Collaborative Problem-Solving Technologies: A Taxonomy of Issues

Steven L. Tanimoto
University of Washington
Seattle, Washington 98195
Email: tanimoto@cs.washington.edu

Sandra B. Fan[*]
University of Washington
Seattle, Washington 98195
Email: sbfan@cs.washington.edu

***Abstract*—This paper defines several types of systems and technologies that support humans in solving complex problems, and then presents a set of issues relevant to the design of these systems. The issues include how to structure workflows, crowd-sourcing, autonomous agents, education and training, version control, collaboration and problem formulation processes. The aim of the paper is to facilitate discussion and future research in the design of effective systems to help humans solve difficult problems, particularly those involving global challenges such as climate change, world poverty, nuclear weapons proliferation, and fake news.**

I. Introduction

The purpose of this paper is to identify and describe a collection of issues that designers and engineers need to address in order to create effective new systems to support humans in solving challenging problems. We hope this paper will facilitate constructive discussions and ultimately lead to improved designs for powerful collaborative problem-solving tools.

Before we define various types of systems and technologies for problem solving, let us describe some of the global challenges that such systems should be able to address, and clarify the context in which to situate this discussion.

A. Global Challenges

How can we use use technology to facilitate solving global problems such as pollution of the oceans, nuclear proliferation, climate change, fake news, and drug-resistant bacteria? Solving these problems is difficult not only because they involve scientific modeling and understanding, but because there are opposing stakeholders that want different or no solutions to the problems. For example, certain stakeholders may be able to push their viewpoints by generating fake news content, and others are able to profit financially from it. At the same time, proliferation of such content may have negative effects on society. New technologies, systems, and processes may be able to help in resolving conflicts and gaining understanding of problems and possible solutions. This paper addresses the question of what sorts of possibilities and challenges there are in designing new systems to help people solve complex problems.

B. Supporting a Culture of Problem Solving

Although an important end goal is to solve specific problems such as arresting global warming, a more accessible and broad goal is to help foster increased awareness and skill on the part of humans to engage productively in problem solving activities. Insofar as stakeholder conflicts are a major roadblock to solving some problems, a change in human understanding and attitudes is an important step towards solving such problems. Even when many conflicts cannot be quickly resolved, the identification of shared goals or shared subgoals can provide milestones along the road to possible broader agreements.

C. Convergence of Technologies

As many computing technologies continue to advance, such as artificial intelligence, supercomputers, software development tools, and interaction devices, there is an opportunity to do more to support problem solving than has been possible in the past. However, it is not clear how best to integrate these technologies to help people solve complex problems. It is timely to address the question of how best to bring together new technologies so as to take advantage of them for problem solving.

II. Definitions

Here we provide working definitions of terms that we use later in the paper. We will ground these definitions using fake news as an example problem.

A. Problem

A *problem* is a need, identified or not. It could be the need to obtain money or some physical object or to obtain the change in some aspect of the state of the world or the state of a puzzle or virtual world.

In terms of fake news, the problem is the need for people who consume news to be aware of who generated the content and what their motives may be.

This paper represents a part of the proceedings of the Workshop on Designing Technologies to Support Human Problem Solving: DTSHPS'18. ©2018, Steven L. Tanimoto and Sandra B. Fan.

[*]Dr. Sandra Fan at the time of this publication is with Google Seattle.

B. Problem formulation

A *problem formulation* is a description or representation of a problem, typically in a way that makes the problem "actionable" or that permits recognition of possible solutions. Formulations may be informal (e.g., textual descriptions) or formal (e.g., executable code).

There are many ways to formulate the problem of fake news as it is a wicked problem (defined in II-D). We could informally say that our problem formulation is to find ways to show news consumers the provenance of new articles, or to educate people on how to distinguish fake news, or that we need to socially or legally disincentivize fake news creation or propagation. One formal way to formulate the problem might be, we could say that we would like to build a newsreader that can filter out unreliable news sources. Possible solutions would correspond to different features that may be implemented in the newsreader.

C. Problem space

A *problem space* is a set of possible "states" or configurations of problem elements that includes the starting situation and potentially includes one or more goal states. A problem space may be finite or infinite, depending on the nature of the formulation.

If we continue the idea of constructing a reliable newsreader, our problem space would consist of the possible combinations of potential features in the system. Some examples of these features include: automated (or manual) validation of a user's identity before allowing them to post content, only allowing content from a whitelist of certain sources, asking gatekeeper editors to fact-check content, crowdsourcing fact-checking of articles, natural language processing or machine learning of content for reliability, and so on.

