

Network Control as a Distributed, Dynamic Game

Sarosh Talukdar and Eduardo Camponogara

ECE Department, Carnegie Mellon University, Pittsburgh, PA 15213

talukdar@cmu.edu, camponog@cs.cmu.edu

Abstract

The operation of large, widely distributed networks can be modeled as distributed dynamic games. This paper assembles a space in which to seek good solutions for such games. Five dimensions of this space—automatic learning, resource shares, additional constraints, altruism and deference—are explained and illustrated.

1. Three Sub-Networks

Large networks for the wide spread distribution of goods and services, such as the electric grid, and the internet, can each be thought to contain three sub-networks:

- DN, a delivery-network to produce and distribute the goods and services.
- AN, an agent-network to control DN. Nodes in AN are agents and arcs are communication links between agents. (By "agent" we mean any decision-maker, no matter how simple or complex, ranging from the most basic relays through optimization software to humans.)
- PN, a problem-network whose nodes are problems, each specifying the goals of an agent, and whose arcs represent couplings between problems. (Two problems are coupled if they share variables.)

The physical networks, DN and AN, are homogeneous (meaning that they are synthesized from only a far fewer types of primitives, such as generators, transformers, breakers and lines, than the number of nodes), and sparse (meaning that the average connectivity of a node is far smaller than the number of nodes). We believe that the problem network also has these properties, specifically:

Conjecture-1: each problem in PN can be formulated as an optimization problem. That is, the goals of each agent, whether they are conventional control goals, such as trajectory following, or commercial goals, such as maximizing the profits from an auction, can conveniently be expressed in terms of objectives and constraints.

Conjecture-2: PN is sparse and its problems are unaffected by growth. That is, each problem in PN is coupled to only a few other problems, and PN grows by adding problems, not by increasing the size of its existing problems.

2. Games and Regulations

How well the agents solve their problems depends, in part, on their organization. The organizational possibilities cover a wide range. At one end, is the completely unregulated game [1], in which the agents are autonomous and work asynchronously (in parallel, each at its own speed), driven by self-interest, and unrestricted by rules or regulations. At the other end, is the totalitarian regime in which the agents have no autonomy, rather, they serve only to execute the instructions issued by a centralized planner. In between these two extremes, is a continuum of increasingly regulated games. The regulations allow the agents to compete along some dimensions while insisting that they cooperate along others, and require the agents to temper their self-interest with altruism. (An agent is altruistic when it modifies its actions to help other agents.)

3. Goals

We seek the sub-set of organizations (game-structures) that elicit high-quality solutions from software agents while allowing these agents to work asynchronously.

The reason for the emphasis on asynchronous work is that makes possible the generation of solutions in a timely fashion. When agents work asynchronously, a fast solution is obtained by assigning a fast agent to its production. Synchronous work, however, imposes precedence constraints on agents; fast agents may be delayed by slower ones, and it is more difficult to produce timely solutions.

Note that in the search for this set of organizations, we cannot rely exclusively on human organizational theory. The right organization for a set of agents depends on their social traits (things such as trustworthiness,

competitiveness, propensity for cooperation, self-interest and altruism). These traits are difficult to change in humans, but are programmable in software agents.

4. Distributed Model Predictive Control

In this and the next section, we will examine the issue of solution-quality.

The problem-network, PN, can be expected to contain a mix of dynamic and static problems. However, each dynamic problem can be replaced by a series of static approximations through the invocation of an appropriate discrete variable method. Model predictive control [2,3] is one such method. It has both strengths and weaknesses. The strengths include systematic procedures for handling constraints, model-based predictions to look ahead (thereby, avoiding short-sighted strategies), and feedback to correct for errors in the predictions. The weaknesses include the possibility of divergence (stability conditions are, as yet, to be found), and a significant expansion in the number of decision variables (typically, by one or two orders of magnitude). Because of this expansion, model predictive control does not scale well, unless it is applied to sparse networks, such as PN [4]. Then, it can be applied separately to each dynamic problem to yield a set of static optimization problems, $\{p_m\}$, of the form:

