identified with the rule-based approach are based on the general theory of collaboration. However, on the one hand, the process of translation from theoretical recommendations to specific rules may introduce some bias, on the other hand, in many cases, there are several relevant options of addressing some collaboration problem, and certain team members may prefer some of them (for example, due to having more experience with them). Therefore, the list of general recommendations generated according to rules has to be evaluated with a feedback-aware model, to offer the most appropriate recommendations.

We have chosen contextual bandits theoretical framework, popular in the field of recommender systems [18], [19], to implement this feedback-aware ranking of the possible recommendations. Contextual bandits algorithms are online learning algorithms that are aimed at making sequential choices between actions (with initially unknown payoffs) in various contexts, so that the average payoff is maximized. On each round, a contextual bandit algorithm a) observes a set of actions (typically called "arms") together with their features (context), b) chooses an arm, based on the previous payoffs and receives some payoff, c) improves the selection strategy taking into account this observation. The advantage of this framework is that it allows to balance exploration (searching for good options) and exploitation (offering good options to the users) and can take into account additional context factors (like user profile, situation description, etc.).

In collaboration pattern recommendation, a possible collaboration protocol recommendation (identified by the rules) plays the role of an action (or, "arm"), besides, there is also a null-action, which is not to offer any recommendation. Features of the anchor entity of the rule used to infer the recommendation play the role of the context. Finally, the team's feedback for the recommendation is interpreted as the payoff. The feedback is collected explicitly – if the team decides to follow a particular recommendation, it answers positively and this positive answer is treated as a positive reward for the recommended action. Each recommendation not selected by the team receives zero payoffs. The action of not showing any recommendation always has some small payoff.

In particular, we use LinUCB [18] algorithm for action value estimation and picking the action on each time step (when there are recommendations, offered by the rule engine). This algorithm evaluates payoff as a linear combination of feature values and learnable parameters, estimates confidence bounds, depending on the quantity of feedback, and adjusts the learnable parameters with the new experience. More specifically, we have an independent LinUCB model for each anchor entity class (for Problem, Alternative, Evaluation, Statement).

The overall schema of the proposed method is shown in Fig. 4. The contents of the discussion smart space are processed by the feature extractor component, which creates the aggregate features (Table 1), describing each of the important pieces of problem information. The knowledge-based recommendation engine uses these feature representations (and their relationships) to identify if the situation requires the recommendations. The output of the rule engine is a set of all recommendations that are adequate for this particular situation, according to the contents of the knowledge base (basically, it is a list of consequents for all the rules, which antecedents hold). After that, the list of possible recommendations is processed by the ranking module that orders the recommendations according

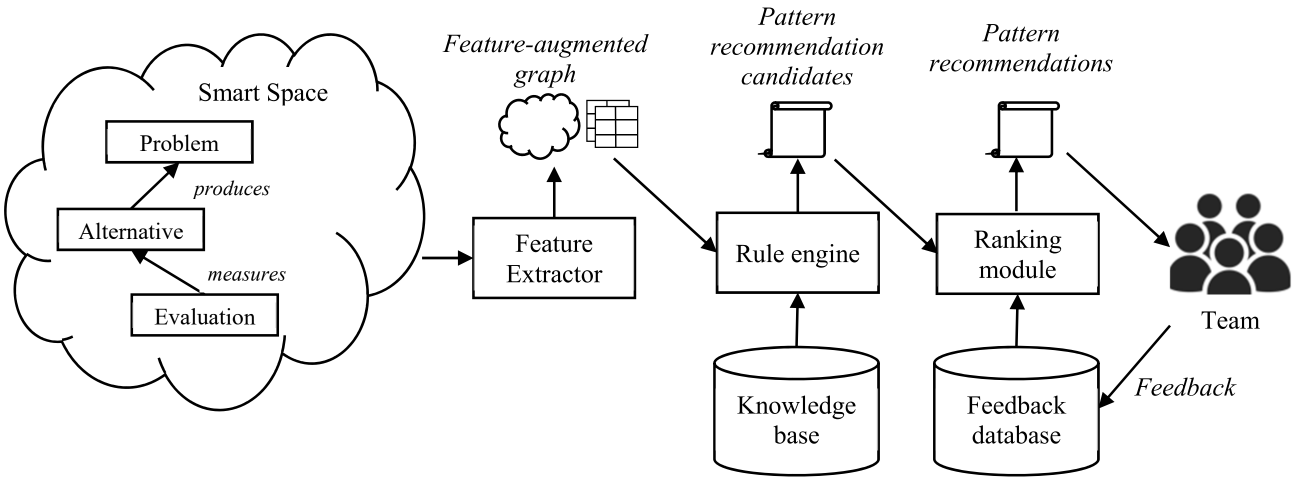

Fig. 4. The schema of the approach

to the context given by the situation features. The top 3 recommendations are shown to the team. The team's feedback on the recommendations is collected to be used in adjusting the context-aware ranking strategy.

V. Conclusion

The paper proposes an approach to increase the efficiency of the collective activities in the context of the human-machine collective intelligence environment for decision support. The approach consists of two stages:

1) Identification of the non-productive situations of the team work with a set of rules, and forming a set of candidate recommendations of team behaviors (collaboration patterns). The rules are evaluated on the ontology-based smart space containing the structured representation of the collective work.

2) Ranking and filtering of the candidate recommendations set according to teams' feedback by a contextual bandits algorithm (particularly, LinUCB).

The proposed approach is flexible and extensible. In particular, it can be extended by adding new rules and introducing new features, describing the status of the collective work.