D. Wicked problem

A *wicked problem* is a problem that meets certain criteria [2] that make it particularly difficult to solve. We will define them further in IV-C. Common characteristics of wicked problems are: difficulty in formulating the problem, lack of a clearly right or wrong solution, only better or worse ones, and conflict where opposing stakeholders impede reaching consensus on solutions.

Fake news [5] is a wicked problem in that the issue could be formulated in different ways as described in II-B. There is also no single correct solution. Sometimes it may not be possible to fully verify a news source; does that mean it should be hidden? Some beliefs clearly known to be true are later disputed by further scientific research or additional information. All content creators have their own biases; does this mean it is best to limit news to only concrete facts–dates and times of events–and no analysis of the implications of these events? These questions can perhaps only be answered on a spectrum and in combination with each other.

E. Stakeholders

A *stakeholder* is a party (person or group) to a process that has the power to act on solving a problem, and which has an interest (e.g., financial, healthwise, or ideological) in the outcome of the process. With regards to the problem of fake news, there are content consumers who read the articles, each with their own motivation for doing so–gathering information to make a voting decision, or trying to understand how a policy or news event might affect them. Another stakeholder is the content creator, who may be trying to make a living in some way off of their content, or convince others of a viewpoint. The people who are the subjects of news content–public figures, or sometimes everyday people–are also stakeholders. Their stories are the ones being told to others, and this can affect their lives. All of these stakeholders have different interests in the content.

F. AI Service, AI Solving Agent, ML Agent

An *AI service* is a computational function, typically involving inference, or pattern classification, but which may involve solving a pre-formulated problem. An AI *solving agent* is a computational process that not only can perform one or more AI services, but which may take initiative by watching for opportunities to act and then taking actions at those times.

An *ML agent* is an AI solving agent which performs machine learning. This means that it performs data mining, clustering, classifier training, probability estimation, or rule induction. In our example of a newsreader framework, we might design NLP or ML algorithms for detecting the reliability of a news article.

G. Technical Transparency

An AI service, AI solving agent, or ML agent is *technically transparent* when the function(s) it computes are easily identifiable to users, rather than hidden from them, either intentionally or simply due to the nature of the function. Even if the definition or implementation of such functions are available for end users to examine, it may be difficult to understand their implications.

H. Stakeholder Transparency

A problem formulation, problem solution, AI service, AI solving agent, ML agent or data set possesses *stakeholder transparency* when its biases in favor of or against particular stakeholders are readily apparent, typically because of explicit disclaimers. An ML agent may be biased by its training data.

III. System Types

A. Brokering Systems

Systems that help connect problems with people who can solve them are problem "brokers." For example, Innocentive.com, in operation since 2001, runs a brokering service in which users can post problems they would like help with solving. Most of the problems are engineering or chemistry oriented, but others are business-process oriented. Solvers are users who compete with other solvers to win a prize for a best solution.

B. Automatic Solvers

An automatic solver is an AI service or solving agent that takes a suitably formulated problem and tries to return either a legal goal state or a lowest-cost path to a goal state for the problem. Automatic solvers typically use technologies such as heuristic search, simulated annealing, genetic algorithms, case-based reasoning, and/or constraint satisfaction.

C. Collaborative Solving Managers

A system that administers the efforts of a group of humans and/or AI solving agents in exploring a problem space is called a *collaborative solving manager*. An example is CoSolve [3], a web interface for collaboratively exploring a visually-depicted problem space. Users could formally formulate problems (III-D), and then traverse a graph, generating nodes that represent solutions or steps toward a solution. Affordances for collaboration included the ability to see, comment on, and rate these nodes, or to take turns generating the steps toward a solution.

D. Posing Tools

Problem formulation is often the most difficult part of solving a problem. Tools and methodologies supporting formulation we call *posing tools*. Methodologies include ones used in mathematics and science (e.g., see Jonassen). CoSolve's problem formulation interface was in the form of generated Python code snippets that the problem posers could edit to try out different ways of representing their problem.

E. Online Community Support

Mathematician Tim Gowers started the Polymath project to engage the mathematics research community in solving several open problems [6]. An important communication tool in this effort was Gowers' blog, in which he summarized the progress on solving, so that the community efforts could be coordinated.

In addition to blogs, communication methods such as email, discussion forums, and chat rooms can be used as means for sharing status and intentions on the part of solvers in a joint effort. When these communication tools can be effectively integrated with solving activities, some of the difficulties related to identifying session objects, locations in documents, versions and contexts can be avoided.