$$p_m: \text{Minimize } f_m(x_m, Y_m)$$

$$\text{Subject to: } G_m(x_m, Y_m) \leq 0$$

$$H_m(x_m, Y_m) = 0$$

where p_m is a problem to be solved by the m -th agent at the current instant of time in order to determine what this agent should do in the immediate future; x_m is the vector of decision variables for this agent (by "decision variables" we mean the state variables the agent can sense and the control variables it can adjust); Y_m is a vector of the decision variables of the neighboring agents; while G_m and H_m are function vectors representing the constraints that agent- m must meet.

Notice that p_m is coupled to other problems, $p_{j \neq m}$, through the variables, Y_m . These couplings can vary in strength, as illustrated by the two simple examples that follow.

Example 1: shared variables

$$p_1: \text{Min } f_1(x_1, x_2)$$

$$x_1$$

$$\text{s.t. } 0 \leq x_1 \leq 10$$

$$p_2: \text{Min } f_2(x_1, x_2)$$

$$x_2$$

$$\text{s.t. } 0 \leq x_2 \leq 10$$

where x_1, x_2 are scalars,

$$f_1(x_1, x_2) = 44.76x_1^2 - 28.87x_1x_2 + 10.24x_2^2 - 150x_1 - 20x_2,$$

$$\text{and } f_2(x_1, x_2) = 19.49x_1^2 - 34.48x_1x_2 + 25.51x_2^2 - 120x_1.$$

Example 2: shared variables and constraints

$$p_1: \text{Min } f_1(x_1, x_2)$$

$$x_1$$

$$\text{s.t. } x_1 + x_2 \geq 6$$

$$x_1 + x_2 \leq 10$$

$$0 \leq x_1 \leq 10$$

$$p_2: \text{Min } f_2(x_1, x_2)$$

$$x_2$$

$$\text{s.t. } x_1 + x_2 \geq 6$$

$$x_1 + x_2 \leq 10$$

$$0 \leq x_2 \leq 10$$

where f_1 and f_2 are as in Example 1.

Additional examples (for small power-systems and arrays of spring-connected-pendulums) can be found in Camponogara's Dissertation [4].

5. Solution Classes

Each problem in the set $\{p_m\}$ is assigned to a different agent. The solutions they obtain depend on their individual capabilities, on how they interact. Some of the classes of solutions the agents can obtain are listed below.

Reaction Sets

Suppose agent- m is autonomous and competitive. Suppose it knows exactly what every other agent is going to do, in particular, suppose it knows Y_m . Then its best course of action (from the viewpoint of self-interest) is the optimum solution of its problem, p_m . This solution is called its reaction, and the collection of all possible reactions is called its reaction set, $R_m(Y_m)$. In other words: $R_m(Y_m) = \{x_m \mid x_m = \text{Argmin}[f_m(x_m, Y_m)] \text{ for the given } Y_m \text{ such that } G_m(x_m, Y_m) \leq 0 \text{ and } H_m(x_m, Y_m) = 0\}$.

Nash Equilibria

Suppose all the agents are autonomous and competitive. Suppose they are allowed to work on their problems iteratively and in parallel. Suppose each agent is informed about how the plans of the other agents are developing and continually reacts to these developments. From the point of view of agent- m , the value of Y_m is changing (as the other agents readjust their plans) but agent- m always knows the latest values of Y_m and reacts to it. Then, convergence, if it occurs, will be to a fixed point (an attractor) of the system of equations:

$$x_m = R_m(Y_m) \forall m.$$

These fixed points are called Nash equilibria [1] and the set of all such equilibria is called the Nash set, N . Thus, N is a subset of the intersection of the agents' reaction sets.

Pareto Sets

Suppose that the agents work together on the aggregate of their individual problems. Suppose that they do this either by placing themselves under centralized control, or by abandoning competition and self-interest in favor of cooperation and altruism. Then, Pareto optimal solutions [1] can be obtained.

