Some Agent Theory for the Semantic Web

Leona F. Fass
P.O. Box 2914, Carmel CA 93921
<lff4 [AT] cornell [DOT] edu>

Abstract

We take the position that for any goal achievable on the Semantic Web, there will be a "best" system of Web-dwelling software agents to realize that goal, and that such a system may be discovered effectively. The process of determining the "best" agent system may be overseen by a distinguished Manager Agent. But with realistic time and space constraints, and the dynamic nature of the Semantic Web, finding an approximating system may be acceptable. The approximation then may be adapted iteratively, to approach the ideal. We show that very practical researchers have looked at software agents and Semantic Web problems in a similar way, determining approximating sub-optimal systems and subsequently adapting them. Their applied research confirms that theory provides a good foundation for practice.

Keywords: Software agents, multi-agent systems, Web services, the Semantic Web, theoretical foundations.

Introduction

The aim of Semantic Web designers and developers is to enhance the existing Web, supplying structure and meaning that will facilitate machine-to-machine interaction. With agent and multi-agent system research and development integrated into Web development, it should be possible for software agents to utilize the Web environment, fulfilling complex tasks that human users have specified. Given a human-specified task (e.g., "Schedule my appointments for the East Coast trip", or "Get me contact information for the Cornellians and Penn grads who live in this area") capable agents could produce a satisfactory result. They would access machine-readable information, discover fulfilling Web services and automate their use, making logical decisions and invoking and composing services as needed.

In inceptive articles about the Semantic Web James Hendler [13] and Tim Berners-Lee, James Hendler and Ora Lassila [2] described some of its proposed features including languages, communicating ontologies and inferencing mechanisms. They provided illustrative examples of "agent-based computing" and the role that agents can play in Web applications, interacting with each other and with human users. There is typically an Artificial Intelligence emphasis on Semantic Web and agent research. Agents often are depicted as lifelike, as they may navigate the Web and interact with other agents or with people. (The latter aspect also brings a Human-Computer Interaction perspective to the field.) While they may communicate, learn and adapt, these agents are really just software. And so, as well, is the Semantic Web. Hence the development of the Semantic Web and Web-navigating agents must involve substantial Software Engineering. In a project as massive as the Semantic Web there is room, and a need, for multiple and coordinated research emphases.

We ourselves take an automata theory approach to agents and the Semantic Web, motivated by some intriguing software agent- and Web-related research we identify as theoretically grounded. Thus we look at tasks to-be-fulfilled or problems to-be-solved on the Web as behaviors to-be-realized. Viewing the Semantic Web itself as a multi-agent system we look at sub-systems constructed of Web-dwelling agents to fulfill specific tasks, or to solve specified problems, as "behavioral realizations". With this very theoretical perspective we propose that any goal achievable on the Semantic Web will have a "best" multi-agent system behavioral realization. We take the position that this realization may be constructed or adapted effectively from Semantic Web-dwelling software agents, perhaps by a distinguished Manager Agent. The system agents may invoke and compose Web services, as needed, to achieve behavioral goals.

Since this theoretical result may disregard real constraints on time and space, we next discuss the necessity of accepting approximating Semantic Web-dwelling multi-agent systems. These can adequately fulfill tasks or effect problem solutions that are "good enough", relative to time, space, and information they may obtain. We describe adaptation of such approximating systems, when tasks, goals, information, or available components change, or when anomalies or defects are found.

Despite the abstract approach we take when considering these issues we find other, very practical, researchers have looked at multi-agent systems and the Semantic Web in a similar way. We describe their related work with multi-agent system managers, approximating results, iterative approaches to the "optimal", and constraints applied so that features of the dynamic Semantic Web can be represented to software agents effectively. Certainly, we believe that there are problems for which optimal results may never be found. But we also believe, from the practical work we have seen, that when considering agents and the Semantic Web, theory provides a good foundation for practice.

Research Motivating Our Approach

We were motivated to consider theoretical problems related to the Semantic Web and Web-dwelling software agents from our analysis of some very interesting research in several related areas. These included fairly recent work in: emergent semantics; automated composition of Web services; deduced interaction; collaborative learning; agent coalition formation; and an agent's locally-closed view of its world. Each bears some relation to our automata theory view of behavioral modeling, and each has convinced us to extend our theoretical work into problems of the developing Semantic Web .

In describing emergent semantics Steffen Staab [22] noted the possibility that semantic links to Web pages could be inferred from observations of actions of human users. Luc Steels [23] described knowledge representation for agents, so that *they* could interact. Equivalent external representations that agents needed to communicate did not imply equivalent internal representations. In our own inductive inference research (such as described in connection with model-based validation [5]), we had shown that the structure of a language (or device) could be inferred by observing examples of behavior. We'd also shown that behaviorally equivalent results, with different underlying structures, could be found. Thus we were interested in the concept

of inferring semantics, and additional features enabling diverse agents to communicate on the Semantic Web .

