Towards Cooperative Negotiation for Decentralized Resource Allocation in Autonomic Computing Systems

Craig Boutilier
Department of Computer Science
University of Toronto
Toronto, ON, M5S 3H5, Canada
cebly@cs.toronto.edu

Rajarshi Das Jeffrey O. Kephart William E. Walsh
IBM T.J. Watson Research Center
19 Skyline Dr.
Hawthorne, NY 10532, USA
{rajarshi, kephart, wwalshl}@us.ibm.com

Abstract

Resource allocation is a key problem in autonomic computing. In this paper we use a data center scenario to motivate the need for decentralization and cooperative negotiation, and describe a promising approach that employs preference elicitation.

1 Resource Allocation in an Autonomic Computing System

An autonomic computing system is designed to drastically reduce the role of human administrators by automating most of the managerial decision making [Kephart and Chess, 2003]. Automated resource allocation is necessary for an autonomic computing system to optimize its performance. In the large, distributed autonomic computing systems we would expect in big businesses, resource allocation will occur at multiple scales. Local allocation decisions will be made within individual elements (servers, databases, storage units, etc.) and small clusters of elements. Local clusters will contend for pools of resources in the larger domain, or across administrative domains. Although we can generally assume that elements in an autonomic computing system of a single corporation will be cooperative, sharing the common goal of optimizing total business value, the complexity of local information will often preclude centralized allocation across the entire system. Cooperative negotiation, using preference elicitation techniques, can be an effective approach to decentralization.

To motivate the problem, consider the problem of resource allocation within a *data center.* The data center provides information technology resources to multiple organizations, each in a separate domain governed by a *workload manager* (WLM). Each WLM decides how to allocate resources in its domain to maintain quality of service (QoS) for each of a set of n classes of transactions.

The QoS specification for a transaction class c includes terms describing monetary payments or penalties as a function of the measured attributes of the service provided to that class by the data center. For simplicity, we shall consider here just a single attribute, the response time t_c for class c. We denote the revenue function for class c (including all rewards and penalties) by $r_c(t_c)$.

In a real system, a WLM can adjust various parameters / for multiple resources. Here, we assume a single resource, with quantity L. Given fixed L, $\vec{f}$, and class demand vector $\vec{d}$, and assuming that the resources have no incremental costs, the expected total revenue is then:

$$R(\vec{f}, L, \vec{d}) = \sum_c \int_{t_c} r_t(t_c) \Pr(t_c \mid \vec{f}, L, \vec{d}) \quad (1)$$

A WLVTs internal resource optimization problem is to find a feasible $\vec{f}$ that maximizes Eq. (1).

To handle fluctuations in client demand, a single provisioncr at the data center periodically reallocates resources among the WLMs. Denoting an individual WLM by i, the resource allocation problem for the provisioner is to compute:

$$\arg\max_{\vec{L}} \sum_i U_i(L_i) \text{ where}$$

$$U_i(L_i) = \max_{\vec{f}} R_i(\vec{f}, L_i, \vec{d}) \quad (3)$$

The provisioner can compute (2) centrally if it has a good model of the internal operation of each WLM and can obtain all relevant state information. In a real system however, the model and data necessary to compute Pr(tc) may be complex and large. Moreover, in a system with transient, heterogeneous components (e.g., differently configured WLMs or different components altogether), the internal models of the components may simply not be available to the provisioner. In these cases, it is necessary to decentralize the resource allocation problem. That is, WLMs would perform local computations and send summary information to the provisioner.

A natural division of labor is to have each WLM send its entire U_i curve, so that the provisioner can compute Eq. (2). Figure 1 shows an example of curves from two WLMs, each with two transaction classes.[1] The provisioner wishes to find the peaks of the aggregate curve, also shown.

2 Cooperative Negotiation

To compute its full U_t curve, a WLM must solve Eq. (3) for each feasible *L1.* If each r_c is in a computationally manageable form (e.g., piecewise linear or quadratic) and we have a

[1] We computed the utility curves assuming a simple M/M/l queue model for each t_c. I is the total service rate available, and the f_c component of $\vec{f}$ indicates the fraction of L given to class c.

