Human Computation and Serious Games: Papers from the 2012 AIIDE Joint Workshop
AAAI Technical Report WS-12-17

Metaphor Computing

Dan Fu
Stottler Henke Associates, Inc.
1670 S. Amphlett Blvd, Ste 310
San Mateo, CA 94402
fu@stottlerhenke.com

Matt Bishop
Department of Computer Science
University of California at Davis
Davis, CA 95616-8562
mabishop@ucdavis.edu

Abstract

We define metaphor computing as a way to transform difficult computational problems into easier human-solvable problems, and transform solutions back into computational solutions. This report explores initial ideas.

Introduction

Training for today's workforce is a key ingredient for ensuring success for tomorrow's business. The development of critical skill sets results in productive personnel able to handle complex problems. As the complexity of problems we encounter increases in step with industrial growth, there has existed a demand for an ever more sophisticated workforce. To improve productivity, especially with regard to large software systems, interface designers are continually challenged to study activity and make the user's interaction as simple and efficient as possible. In this report we do not advocate ways to make training more effective for specialized skill sets. In fact, we advance the opposite: that we should examine what people are good at, and leverage those skills to solve complex problems. Unfortunately, what most people are good at has little to do with solving complex problems. However, it may be possible to transform a complex problem into a set of simpler problems that most people can solve, and then map the solutions back to a solution for the complex problem.

Consider research work on linguistic metaphor. Metaphors are pervasive (Lakoff and Johnson, 1980). Whether we talk about panes, windows, screens, or firewalls at the local Home Depot or Apple computer store, people move fluidly among these multiple metaphors to understand the world and to act effectively. Rather than relegate metaphor to an ill-fitting paradigm of idiom creation, we advocate metaphor taking center stage by providing the key insights that enable people to solve hard problems. Indeed, the fundamental motivation for using a metaphor is to articulate a new concept by using an already familiar one. So then, we ask, why not use the familiar one in the first place?

The innovation afforded by a computational form of metaphor enables less sophisticated workers, already trained in a given occupation, to leverage their skills to solve problems in different domains that to them appear very difficult. As an example, look at the problem of computer network intrusion. Highly trained personnel and sophisticated software are necessary to protect the network—but what if metaphor software could transform the network into a virtual environment such as a base with a protective perimeter, security gate, buildings, rooms, doors, locks, sensitive areas, and so forth? Then the protection of the network amounts to defending this "virtual fort." Detecting intrusions into the network amounts to recognizing suspicious individuals and activities. Subduing the individual might be mapped to suspending activity for an unverified user or software agent in the network. We suggest this metaphor because protecting a computer network as if it were a castle has long been used in undergraduate computer science classes (Frincke and Bishop, 2004). Thus, most people are familiar with the metaphor and can leverage this already-understood knowledge. If computational metaphor transformations are successfully realized, there would be fewer training requirements on personnel. Security guards are a lot easier to hire and train than white hat hackers. Less time would be expended on training.

Using this technique, a lightly-trained force could be quickly mobilized to defend against cyber attacks. Consider the attack on Estonia's network infrastructure on April 26, 2007 (Evron, 2008). Russian public forums discussed attacks for days before the Estonian attacks. In fact, instructions on how to attack the Estonian infrastructure were published in these forums, potentially enabling anyone with an Internet connection to participate in the attacks. This is in contrast to the average Estonian citizens who might have had an interest in fending off the attacks. They were powerless; indeed, even though Estonia is one of the most Internet-integrated countries in the EU, citizens could only rely on their CERT team to protect them. One might even assume that a portion of their own computers were used against them.

This example describes an equally important property of metaphor. Most people know how to protect their home: lock doors and windows when you leave, don't leave a key under the mat by the front door, etc. If this metaphor can be translated such that protecting a home network or system required the home user to "lock" a virtual house, then people who have no security training, and, in all probability, would actively resist such training, can provide their system or network with basic protection. The intuition involved makes securing the system less difficult, and at the same time more acceptable.

Problem Transformation

Consider a two player game where each player, in turn, chooses a number from 1 to 9. Numbers cannot be repeated. The first player to have uttered any 3 numbers that add to 15 wins. This problem is analogous to identifying a legal row, column or diagonal on a magic square. Do you recognize this game? It's tic-tac-toe. The two games are isomorphic: being good at one game translates into being good at the other. Although a contrived example, it embodies what we'd first like to characterize; namely, classes of problems that can map to other problems. In the ideal case, we'd discover classes of problems that are difficult for machines, but easy for humans.

