Comparative Analysis of Top–Down and Bottom–up Methodologies for Multi–Agent System Design

Valentino Crespi
Dept. Computer Science
Cal State Los Angeles
Los Angeles, CA
vcrespi@calstatela.edu

Aram Galstyan
Information Sciences Institute
Univ. of Southern California
Marina del Rey, CA
galstyan@isi.edu

Kristina Lerman
Information Sciences Institute
Univ. of Southern California
Marina del Rey, CA
lerman@isi.edu

ABSTRACT

Traditionally, top-down and bottom-up design approaches have competed with each other in Algorithmics and Software Engineering. In the top-down approach, design process starts with specifying the global system state and assuming that each component has global knowledge of the system, as in a centralized approach. The solution is then decentralized by replacing global knowledge with communication. In the bottom-up approach, on the other hand, the design starts with specifying requirements and capabilities of individual components, and the global behavior is said to emerge out of interactions among constituent components and between components and the environment. In this paper we present a comparative study of both approaches with particular emphasis on applications to multi–agent system engineering.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Design, Algorithms

Keywords

MAS design, Top Down, Bottom–up

1. INTRODUCTION

Traditionally, two different design methodologies, called top-down and bottom-up have competed with each other. In the top-down approach, the design starts from the top with the assumption that resources are globally accessible by each subcomponent of the system, as in the centralized case. The specification is then defined in terms of the global systems state and implies that each individual component should be able to retrieve or estimate, with sufficient accuracy and within a reasonable time delay, resources that are local to other agents of the system. Under these conditions the properties of a classical centralized solution to the global specification are expected to hold, up to some tolerable performance degradation, also in a decentralized environment. In the bottom-up methodology, on the other hand, the rules of agent interactions are typical. In systems designed starting from the bottom, the global state of all the components is assumed to be impossible to obtain, and the desired collective behavior is said to emerge from interactions among individual agents and between the agents and the environment. In summary, in the top-down design the final distributed solution is obtained as a process of relaxation of the constraints that require instant access to remote resources with infinite precision. The bottom-up design starts with a rigorously pre–decided set of rules for the individual behaviors and local interactions and then proceeds with the inference of the global emergent behavior.

While the question of which design is appropriate for a given system extends over the most diverse areas in computer science and computer engineering, in this paper we compare two approaches in a typical domain of multi-agent systems engineering.

2. FOUNDATIONS OF THE DESIGN

2.1 Top Down

The top-down methodology has been recently developed to produce provably performant designs relative to what is achieved in classical centralized control theory. This approach has been also successfully applied to problems of sensor localization, distributed vehicle flow control and distributed surveillance [1, 2]. Ideally the designer should start from the definition of an objective that involves global quantities, then devise a centralized optimization algorithm and finally proceed to the synthesis of the decentralized (agent-based) solution. The design process consists of three steps: modeling, synthesis and analysis/optimization.

Modeling: In this phase the designer identifies and categorizes system's agents according to the following taxonomy derived from classical Control Theory. *Information agents* gather information about the environment and allow its dissemination (analogous to "sensors" in classical systems theory). *Modeling agents* collect data from many information agents and update internal estimates of the "real world" state (analogous to state estimators, like Kalman filters, in classical systems theory). *Planning agents* use the current world state estimates, the viable action or control options and the current goals to plan new actions to carry out. These agents may need to task **brokering** agents (that have no conterpart in classical control theory) to report on available resources such as additional state and action information.

Synthesis: Agent controllers are designed following the lines of a three-stage top-down process: a) At first, it is assumed that each agent can access remote resources local to other agents instantly and with infinite precision (communication delays and bandwidth limitations neglected). So a first centralized solution aimed at optimizing a global objective is designed; b) Next, limitations of the distributed environment are applied and so the visibility of each agent gradually reduced. Consequently inter-agent communication issues arise for now each agent needs to replace global resources with local resources. The result is a fully decentralized solution; c) Finally, the obtained solution must be calibrated via parameter tuning.

In order to solve problems of sensor localization and vehicle flow control, Crespi and Cybenko a) defined a suitable artificial potential function $V(X)$ of global quantities $X = \{x_1, x_2, \ldots, x_n\}$ spread over the system's agents. Then b) they applied the gradient descent method to minimize V: $x_i(t+1) = x_i(t) - \gamma_t \nabla_{x_i} V(X)$, where $x_i(t)$ was intended to be local to agent i and $\nabla_{x_i} V(X)$ was a function of the global quantities X; and finally, c) they replaced $\nabla_{x_i} V(X)$ with a locally computable estimate and proved convergence properties of the tunable solution.