Koios.org [8] hosts a forum organized according to various social problems that have been identified by users. The site encourages problem solving by identifying top solvers and posting their names.

F. Crowdsourcing Management

Certain problems such as galaxy surveys are well-suited to the citizen science collaboration structure used by Galaxy Zoo. Here, the overall survey problem divides up nicely into thousands of microproblems each of which can be solved independently. The system keeps track of the problems and employs redundancy in the crowdsourcing to control the reliability of results.

G. Data Management

As more and more problems involve digital data, the management of such data becomes important in solving the problems. While it is out of the scope of this paper to characterize the wide variety of such systems, the role that these systems can play in problem solving must be accounted for here.

H. Education and Training

Due to the complexity of global-challenge problems, and due to the need to have humans involved in solving these problems, a major consideration in the design of technologies for problem solving is to help the human users of the system to understand all of the following: the key aspects of the problem being solved, the processes being followed (both by the people involved and the computational parts of the system), and the viewpoints and interests of all stakeholders to the problem. Learning about the problem comes both from external materials that are problem-specific, and exercising the system using existing formulations of the problem to gain an understanding of some of the problem's possible problem spaces. Learning about the processes embedded in the system can come both from tutorials about the system and practice with the system. Finally, understanding the stakeholders involved can come from reading stakeholder descriptions hosted by the system, or reading external materials. The system might host interactive conversations, or prepared videos and other media that help explain other stakeholders' points. Provisions for stakeholder transparency can also contribute to this key aspect of human understanding within a problem-solving environment.

I. Version Control

Both problem formulations and solution activities are subject to frequent revision and negotiation, especially when the problems being addressed are complex. The ability to manage multiple variations of these objects is essential. Version control tools are commonly used in software engineering environments. Similar facilities are needed in complex problem solving.

IV. Taxonomy of Issues

This section revisits many of the topics already discussed, but it lists them in a somewhat more systematic order, with a shallow hierarchy. This may make it easier to point to particular design issues and see them in a context of other issues relevant to the design of collaborative problem-solving systems. Fig. 1 shows the hierarchy as a tree.

A. Problem-Space Theory

1) Identifying and developing theory components most relevant to DTSHPS: Some artificial intelligence (AI) theory is more relevant to automated problem solving than human problem solving, but where humans interact with AI, the theory may be important in facilitating the interaction.