We posit that there exist problems that are hard for machines (machine-hard or M-hard), but easy for people (people-easy or P-easy). We believe these problems, when represented formally, are amenable to transformation from M-hard to P-easy, but further, that the P-easy solution can be transformed back into an M-hard solution. Figure 1 is an informal chart of problems. An initial objective is to scope and characterize these spaces formally so that we can understand what makes a problem difficult to solve. Most importantly, we want to mine a space of problems in the lower right hand quadrant that are M-hard but P-easy. Ideally we would discover classes of real world problems that can be associated with as yet undefined classes of "metaphorical environments". This is a catch-all term for where people can perform P-easy tasks. One can think of the environments as being virtual worlds specifically crafted to help solve M-hard problems.

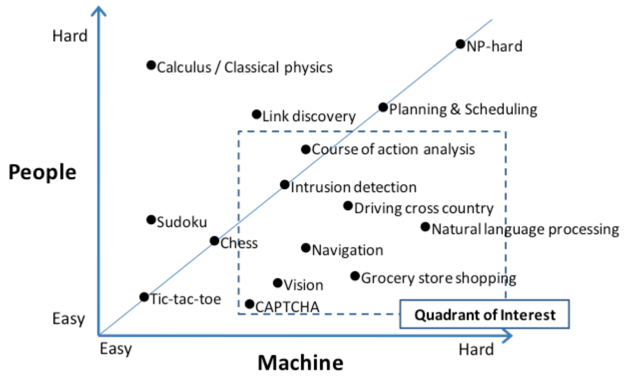

Figure 1: Chart of problems to examine.

Lakoff and Johnson (1980) have identified several types of metaphors. For example, one can use a building; e.g., when talking about a research paper, "I didn't like the façade, but the foundation was firm," or, "They buttressed their argument with solid references." One could use a transfer metaphor; e.g., "They conveyed their concepts well," or, "The few good ideas were buried in an avalanche of jargon." The point being that these metaphors use physical concepts to facilitate understanding. Because these concepts are rooted in the real world—perhaps the gist of which a five year old could understand—we believe it's possible to automatically fabricate virtual worlds in which a layman could work.

Figure 2 shows an idealized process where we have a formal description of the real world problem which then gets transformed into a virtual world problem. The transformation results in a problem description consisting of a virtual environment and set of user goals & tasks. The environment consists of three elements:

1. Virtual 3-D Terrain: Holds the environmental data in which the user operates. This should resemble a real environment such as a house;
2. Actions: These dictate the ways in which objects interact in the environment. The user's ability to act or manipulate objects is defined here;
3. Perception: A depiction of the environment.

The user's "goals & tasks" are what will occupy the user during a problem solving session. For instance, the user may traverse the grounds of a house, searching for anything out of the ordinary. That could be in an obvious form, such as greeting a visitor knocking on the front door. What the user does next will affect not only the virtual world, but also the real world. In the case of the Estonian attack, citizens could at minimum "patrol" their own computers to prevent unauthorized outgoing DoS attacks.

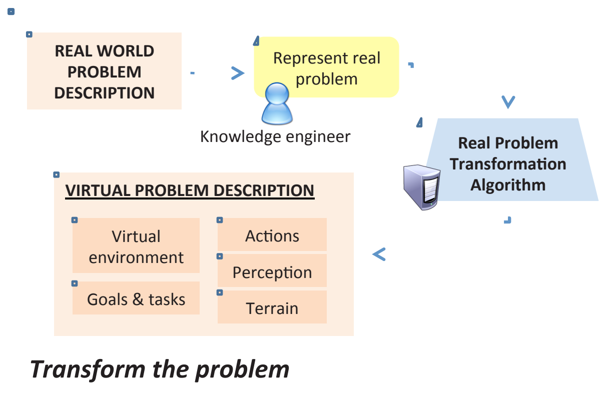

Figure 2: Transforming a real world problem into a virtual problem.