Analysis/Optimization: The inter-agent communication must be optimized in order for the distributed system to perform as predicted at the beginning of the synthesis phase. The **Analysis** conducted in this phase may lead to a review (feedback) of the original Modeling of the agent system thus creating a *cycle*.

2.2 Bottom–Up

The bottom-up design methodology is very popular for producing autonomous, scalable and adaptable systems often requiring minimal (or no) communication. The design process consists of three steps: Synthesis, Modeling and Analysis, and Optimization.

Synthesis: In the Synthesis phase one has to define the agent controller which can be described by an automaton that is the behavioral representation of an agent. In the case of a reactive agent the controller can be characterized by a finite state automaton (FSA). Each state of the automaton represents the action or a behavior the agent is executing, with transitions coupling it to other states. Consequently, the behavioral dynamics of a reactive agent can be considered as an ordinary Markov process.

Modeling and Analysis: Once a controllers for individual agents have been constructed, one need to develop a mathematical model of the collective behavior. Remarkably, the finite automaton of a single agent in many cases can be used for adequately describing the macroscopic or collective behavior of a large-scale system composed of many such controllers. In particular, Lerman et. al. have developed models based on Stochastic Master Equation and its first moment, Rate Equation, to describe the average collective behavior from the details of the agent automaton. The model consists of coupled differential equations describing how the average group behavior changes in time. This modeling approach is based on the theory of stochastic processes. For instance, in apllying this approach to robotic systems, one does not assume knowledge of agent's exact trajectories; instead, we model each agent as a stochastic process and derive a probabilistic model of the aggregate, or average, behavior [3, 4].

Optimization: Mathematical model can be used not only to validate the controller, but also to estimate individual parameters that optimize group-level performance. Using mathematical analysis one can finally answer a number of design questions. Controller synthesis may produce a range of values for internal parameters that result in a valid controller, but these different controllers might result in different group–level efficiency. For example, some parameter values may lead to faster convergence to the desired steady state, while others will lead to smaller deviations from it. Moreover, in a case when agent's controller is represented as a Finite State Automata (FSA), analysis can be used not only for estimating the parameters of the controller, but also suggesting appropriate structure and transition probabilities for the FSA, so that the desired global behavior will be achieved in average.

3. CONCLUSION

In conclusion, we suggest that the main difference between two approaches is the requirements on the availability of local resources to the rest of the system. This requirement is very important for top-down approach, while not so imperative for the bottom-up. More generally, we have identified the following three main elements in our comparison:

Specifications: Bottom–up approach starts with the specification of the individual agent behavior through a set of agent capabilities or rules of engagement which delimit the set of obtainable group–level behaviors. The top-down approach starts with global requirements as in a centralized control system and translates those into necessary agent capabilities. Note that the last step assumes implicitly that the global system requirements can be delegated to individual components. For some tasks this might be not straightforward.

Communication and Noise: Communication is important in both the approaches but its vision is completely different if not opposite. In the top-down case a form of explicit communication is a requirement implied by the necessity for individual components to access remote resources according to the global design. In the bottom-up case, communication is optional in so far as the impact of the propagation of the information throughout the system on the emergent behavior is more like a positive side effect of the design rather than an expected feature required in the specification.

Analysis and performance guarantees: Both approaches are similar in their reliance on simulations to analyze global system properties. Whenever mathematical analysis is possible, however, two approaches differ in their choice of mathematical tools. Top-down approach usually utilizes mathematically rigorous tools such as Classical and Stochastic Control Theory, Optimization Theory, Parameter Estimation, whereas the bottom–up design relies heavily on mean–field methods such as Master equation, mean–field statistical mechanics, dynamical system theory, etc. Hence, in the top-down approach it is possible to establish stringent bounds on the system behavior and make performance guarantees within its range of applicability, while the analysis tools in the bottom-up approach can be extremely efficient in describing the average system behavior.

4. REFERENCES

[1] V. Crespi and G. Cybenko. Decentralized Algorithms for Sensor Registration. In *Proceedinds of the 2003 International Joint Conference on Neural Networks (IJCNN2003), Portland, Oregon*, July 2003.

[2] V. Crespi, G. Cybenko, D. Rus, and M. Santini. Decentralized Control for Coordinated flow of Multiagent Systems. In *Proceedings of the 2002 World Congress on Computational Intelligence. Honolulu, Hawaii*, May 2002.

[3] K. Lerman and A. Galstyan. Mathematical model of foraging in a group of robots: Effect of interference. *Autonomous Robots*, 13(2):127–141, 2002.

[4] K. Lerman, A. Galstyan, A. Martinoli, and A. Ijspeert. A macroscopic analytical model of collaboration in distributed robotic systems. *Artificial Life Journal*, 7(4):375–393, 2001.