The aggregate of the agents' individual problems is a multi-objective problem [5] of the form:

$$\begin{aligned} \text{AP:} \quad & \text{Minimize } F(X) \\ & X \\ & \text{Subject to } G(X) \leq 0 \\ & \quad H(X) = 0 \end{aligned}$$

where $X = \text{Union}(x_1, x_2, \dots, x_M)$, is the complete set of the system's state and control variables, $F = [f_1, f_2, \dots, f_M]$, is the vector of the agents' conflicting objectives, $H = \text{Union}(H_1, H_2, \dots, H_M)$, $G = \text{Union}(G_1, G_2, \dots, G_M)$, and M is the number of agents.

A solution X^* to AP is said to be Pareto optimal, if X^* is feasible and if X^* is not dominated by any other feasible solution. A solution is feasible if it meets all the constraints. A solution, X^a , dominates another solution, X^b , if $f_m(X^a) \leq f_m(X^b)$ for all m , and if $f_m(X^a) < f_m(X^b)$ for at least one m . The set, P , of all the Pareto optimal solutions is called the Pareto set.

Comments

In being non-dominated, the members of the Pareto set represent the best possible tradeoffs among the conflicting objectives of the agents. The members of the Nash set are not necessarily part of the Pareto set, and as such, represent lesser tradeoffs (see Figs. 1 and 2). But the members of the Pareto set are not equilibria for competitive agents.

By comparing Figs 1 and 2, we see that one of the additional constraints of Example-2 causes the Nash set to move closer to the Pareto set. In other words, regulations expressed as constraints, if carefully designed, can move the Nash equilibria closer to the Pareto set. (Economists have long known that the efficiency of a completely free market—their version of an unregulated game—can invariably be improved by the addition of regulations. We suggest that in designing these regulations it is helpful to represent them as constraints in the agents' problems.)

6. A Design Space For Organizations Of Software Agents

Recall that the goals of this paper are to find organizations that elicit high-quality solutions from agents that are allowed to work asynchronously. The material from the preceding section makes it possible to express these goals more precisely as follows.

Find organizations that:

- make N (the Nash set) coincide with P (the Pareto set) or at least, make N draw close to P .
- ensure the convergence of asynchronous iterations of the agents to members of N , that is, to fixed points of the system of equations:

$$x_m = R_m(Y_m) \quad \forall m$$

where R_m , the reaction set of agent- m , is the set of solutions to p_m , agent- m 's problem.

[4] shows that the requirements for asynchronous convergence are two-fold. First, each agent must know approximately what the other agents are going to do (that is, agent- m must be able to predict Y_m), and second, there must be a fair sharing of resources among fast and slow agents (that is, the agents which make decisions quickly, should not be allowed to use up all the slack provided by the inequality constraints that they share with slower agents). Two ways to meet these requirements are:

- automatic learning: techniques, such as neural nets and Bayesian nets, by which agent- m can translate experience into predictions for Y_m ,
- resource shares: the allocation of fair shares, S_m , by which the constraints, $G_m(x_m, Y_m) \leq 0$ can be replaced by tighter constraints, $G_m(x_m, Y_m) \leq -S_m$, to prevent fast agents from using more of the shared resources than they deserve.

The goal of moving N closer to P can be achieved by:

- additional constraints: designing regulations, represented as constraints, that force N closer to P , as illustrated by Figs. 1 and 2.
- altruism: the adoption by agent- m , of objectives and constraints from the problems of other agents. (The ultimate in altruism is for agent- m to take on the problems of all other agents, that is the aggregate problem, AP.)
- deference: agent- m cedes some control of its decision variables to other agents.

The five dimensions—automatic learning, resource shares, additional constraints, altruism and deference—define a design space (a set of alternatives) for asynchronous organizations that produce high quality solutions. We have not, as yet, explored this space. But samples from it have provided promising results, for instance, those of Fig. 3.