Metaphor Implementation

Once a virtual problem description has been defined, we must examine how to make the metaphor operational. Figure 3 illustrates this process. On the top portion there is the relation between the real world and the virtual world, linked together via a transformation algorithm. The real world has effectors and sensors. The effectors cause tangible change in the real world, while the sensors report on real world state. The transformation algorithm maps sensor information into the virtual world. Actions (events) in the virtual world drive the effectors in the real world. On the lower right there are the user who performs tasks and the interface elements. The user acts using the monitor, keyboard, joystick, etc.

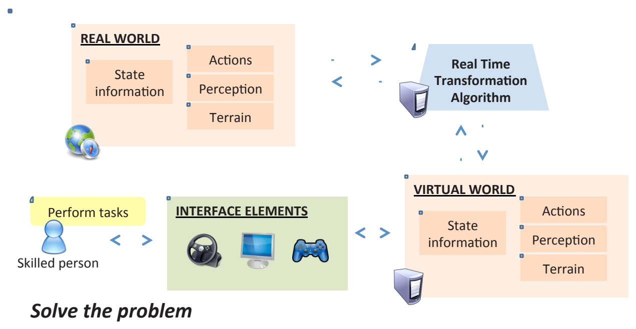

Figure 3: Mapping virtual world problem to real world problem.

Using the security example, an effector action could be to block a port, while a sensor logs an attempted connection to the port in question. In the virtual world, the event driving the blocked port is likely a direct action by the user. If a castle, then the action could be to lock a door.

Related Work

Several bodies of research informed the work as described by Fu and Bishop (2009) along with more recent advances.

Few security researchers have explored 3-D technologies for visualization of network activity. Fisk et al. (2003) describes a method for animating network traffic by constructing a 3-D "shield" that one defends from external connections. Fisk observes the use of metaphor provides a comfortable level of abstraction for observers to talk meaningfully. Von Ahn et al. (2003) through the use of CAPTCHA distinguish between human and computer capabilities for protection against automated misuse.

The field of human computation studies the interaction between human and machine for solving problems, each with their own set of capabilities. For example, von Ahn et al. (2008) describe a method for solving optical character recognition discrepancies by embedding them into CAPTCHA challenges. The solutions are aggregated across several human responses to achieve a high rate of recognition. A related field is crowdsourcing which poses a problem to a distributed, anonymous group of one or more humans to solve. Amazon's "Mechanical Turk" is a marketplace for humans to be paid by solving "Human Intelligence Tasks" which vary in complexity from writing a website article to tagging images. Foldit is a successful multiplayer videogame (Cooper et al., 2010) for solving a molecular biology problem of predicting protein structures. The game makes use of player's spatial capabilities to solve puzzles, some of which have been significant. Dietl et al. (2012) describe a crowdsourcing game "Pipe Jam" for computer program verification. They do this by transforming a program and security property into a videogame level whose solution (if it exists) can prove partial correctness. These last two approaches transform the problem into a more easily understood representation for players.

Case-based reasoning (CBR) attempts to solve a problem using a previously solved past problem (Kolodner, 1993). A problem is retrieved from a case base, mapped to the current problem, tested, and possibly added to the case base. CBR has historically been applied to narrow domains. Whereas the foregoing work speaks to metaphor implementation and refactoring of results, CBR is also relevant to problem transformation since it must explicitly link the current problem to the past problem, and then leverage that case to generate a solution based on the past solution. If one were to automatically generate videogames on demand—i.e., transform a real world problem description into a virtual world—there is the prospect of automatic game design and development. This is a nascent area that has received little attention in the research community. A scattershot of efforts have presumed a narrow game domain—e.g., chess-like turn taking (Pell, 1992)—or focused on game design ontologies as a precursor (Nelson and Mateas, 2007; Zagal et al., 2005). Work has been done in the areas of procedural generation: creating the art content one would visually see, and procedural narrative: directing the narrative that one experiences (Newell, 2008). Analogical problem solving (Gick and Holyoak, 1980) is related to CBR, but more concerned with cross domain relationships. It is a psychology discipline that relates one problem to another, relying on semantically deep structural similarity (Falkenhainer et al. 1986; Gentner, 1993). For example, the physics of heat flow can be understood in terms of fluid flow by mapping temperature to pressure and metal conductor to tube.