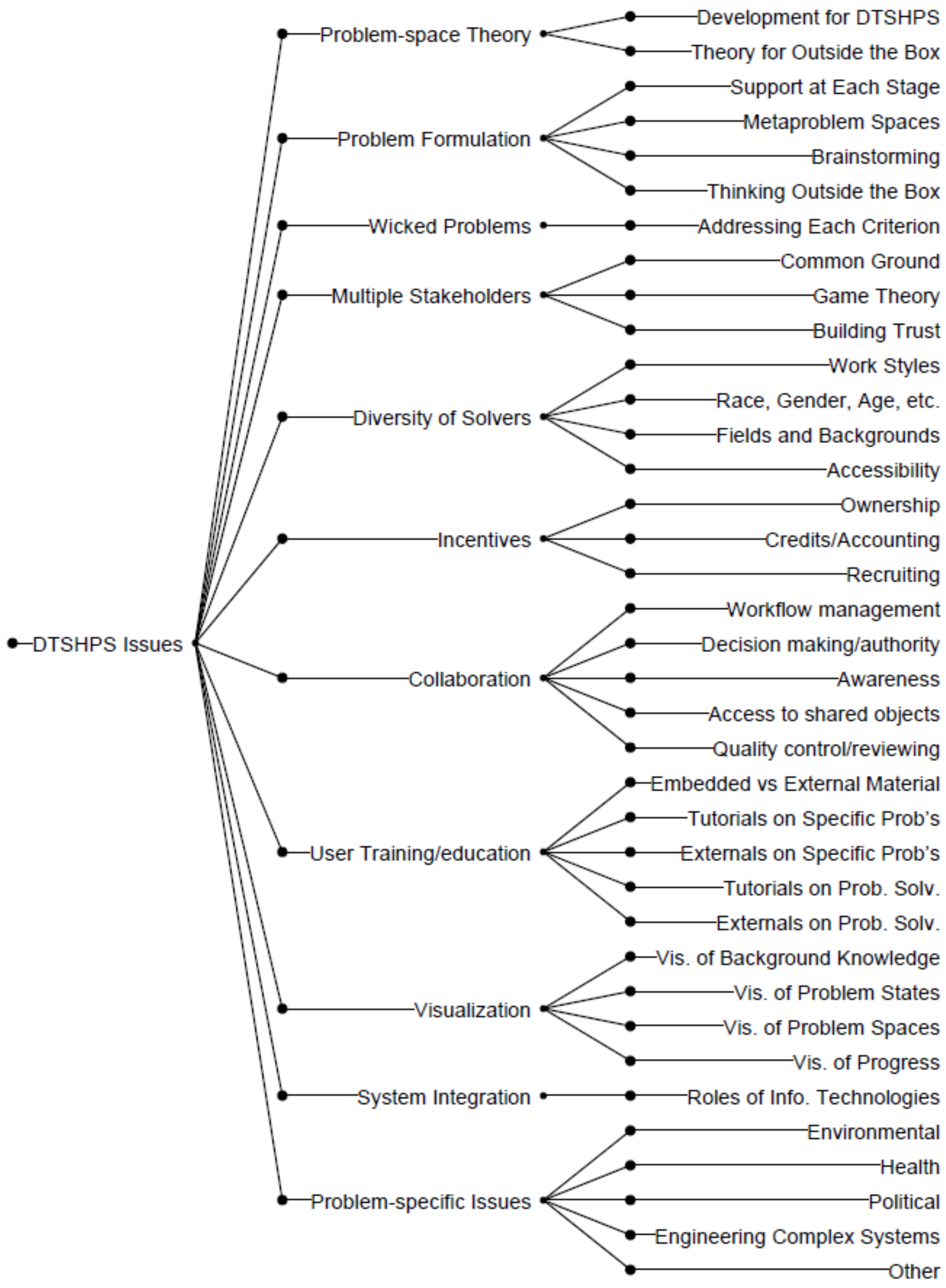

Fig. 1. Taxonomy of issues for designing technologies to support human problem solving. Each node of this tree is discussed explicitly in Section IV.

2) Ways to support "thinking outside the box" (TOTB): The classical theory of state-space search (see Simon and Newell [17]) may seem to be overly prescriptive. How can the theory accommodate human solvers who need to question assumptions and think outside the box?

Approaches include (1) the use of systems of "nested boxes" instead of one "box" (i.e., strict formulation), and (2) using constraints that can be dynamically added or removed in a problem formulation [9].

B. Problem Formulation Issues

1) Best ways to support the various stages of formulation: Here are key stages of the problem formulation process. For each stage there is an issue of how best to support users in that stage.

- identification of the need that corresponds to the problem,
- preformulation (resource discovery, retrieval and analysis),
- posing (decisions about what aspects are most important and relevant, as well as modeling the structure of a problem),
- coding (reduction to computer program snippets),
- testing and
- evaluation of a formulation.

Within the preformulation stage, the techniques of sense-making can lead to usable representations of the information relevant to the problem[15][13][11].

2) Metaproblem spaces: A metaproblem space for a given problem is a space of possible formulations of the problem [7]. The main issue here is how to exploit the metaproblem space notion, which can mean considering problem formulation as negotiating a trajectory through the metaproblem space. Metaproblem spaces can be formalized by setting bounds on possible formulations, such as limiting state representations to use particular kinds of program variables, such as integer variables, etc.

3) Brainstorming: Brainstorming refers to early-stage problem solving activity characterized by the free expression of as many ideas as possible, without critique or judgment. The intent is to prevent people from limiting their problem formulation or solution space by prematurely judging their own or others' ideas. How can we incorporate this aspect of brainstorming into problem formulation? Are there ways to create an environment that prevents critique in early-stage problem formulation?

4) How to accommodate "thinking outside the box": This notion seems most relevant when a problem has an existing formulation or an existing mindset associated with it which is so limiting that good solutions do not seem to be available. One way to encourage TOTB is to maintain a mindset that permits multiple formulations of any problem. If one formulation starts to seem too restrictive, then other formulations can be tried.

Another approach could be called "open formulations of problems" in which some parts of a formulation are considered variable and subject to alternative bindings.