We learned about aspects of knowledge representation for the Semantic Web from Sheila McIlraith [15], who discussed language and logic needed to effect automated composition of Web services. The goal of such efforts has been to make Web services agent-oriented, rather than human-oriented, so that invocation of services can be automated. Then software agents could discover information, execute processes, and make selections of actions to take. The satisfying results would be service compositions and interoperability. This seemed to us a very natural way to look at problem solutions: decomposing problems into more-easily-solved sub-problems; finding component sub-solutions; recomposing components into the solution that results.

That solution components may *already* be available on the Web was shown to us by Richard Waldinger [25]. He discussed deductive techniques to find agents appropriate for solving sub-problems, and to compose them into groups that effect problem solutions. His method employed a theorem-prover that, given available agent capabilities as axioms, produced a theorem that "glued" appropriate agents together to achieve a behavioral goal. He showed that this process may *also* detect anomalies and information inconsistencies on the Web. In the theoretically-oriented research we have conducted, we'd first prove that a problem solution (i.e., specified behavioral realization) did exist and then, that it could be constructed effectively from representative components. Thus we could see similarities in Waldinger's work. When, in our work, we'd had what we believed to be a realization (or an approximation) and tested it against a behavioral domain we, too, might detect anomalies. In our case we used the information to correct defects and find better behavioral representations. As "theoretical" as such research may seem, it has important real world applications. E.g., the deductive anomaly detection research described by Waldinger in a symposium on logic-based program synthesis [25] was funded by NASA (the National Aeronautics and Space Administration), DARPA (the Defense Advanced Research Projects Agency) and ARDA (the Intelligence Community's Advanced Research and Development Activity). We have been encouraged to find such recognition of the fact that practical domains can benefit from theory.

Research involving collaborative learning agents, and describing formation of agent coalitions, showed us that software agents can act together in groups and adapt, approaching a behavioral ideal. Many different agent collaboration techniques described in [24] might achieve this. All are more or less goal-directed learning methods, determining "best states" and actions to chose. Software agent coalitions defined by Leen-Kiat Soh and Costas Tsatsoulis [21], and those discussed throughout [20], are formed so that groups of agents can work together solving particular problems or fulfilling particular tasks. These agents might use case based reasoning and reinforcement to evaluate utility of working together and might "argue", "negotiate" or even "coerce" each other into joining the task-oriented group. An initial sub-optimal coalition may be formed, and then refined to approach an optimal result. We, too, have iteratively constructed behavioral models that aim to achieve a goal in the "best possible" way, and have described feasibility of adaptations to software design [5]. Additionally, we have applied our own theoretical approach to forming coalitions of software agents [7]. Here again there are very practical applications: Soh and Tsatsoulis [21] have described real time coalition formation, with results applied to multi-sensor target tracking.

Finally, a somewhat theoretical view of a real Web condition is the agent's local closed world (LCW) view of the Semantic Web, described by Jeff Heflin and Hector Munoz-Avila [12]. The Web is an enormous and dynamic domain, and a software agent really cannot know when it has completed an adequate search for information. The addition of LCW extensions to a Semantic Web language could provide Web-navigating software agents with sufficiently complete information to conclude they'd done "enough". This work has interested us because we found, in our own work, that perfect results could be obtained in theoretical problem domains *only* because such domains could be constrained. But in real problems without domain constraints, we determined that approximate results could be acceptable results, for they might be "good enough" [5-8].

We recognized the relationships between software agent research and our theoretical work, and between Semantic Web research and our theoretical work, as we have described above. Noting similarities, we decided to apply our automata theory perspective to agents and the Semantic Web. We are not designing the Semantic Web or constructing it ourselves, but we believe there are applications of our theoretically-oriented work to Semantic Web development. This is particularly the case in connection with mechanisms for service invocation, coordinated communities, discovery and selection of services and choreographies, interaction protocols and architectures supporting agents and services within the Semantic Web. We are looking at problems similar to those considered by agent developers and Web designers; they and we just use different terminologies. In our automata world, these problems are all problems of determining the structure of a behavioral realization that will achieve a specified behavioral goal.

The Automata World and the Agents World

In our view, the agents world and the automata world are very much alike. Both worlds involve taking complex problems, decomposing them into simpler sub-problems, determining or constructing realizing sub-solutions, and recomposing the results into the complex problems' solutions. Both worlds involve taking a complex behavioral goal (the problem) and finding a behavioral realization achieving that goal (the solution).

Now, in the automata world, solvable or (finitely) realizable problems are the behaviors of (finite) systems or devices. Device components are "states" connected into sub-systems/devices; these correspond to solutions to sub-problems. Reaching a "final state" corresponds to determining a solution to a problem. In our analogous view of the agents world, agents solving sub-problems correspond to "states". Agent interactions are like state-to-state transitions. And, when a group of software agents or multi-agent system solves a problem or completes a process, it is as if that group has reached a "final state". Thus synthesizing an automata-theoretic system or device that realizes a specified behavior is equivalent to composing a group of agents into a multi-agent system that produces a behavior, or fulfills a task. It is well known in automata theory that there is a relationship between the components of a realizable behavior and the structure of its

realization. Behavioral elements can be grouped (based on experiments that determine their "indistinguishable behavior") into classes that correspond to the components or "states" of a realizing "device". Relationships among the behavioral classes define "state-to-state transitions". Classes of "correct" behavioral elements define final, or goal, states. In the automata world, just by analyzing the components of a specified realizable behavior or goal, one may determine a device that produces or achieves it. One may optimize a result by minimizing it and, for finitely realizable behaviors, one may find the optimal result effectively. Furthermore, if a finite potential realization is given, and the domain of all "correct" and "incorrect" behavior is known, one may conduct effective tests to detect anomalies (and iteratively correct them). Alternately, after testing and perhaps correcting, one may determine the realization to be optimal. The components of an optimal result are found quite easily, just by exploiting the relationship between behavior and structure.