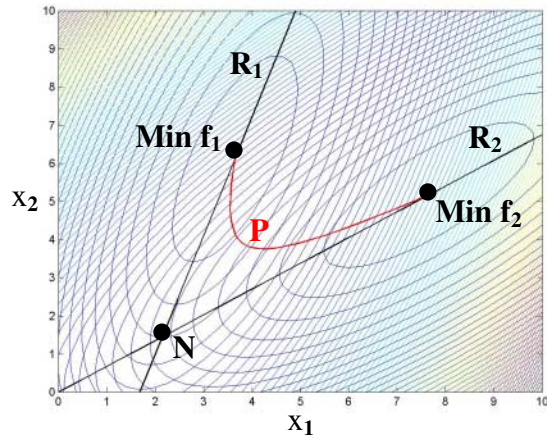


Figure 1. The objective landscape for Example-1. (The ellipses are level sets of f_1 and f_2).

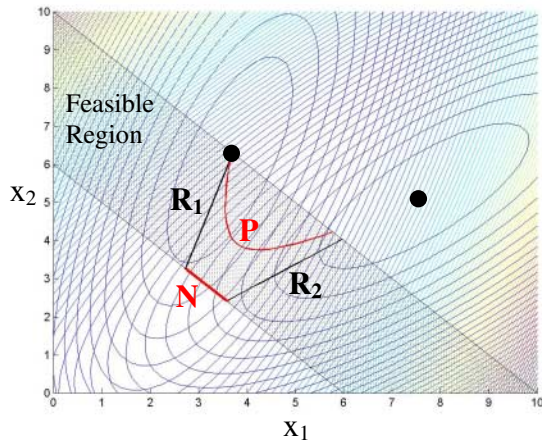


Figure 2. The objective landscape for Example-2.

7. Conclusions

The main contribution of the top-down approach to organization design taken in the paper is the development of a research agenda:

- investigate the validity of the conjectures that problem-networks are homogeneous and sparse (section-1).
- explore the organization-design-space whose dimensions are: automatic learning, resource shares, additional constraints, altruism, and deference.

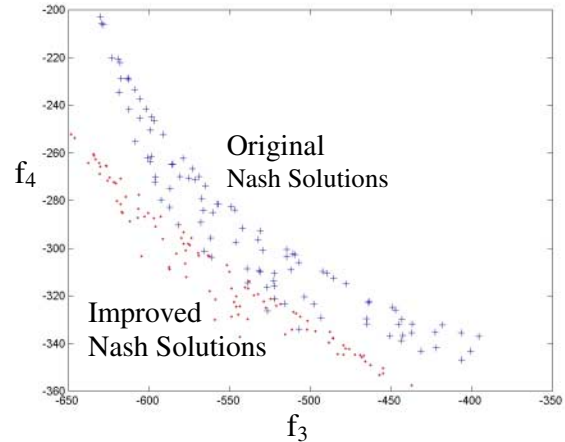


Figure 3. A two-dimensional section showing the improvements in solutions obtained by a four-agent organization using some “resource sharing,” “altruism” and “deference,” but no “learning” or “additional constraints.”

Acknowledgements

This research was supported in part by the Pennsylvania Infrastructure Technology Alliance, by EPRI/ARO under contract number EPRI-W08333-05, and by the Brazilian National Council for Research and Technology Development (CNPq) under grant number 20.0047/96-5.

References

- [1] T. Basar and G.J. Olsder, *Dynamic Noncooperative Game Theory*, Academic Press, 1982.
- [2] E. Mosca, *Optimal, Predictive, and Adaptive Control*, Prentice-Hall, 1995.
- [3] T.E. Morton and D.W. Pentico, *Heuristic Scheduling Systems*, John Wiley and Sons, New York, 1993, pp. 221-223.
- [4] E. Camponogara, “Controlling Networks with Collaborative Nets,” Ph.D. Dissertation, ECE Department, Carnegie Mellon University, Aug. 2000.
- [5] K.M. Miettinen, *Nonlinear Multiobjective Optimization*, Kluwer Academic Publishers, 1999.