C. Wicked problems

Articles on "wicked problems" typically cite Rittel and Weber's paper [14] when explaining the concept of wickedness. Ten properties of wicked problems were listed in their paper. These are:

1) There is no definitive formulation of a wicked problem
2) Wicked problems have no stopping rule
3) Solutions to wicked problems are not true-or-false, but good-or-bad
4) There is no immediate and no ultimate test of a solution to a wicked problem
5) Every solution to a wicked problem is a "one-shot operation"; because there is no opportunity to learn by trial-and-error, every attempt counts significantly
6) Wicked problems do not have an enumerable (or an exhaustively describable) set of potential solutions, nor is there a well-described set of permissible operations that may be incorporated into the plan
7) Every wicked problem is essentially unique
8) Every wicked problem can be considered to be a symptom of another problem
9) The existence of a discrepancy representing a wicked problem can be explained in numerous ways. The choice of explanation determines the nature of the problem's resolution
10) The planner has no right to be wrong

The central issue here is how to overcome each of these sorts of challenges. Possible approaches for each of the 10 criteria are the following.

(1) an acknowledgment that multiple formulations, with a good analysis of the strengths and weaknesses of each formulation, are more appropriate for wicked problems than a single, oversimplified, formulation.

(2) stopping rules may correspond to goal states in problem spaces, or they may correspond to criteria that determine when a solution is good enough. The concept of satisficing [16] addresses the issue of "optimal or good enough?"

(3) although some traditional problem-solving structures required "true-or-false"-like solutions (puzzles, logic questions, etc.), optimization theory and other methodologies readily accommodate not only fine gradations of solution quality, but multiple criteria.

(4) While no immediate or ultimate test for solutions may exist for wicked problems, rational evaluation criteria can be designed that serve as practical proxies for tests of success.

(5) The one-shot nature of solution opportunities for wicked problems is generally based on the assumption that no good-quality simulation software is available for the problem's domain. Today, simulation software is increasingly effective, and it can permit planners to test many possible solutions before making the large commitment of resources necessary to implement a solution to a wicked problem.

(6) The lack of enumerable potential solutions or permissible operations for a problem may be a consequence of a failure to formulate the problem appropriately. Also, at the time the

Dilemmas paper was written, computer programming practice was not as sophisticated as it is today.

(7) The uniqueness of each wicked problem, while a valid point, need not be an obstacle to principled solution techniques. There is an infinity of unique two-number addition problems, and yet each one of them is solvable by the same simple method of long addition. Qualitative uniqueness is more difficult to deal with, but then patterns do emerge among complex problems; for example, many global challenge problems have to do with resource distribution–food to people, clean water to people, health care to people, etc., and the uniqueness of each problem does not mean uniqueness in every respect.

(8) The fact that a wicked problem may be a symptom of another problem, is a way of saying that the underlying causes of a problem may be initially hidden or considered out of scope. Clearly, a good formulation must take into account all the available variables that might affect the ability to solve the problem.

(9) The alternative representations of a wicked problem relate to the need for multiple formulations, as mentioned for criterion #1.

(10) When Rittel and Webber wrote that ”The planner has no right to be wrong” the government policy planning context probably put unreasonable expectations on what professional planners could deliver, given the tools they had. Today, an enlightened city council can engage community representatives and professionals that can show the results of simulations that account for uncertainty using probabilistic models. The planners then have a right to be wrong in the sense that an implemented solution may not turn out perfectly and under budget, but may fall within the range of anticipated possible outcomes.

Thus an important set of issues regards finding, implementing, and evaluating means of “taming” wickedness by addressing each of the wicked-problem properties. Such means can include various forms of simplification and stating of assumptions, and the incorporation of simulations in the problem-solving environment.

D. Multiple Stakeholders

1) Approaches to common ground: In one model, each party has a hierarchy of objectives. If these are sufficiently compatible, then common subgoals can be used as areas of cooperation, and hopefully the existence of some of these makes it possible to bring various stakeholders to the table.

2) Game-theoretic approaches: By recognizing instances of Nash equilibria and prisoners dilemmas, steps may be taken to ameliorate their negative effects on reaching globally optimal states.

3) Building trust: How to engender trust is a key issue when the stakeholders to a problem are in conflict. An important question is how to create protocols that scaffold the building of trust, or that reduce the need for trust while permitting cooperation to move forward.

E. Diversity of Solvers

A general sociological issue is encouraging diversity in solver communities. Breadth of perspectives can be valuable for solving when the knowledge, techniques, or motivations to solve a problem might be insufficient if the solving team is too homogeneous. Another reason to support diversity in problem solving teams is to enable a broad sense of ownership in a resulting solution that might actually affect the lives of these same or similar people.