Because of the similarities between the agents world and the automata world, we believe that well-known automata-based techniques can be adapted to software agents. Then suitable agents may be collected into groups that can solve specified problems effectively. By extending proven automata theory techniques to processes in the agents world, agents may be collected into interacting systems or "devices" that realize behavioral goals or fulfill specified tasks. Once again, this just requires determining components and architecture of an agent system from analysis of its proposed behavior.

We believe that automata theory methods can be applied to systems of agents so that optimal (e.g. minimized) agent systems will be discovered to fulfill specified tasks. Utilizing this theory: useless agents in a system may be discovered and removed; agent configurations behaving indistinguishably can be merged; and communication paths can be minimized. We believe that when a behavioral goal grows or changes, automata theory techniques will help collected agents (or, a multi-agent system) expand monotonically or adapt as needed, with amended architecture and/or different "final states". Additionally, successfully performing agent sub-systems can be reused within future configurations. On the other hand, a multi-agent system or agent group can be tested to detect anomalies, if a goal behavior and its complement can be finitely categorized. Once again, we propose this can all be done, just by exploiting the relationship between the goal behavior and the agent group, or multi-agent system, structure.

We have illustrated some of these claims with a Personal Travel Assistant (PTA): a collection of task-fulfilling software agents initially described as a theoretical construction in [7]. PTA components that may plan our possible trip to a symposium in Washington DC could be adapted quite easily from the components that planned our trip to the ICSE meeting in Edinburgh, Scotland, or to an Information Sciences conference in Salt Lake City. All results could be optimized with respect to our established criteria (e.g., our specific constraints on finances, diet and time). Components that plan our trips to meetings in San Jose would be useless, and might be removed to save PTA space. (We travel to San Jose by bus and train; the other specified destinations require taxi, airport van and plane.) We may not be able to finitely characterize all possible "correct" or "incorrect" travel behavior that the PTA might produce. But from our personal-experience tests, we know the PTA works "well enough". I.e., within time limits, with sufficient protein-rich energy bars, and without exhausting our travel budget, we have arrived safely at the respective destinations. The specified behavioral goals have been achieved.

And What About the Semantic Web?

If we consider the collection of software agents that may dwell on the (Semantic)Web, the Web *itself* is a multi-agent system. At least for our current purposes, that is how we view it. So, just like Waldinger's approach [25] of locating agents on the Web and deductively "gluing" them into problem-solving systems, our automata theory approach, too, can apply to the whole Semantic Web. When synthesizing a software agent "device" to fulfill some behavioral specification, agents will be collected from the Semantic Web agent universe. The very same possibilities we described above (merger, minimization, elimination, optimization, adaptation) that might apply to "one's own personal multi-agent system", need only be scaled up to apply to the whole Semantic Web. In actuality this is, of course, quite a scaling project. But it is all quite simple in theory.

We believe that the best way to approach this project is to develop a distinguished Manager Agent. This agent would oversee the selection, composition, and adaptation, etc., of the multi-agent systems constructed from the *other* software agents dwelling on the Semantic Web. The Manager Agent would assign agents to groups, based on classes of tasks and the agents' abilities. The Manager would experiment, to effect mergers and reductions. The Manager would learn of new agents and services and utilize them as warranted. Adaptation would occur as new tasks developed, too. In our inference work we had a theoretical informant (many call this an "oracle") who oversaw the construction and adaptation of behavior-realizing models. This seems the ideal theoretical solution for determining architecture of each goal-directed agent system that could be composed, or exist, on the Semantic Web. The Manager (acting as informant) would oversee, track, select, construct and watch the Semantic Web grow and change.

Of course there are time and space considerations relating to how this Manager would work. There would be obvious intractability in making "perfect" system plans. Real time solutions to most problems could not possibly be obtained. Not if the Manager must categorize classes of all Semantic Web-dwelling agent capabilities and collect them into systems fulfilling behavioral goals. Not if the Manager must find optimal structures formed from among *all* other agents. Even with a Local Closed World assumption, keeping a Manager in touch with "much" information as the Semantic Web grows, the job could not, in reality, be done. But in Good Old Fashioned Automata Theory we may establish the following *theoretical result*:

> For any finitely realizable behavioral goal achievable on the Semantic Web, there will be a "best" (perhaps minimal, or most efficient, or other designated optimal) finite system of Web-dwelling software agents to realize that goal. This system may be discovered effectively: by construction from existing agents; or by adaptation from an agent system that is already known.

The key words in the above assertion are "achievable" and "effectively". If the goal is finitely realizable there will be a finite