Conclusion

This report lays out some basic ideas for metaphor computing. As we've argued, the chief benefit is that relatively unskilled personnel, such as a high school age videogame players, could be used to perform complex, skilled tasks at the level of a network administrator or information assurance red team. Rather than present neat solutions, this report raises several questions of whether metaphor computing is really possible. We view these questions as basic precursors to finding an answer:

1. Can real world problems be represented to permit automated 1-to-1 transformation?
2. Which metaphors will work best for a given person or problem? Though it might seem intuitive, when was the last time you protected a castle?
3. Current real worlds aren't mature. Events in the real world require sensor output to map into the virtual world, and conversely, actions in the virtual world require effectors in the real world. It's likely that for now the real world must be a software-driven world.
4. The security purpose could mean lives are at stake. Witness the unwillingness to adopt telepresence for gunnery—an instance where the person actually knows what they're doing.
5. A metaphor will prove brittle if and when it breaks. Graceful degradation of performance may not be possible. Orchestration and transition between multiple metaphors is likely.
6. Users will not know exactly what they're doing. Two things to infer are that (1) pulling a user further away from the raw, complex problem may preclude insight and innovation; and (2) there are ethical considerations.

References

Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., Leaver-Fay, A., Baker, D., Popovic, Z., and Foldit Players. (2010). Predicting protein structures with a multiplayer online game. Nature, 466(7307): 756–760, 2010.

Dietl, W., Dietzel, S., Ernst, M. D., Mote, N., Walker, B., Cooper, S., Pavlik, T., and Popovic, Z. (2012). "Verification games: Making verification fun," in FTfJP'2012: 14th Workshop on Formal Techniques for Java-like Programs, (Beijing, China), June 12, 2012.

Evron, G. (2008). Battling Botnets and Online Mobs: Estonia's Defense Efforts during the Internet War. Georgetown Journal of International Affairs. Winter/Spring 2008.

Falkenhainer, B., Forbus, K. D., and Gentner, D. (1986). "The structure-mapping engine," in Proceedings of the Meeting of the American Association for Artificial Intelligence, 272-277.

Fisk, M., Smith, S.A., Weber, P.M., Kothapally, S., and Caudell, T.P. (2003). "Immersive network monitoring," in Proceedings of the 2003 Passive and Active Measurement Workshop.

Frincke, D.A., and Bishop, M. (2004). Guarding the Castle Keep: Teaching with the Fortress Metaphor. IEEE Security and Privacy 2(3): 69-72.

Fu, D., and M. Bishop. (2009). Metaphor Computing. Stottler Henke Technical Report. San Mateo, CA. August 10, 2009.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. Cognitive Science, 7, 155-170.

Gick, M. L., and Holyoak, K. J. (1980). Analogical problem solving. Cognitive Psychology, 12, 306-355.

Kolodner, J. L. (1993). Case-based Reasoning. San Francisco, CA: Morgan Kaufmann.

Lakoff, G. and Johnson, M. (1980). Metaphors We Live By. Chicago: University of Chicago Press.

Nelson, M. J., Mateas, M. (2007). "Towards Automated Game Design," in Proceedings of the 10th Congress of the Italian Association for Artificial Intelligence, pp. 626-637.

Newell, Gabe (2008-11-21). "Gabe Newell Writes for Edge". Edge. Retrieved 2012-05-29.

Pell, B. (1992). METAGAME in Symmetric Chess-Like Games. H.J. van den Herik and L.V. Allis, editors, Heuristic Programming in Artificial Intelligence 3 – The Third Computer Olympiad. Ellis Horwood.

von Ahn, L., Blum, M., Hopper, N. J., and Langford, J. (2003). "CAPTCHA: Using hard AI problems for security," in Lecture Notes in Computer Science -- Advances in Cryptology, E. Biham, Ed., Vol. 2656, Berlin, Germany: Springer-Verlag, 2003, pp. 294-311.

von Ahn, L., Maurer, B., McMillen, C., Abraham, D., and Blum, M. (2008). reCAPTCHA: Human-Based Character Recognition via Web Security Measures. Science, September 12, 2008. Pages 1465-1468.

Zagal J.P., Mateas, M., Fernandez-Vara, C., Hochhalter, B., Lichti, N. (2005). "Towards an Ontological Language for Game Analysis," DIGRA 2005.