Dimensions of diversity relevant to problem solving therefore include (a) knowledge of the problem-domain, (b) problem-solving skills, (c) motivation (ideally at least one team member will be motivated to address each element of the problem), (d) sociological group, including age, gender, race, ethnicity or religion. On the other hand, all parties in a solving team must be sufficiently compatible that they can collaborate effectively. They must either speak/write in a common language or have sufficient language interpretation services that they do not misunderstand each other too frequently.

1) Supporting alternative work styles: Luther and Bruckman found that online work groups tasked with creating original movies were difficult to lead [10]. Part of the difficulty stemmed from the artist culture of wanting to release only perfected work, and artists wanting ownership in the product of their work. On the other hand, with the advent of web programming, much of software engineering focuses on a collaborative launch-and-iterate approach. The general issue is how to accommodate such differences in work preferences.

2) Avoiding biases related to gender, race, and other factors: Automated agents today exhibit some troubling biases. A case in point is the failure to distinguish images of various primates from people. One way to avoid the bias is to remove the technical features that exhibit the bias. In the meantime, researchers are trying to reduce the degree of bias in agents [18].

3) Designing to encourage beginners and participants from varying fields and backgrounds: How can tools and systems encourage new solvers? By providing specific roles that newcomers can easily fill, some of the perceived barriers to participation can be removed [4].

4) Making problem-solving support tools accessible to people with disabilities: Recent work in the design of coding interfaces for visually impaired programmers suggests that new interfaces to support blind problem solvers could be created [12]. Special features to help visually-impaired users locate blocks on a screen might be a key part of such an interface.

F. Incentives

Successful collaborations require that all participants have incentives to participate. Participants may be intrinsically motivated to solve a problem that they personally care about, or because they wish to learn more about a problem, but there can also be extrinsic motivators such as monetary incentives (hourly pay, prizes for winning accomplishments, lotteries, etc.) or social recognition. Some of the subissues relating to incentives are ownership, credits/accounting, and recruiting.

1) Ownership: The results of a problem solving activity may include solutions, partial solutions, constructed objects, or curated information gathered from the Internet. Intellectual property rights related to collaborative problem solving systems need to be spelled out, so that participants feel that they are being treated fairly by the community they join, and by the system.

2) Credits/Accounting: The credit that a person receives from work performed may be monetary or handled in other forms. If it is accounted for numerically, then appropriate mechanisms must be in place. They should be transparent, so that a user knows how s/he is being treated. If credit is handled in a non-numerical way, such as by citing usernames within each object or document created or added to by a user, then the rules for awarding such bylines should be clear.

3) Recruiting: The quality of problem solving activity within a human-user system will clearly depend on the abilities and attitudes of its users. A system is more likely to be successful if it supports guidelines for recruitment. Recruitment may include discovery and communication with prospective users, means for describing needed skills and task requirements, as well as means for allowing a group of solvers to make pitches to potential new members. System support for provisional group members can enable one approach to recruiting.

G. Collaboration

The mechanics of group work involves another set of issues that are not unique to problem solving but are nonetheless important in such work, and collaboration in problem solving may require that standard techniques be adapted.

1) Workflow management: The particular ways that problem-solving tasks get broken into subtasks and shared among multiple human solvers are important in shaping the experience of the group. For example, the balance between independence and close coordination may need to be adjusted both for the problem being solved and the preferences of the group members.

2) Decision-making/authority: Social structure within a team may be flat or hierarchical. Systems may support various structures and provide tools for voting, reaching consensus, representing group structures, etc.

3) Awareness: In group collaborations, it can be helpful for each user to be aware, to some degree, of other users' activities. For example, what other users are logged in, what work they've done, what they are working on, and any events that might affect one's own activities during the session. How such affordances such as notifications are shown and managed can be important in maximizing productivity and satisfaction.

4) Access to shared objects: In a system such as CoSolve, it is easy to imagine a situation where a solver is working on a formulated problem, but a poser is editing the formulation. There are many possible cases in problem solving where two users may wish to edit the same object but inconsistencies need to be avoided. Flexible mechanisms are needed that permit users to be maximally productive.

5) Quality control/reviewing: Mechanisms are needed to help team members evaluate their own work as well as the work of their teammates. Evaluations may involve technical tools for testing and validation as well as qualitative descriptions.

H. User training and problem-solving education

Because solving complex problems may require a great deal of information about a problem as well as about how to solve complex problems, a suitable problem-solving system must either internally provide tools for learning to problem-solve, or it must rely on external training resources.

1) Embedded tutorials vs external courses.: A first question to ask when developing the learning materials is what parts of the tutorials or resources to include inside the system and what parts to make external. Internal learning aids can be more closely integrated with other interface tools or problem elements. External aids are easier to provide, they can have varying formats, and they may be easier for new users, who don't yet know how to access internal resources, to access.