Our future work is related to:

- Using existing argumentation ontologies, which will contribute to semantic interoperability between team members and richer representation of relationships between statements.
- Considering time progression. The analysis module might classify not only the current status of the problem, but also take into consideration previous states. For example, it might allow detecting that a team is following some procedure, therefore, the recommendations should be aligned with this procedure.
- Accounting for implicit feedback. Currently, feedback on recommendations is collected explicitly, but, on the one hand, it adds some burden to the team, on the other hand, it might be non-reliable (a team may agree to follow a recommendation, but do not follow it). Detecting if a team is actually following a recommendation would help to address the both drawbacks.

Acknowledgment

The project is funded by the Russian Science Foundation (grant # 19-11-00126).

References

[1] A. Smirnov and A. Ponomarev, 'Human-Machine Collective Intelligence Environment for Decision Support: Conceptual and Technological Design', in *2020 27th Conference of Open Innovation Association,* Trento, Italy, 2020, pp. 253-259.

[2] A. Smirnov and A. Ponomarev, 'Decision Support Based on Human-Machine Collective Intelligence: Major Challenges', in *Internet of Things, Smart Spaces, and Next Generation Networks and Systems. Lecture Notes in Computer Science, vol. 11660*, Springer, Cham, 2019, pp. 113–124.

[3] D. Retelny, M. S. Bernstein, and M. A. Valentine, 'No Workflow Can Ever Be Enough: How Crowdsourcing Workflows Constrain Complex Work', *Proceedings of the ACM on Human-Computer Interaction*, vol. 1, no. 2, article 89, Dec. 2017.

[4] C.-S. S. J. Dong and A. Srinivasan, 'Agent-enabled service-oriented decision support systems', *Decision Support Systems*, vol. 55, no. 1, pp. 364–373, Apr. 2013.

[5] A. Adla, P. Zarate, and J.-L. Soubie, 'A Proposal of Toolkit for GDSS Facilitators', *Group Decision and Negotiation*, vol. 20, no. 1, pp. 57–77, Jan. 2011.

[6] R. Razzouk and V. Shute, 'What Is Design Thinking and Why Is It Important?', *Review of Educational Research*, vol. 82, no. 3, pp. 330–348, Sep. 2012.

[7] E. Bittner and O. Shoury, 'Designing Automated Facilitation for Design Thinking: A Chatbot for Supporting Teams in the Empathy Map Method', *Proceedings of the 52nd Hawaii International Conference on System Sciences*, pp. 227–236, 2019.

[8] W. Gu, A. Moustafa, T. Ito, M. Zhang, and C. Yang, 'A case-based reasoning approach for automated facilitation in online discussion systems', *KICSS 2018 - 2018 13th International Conference on Knowledge, Information and Creativity Support Systems, Proceedings*, pp. 1–5, 2018.

[9] R. O. Briggs, G. J. de Vreede, G. L. Kolfschoten, and D. L. Dean,

‘Defining key concepts for collaboration Engineering’, *Association for Information Systems - 12th Americas Conference On Information Systems, AMCIS 2006*, vol. 1, no. January, pp. 117–124, 2006.

[10] G. L. Kolfschoten and G. J. De Vreede, ‘The collaboration engineering approach for designing collaboration processes’, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4715 LNCS, no. January 2014, pp. 95–110, 2007.

[11] E. Bittner, S. Oeste-Reiß, and J. M. Leimeister, ‘Where is the Bot in our Team? Toward a Taxonomy of Design Option Combinations for Conversational Agents in Collaborative Work’, *Proceedings of the 52nd Hawaii International Conference on System Sciences*, vol. 6, pp. 284–293, 2019.

[12] T. Ito, S. Suzuki, N. Yamaguchi, T. Nishida, K. Hiraishi, and K. Yoshino, ‘D-Agree: Crowd Discussion Support System Based on Automated Facilitation Agent’, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 09, pp. 13614–13615, 2020.

[13] M. E. Turner and A. R. Pratkanis, ‘Twenty-Five Years of Groupthink Theory and Research: Lessons from the Evaluation of a Theory’, *Organizational Behavior and Human Decision Processes*, vol. 73, no. 2–3, pp. 105–115, Feb. 1998.

[14] D. G. Korzun, S. I. Balandin, and A. V. Gurtov, ‘Deployment of Smart Spaces in Internet of Things: Overview of the Design Challenges’, 2013, pp. 48–59.

[15] L. Roffia, P. Azzoni, C. Aguzzi, F. Viola, F. Antoniazzi, and T. Salmon Cinotti, ‘Dynamic Linked Data: A SPARQL Event Processing Architecture’, *Future Internet*, vol. 10, no. 4, p. 36, Apr. 2018.

[16] A. Smirnov, T. Levashova, A. Ponomarev, and N. Shilov, ‘Methodology for Multi-Aspect Ontology Development: Use Case of DSS Based on Human-Machine Collective Intelligence’, in *Decision Support Systems XI - Decision Support Systems, Analytics and Technologies in response to Global Crisis Management. Lecture Notes in Business Information Processing*, U. Jayawickrama, P. Delias, M. E. Urmeneta, and J. Papathanasiou, Eds. Springer, 2021.

[17] ‘The PROV Ontology’. [Online]. Available: https://www.w3.org/TR/prov-o/. [Accessed: 14-Apr-2021].

[18] L. Li, W. Chu, J. Langford, and R. E. Schapire, ‘A contextual-bandit approach to personalized news article recommendation’, in *Proceedings of the 19th international conference on World wide web - WWW ’10*, 2010, pp. 661–670.

[19] X. Xu, F. Dong, Y. Li, S. He, and X. Li, ‘Contextual-Bandit Based Personalized Recommendation with Time-Varying User Interests’, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 6518–6525, Apr. 2020.