2) Embedded tutorials about particular problems: How should such tutorials be a part of the system? Assuming that the tool is domain-independent but the sessions and posed problem are domain-dependent, links from the session to external tutorials that are problem-specific should be created by posers and lead solvers. Such links could be embedded in comments on objects within the posed problem or within a solving session. They could also be inside a chat history that is attached to the solving session.

3) External materials about particular problems: External materials may require some kind of cross-referencing within the system, so that users can transition easily between internal situations and the external resources. How should these links work and what can a system designer do to facilitate their creation and use?

4) Embedded tutorials about problem solving and the use of the tools: One of these could simply be an integrated help system for the tool. Possible technical affordances that facilitate tutorial integration are location codes (within solving-session objects), context snapshots, and the automatic detection of teachable moments when a pattern of user activity indicates that a particular lesson might be pertinent. Another possibility is for the system to help lead users or instructors to construct lessons from "real world" experience – that means work already done by the learner-user or by others on solving the current problem.

5) External materials about problem solving and collaborative approaches: More general educational background on problem solving and collaboration might be better provided using external resources such as Youtube videos, articles, books, and exercises.

I. Visualization

Whereas a fully automated solving agent using a computer algorithm to solve a pre-formulated problem might not have any use for a visualization of any kind, when human solvers are involved, visualizations become essential. Human understanding of a complex problem space can be greatly helped using appropriate visualizations.

1) Visualization of Background Knowledge for a problem: Maps, charts and interpretive diagrams can help new users become familiar with problem background. Visualizations can be particularly helpful during problem formulation when collaborative sensemaking is used to analyze information resources and develop problem representations [11]. The use of large, high-resolution displays can facilitate such processes [1].

2) Visualization of problem states: A problem state typically represents the arrangement of elements of a solution after some steps towards the solution have been taken. Portraying the state of a problem with diagrams such as plots and charts, or image renderings, can be very helpful in explaining what has been accomplished.

3) Visualization of problem space: The entire space of possible states of a problem can sometimes be shown visually, although often in greatly simplified form. Such a display can help users understand the relationships among various states. Here, graph layout, choices of axes, projection and remapping, infinite graphs, and interactive display of graphs are relevant techniques.

4) Visualization of the work done to solve a problem.: As a group works on a problem, simply keeping track of the group's progress becomes nontrivial. Relevant techniques include maintaining and showing session histories (e.g., with tree diagrams), showing the best path through state space found so far, and plotting of landmarks (in the problem space) created by members of the solving team.

J. System Integration

1) Roles for information technologies: This is a broad issue, and we are lumping together the many different information technologies so that we have space to call out the issues above.

Where do various information technologies fit into a system that supports collaborative problem solving? Particular technologies include numerical algorithms, artificial intelligence techniques (logical inference, pattern recognition, language understanding, recommendations, searches, etc.), machine learning, collaborative editors, chat and conferencing, version control, visualization, database management, user account management, security, and others.

K. Problem-specific Issues

Particular problems or categories of problems may have somewhat unique issues associated with them. For example, environmental problems typically have a geo-spatial aspect that may call for an integration with geographic information systems (GIS). We mention a few categories here to illustrate the general issue.

1) Environmental: Such problems include global climate change, ocean acidification, sea-level changes, air pollution, water supply. As mentioned above, these share a geo-spatial aspect.

2) Health: Drug-resistant diseases, affordable health care, health education. Such problems share a possible need for a system of human values related to health care, such as the notion that access to basic health care is a human right.

3) Politics, War, Nuclear Proliferation: Problems related to violence and war have their own issues including the human costs of suffering, deaths (due to small-scale crime, genocide, or apocalyptic war) and systems of human beliefs.

4) Engineering of Complex Systems: Software and cyber-physical systems can be sufficiently complex that some of the problem-solving techniques referred to earlier are appropriate. Debugging alone can require problem-solving skills and tools.

5) Other: Fake news is just one example of a problem that does not fit well into the four categories above. It has its own set of issues related to the nature of truth, and the goals of various parties such as news media, and power brokers.

V. Research Prioritization

This section considers the relative importance of some of the issues discussed earlier.

A. What issues are most important?

The answer here depends on several factors, including which problems need to be solved first and who is ready to engage in solving them, as well as what tools are being considered for modification or a complete design and build.

1) Prioritization of Problems: The relative importance of problems themselves can be considered from several points of view, such as time frame, numbers of people affected, and available opportunities. Time frames run from short-term (needing solutions very soon) to long-term (needing solutions a few years from now). Averting an impending asteroid strike is a problem that may come with a very accurate deadline. Slowing global warming is perhaps just as pressing but without such a clear deadline. Achieving a culture of problem solving through educational curriculum changes may be very desirable, but it seems to fit into a long-term time frame.

Problems that affect larger numbers of people may be considered as higher priority problems.

B. Finding frameworks that can be used to integrate services and solutions to the issues

If existing tools are to be extended to address particular issues identified here, then the existing tool may impose the framework for such additions. Tools such as Jupyter notebooks, SAGE math sessions, CoSolve, or automatic solvers that work with PDDL or its variants [19] are possible candidates for extension, but there are doubtlessly others.

References

[1] Christopher Andrews, Alex Endert, and Chris North. “Space to think: Large, high-resolution displays for sensemaking.” Proc. CHI 2010. Atlanta, GA.

[2] Jeffrey Conklin, Jeffrey. *Dialogue Mapping: Building Shared Understanding of Wicked Problems*. Chichester, England: Wiley Publishing. ISBN 0-470-01768-6. 2006.

[3] Sandra B. Fan, Tyler Robison, and Steven L. Tanimoto. ”CoSolve: A System for Engaging Users in Computer-Supported. Collaborative Problem Solving.” Proceedings of VL/HCC 2012, Innsbruck, Austria.

[4] Sandra B. Fan. *CoSolve: A Novel System for Engaging Users in Collaborative Problem Solving*. Ph.D. dissertation, University of Washington. 2013.

[5] Álvaro Figueira and Luciana Oliveira. “The current state of fake news: challenges and opportunities.” https://doi.org/10.1016/j.procs.2017.11.106

[6] Tim Gowers, “Can Polymath Be Scaled Up?” 2009. [Online]. Available: http://gowers.wordpress.com/2009/03/24/can-polymath-be-scaled-up/

[7] John Jackman, Sarah Ryan, Sigurdur Olafsson, and Veronica J. Dark: “Metaproblem Spaces and Problem Structure” in D. Jonassen (ed): *Learning to Solve Complex Scientific Problems*. Lawrence Erlbaum, 2007.

[8] Koios.com website. Accessed 13 July 2018.

[9] Denis Lalanne, 1999. *Conception créative assistée par ordinateur: un modèle d’intelligence interactive*. Ph.D. dissertation. (in French).

[10] Kurt Luther and Amy Bruckman. “Leadership in online creative collaboration.” In Proceedings of the 2008 ACM conference on Computer supported cooperative work (CSCW ’08). ACM, New York, NY, USA, 343-352. 2008.

[11] Narges Mahyar. *Supporting Sensemaking during Collocated Collaborative Visual Analytics*. Ph.D. dissertation, University of Victoria, BC, Canada. 2014.

[12] Lauren Milne. *Touchscreen-Based Learning Technologies for Children who are Blind or Have Visual Impairments*. Ph.D. dissertation, University of Washington. 2018.

[13] Sharoda A. Paul and Madhu C. Reddy. “Understanding together: Sensemaking in collaborative information seeking.” In *Proc. ACM Conf. on Computer Supported Cooperative Work*, pp.321–330. ACM, 2010.

[14] Horst W. J. Rittel, and Melvin M. Webber. “Dilemmas in a general theory of planning.” *Policy Sciences* (June 1973), Vol. 4, Issue 2. pp.155-169. https://doi.org/10.1007/BF01405730.

[15] Daniel M. Russell, Mark J. Stefik, Peter Pirolli, Stuart K. Card. “The Cost Structure of Sensemaking.” *Proceedings of INTERCHI’93*, pp.269-276.

[16] Herbert Simon. *The Sciences of the Artificial*. Cambridge, MA: MIT Press. 1969.

[17] Herbert Simon, and Allen Newell, “Human Problem Solving: The State of the Theory in 1970.” *American Psychologist*, 1971, pp.145-159.

[18] Tom Simonite. “When it comes to gorillas, Google Photos remains blind.” https://www.wired.com/story/when-it-comes-to-gorillas-google-photos-remains-blind/. 2018.

[19] Volker Strobel and Alexandra Kirsch. “Planning in the Wild: Modeling Tools for PDDL”. In *KI 2014: Advances in Artificial Intelligence, Proceedings of the 37th Annual German Conference on AI*, Stuttgart, Germany, September 22-26, 2014.