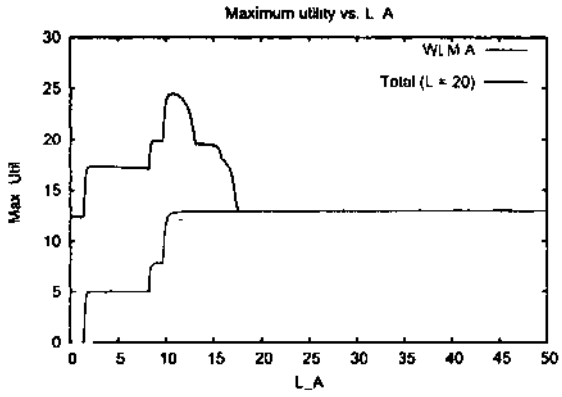

Figure 1: Maximum system utility as a function of allocation, with total resources L — 20. Curve "WLM A" indicates maximum utility to A as a function of L_A. Curve "WLM B" indicates maximum utility to B as a function of L_B. Curve "Total" indicates total utility as a function of L_A provided to A (with L-L_A provided to B).

simple model for t_c (e.g., M/M/I queue), then computing each $U_i(L_j)$ point may be tractable. However, in typical real systems the dependency of the service attributes upon resources and demand is likely to be sufficiently complex as to require a combination of optimization and simulation to compute each $U_i(L_i)$ point. Moreover, WLMs will often have substitutable and complementary preferences over different quantities of *multiple* goods, giving rise to large, expensive-to-compute, multidimensional U_i curves. Such complexities would make it in feasible for a WLM to send its entire U_i to the provisioned

Instead, we propose to model the resource allocation problem as cooperative negotiation. In the context of autonomic computing, cooperative negotiation is not simply non-cooperative negotiation with the simplifying assumption that agents are non-strategic. Rather, the objective is to achieve the right balance between global optimization and negotiation time. We are developing preference elicitation techniques [Boutilier, 2002] to address this problem. We can view the negotiation for resources between the WLMs and the provisioner as involving computation or "elicitation" of relevant parts of the U_i curves.

Partial elicitation may be feasible in the case of negotiation among WLMs for different resource levels. It will often be possible to identify the region of Z-space in which the optimal allocation lies without complete knowledge of these utility functions. The provisioner could be given a small number of samples of the utility functions $U_i(L1)$ for WLMs /. Making simple monotonicity assumptions, the provisioner could determine the region of allocation space in which the optimal allocation lies.[2] For instance, having samples of the two (lower) U_i curves in Figure 1 at points L_x — {10,15,20} for / = A,B, is sufficient to determine that the optimal allocation lies somewhere in the region $L_A \in [10,15]$.

We are currently developing incremental utility elicitation procedures in which the provisioner gradually narrows down the region in which the optimal allocation lies until a decision which is guaranteed to be e-optimal is found. A rough sketch of one such procedure is presented here. Assume that each WLM has provided evaluations of its utility function U_i at a set of m sample points $\tau_i^1 < \tau_i^2 < \ldots \tau_i^m$; we assume that τ_i^m is the maximum value of L_i and bounds U_i under any allocation. With this small set of samples, the provisioner is assured that $U_i(\tau_i^k) \le U_i(L) \le U_i(\tau_i^{k+1})$ for any $\tau_i^k \le L \le \tau_i^{k+1}$. Armed with this information, it is reasonably straightforward to determine the *maximum regret* of any allocation L. Furthermore, we can also determine the allocation that has *minimax regret* given this incomplete knowledge of the utility function:

$$\vec{L}^* = \arg\min_{\vec{L}} \max_{\vec{u}} \max_{\vec{L}'} \sum_i [u_i(L_i') - u_i(L_i)]$$

Here u_t ranges over the set of utility functions U_i consistent with the bounds above. Minimax regret is a reasonable error criterion, and the minimax-optimal allocation L^* can be determined using a tractable scries of linear programs.

Incremental elicitation arises when the minimax-optimal allocation has a max-regret level that is too high, say, greater than E. In this case, the provisioner does not have enough information about the U_t curves to determine an allocation whose worst-case error is less than ε; this necessitates additional samples of the U_i curves. We have developed intelligent query strategies that efficiently determine points L_i for which knowledge of 17,(L,) will "quickly" reduce minimax regret. Once sufficiently many points have been evaluated, minimax regret will reach an acceptable level and an allocation whose error is bounded by e can be offered. Wc do recognize that the procedure becomes more complicated with multidimensional U_i curves.

This procedure obviates the need for WLMs to compute their entire utility curves. The complex optimization required to determine $U_i(L)$ need only be applied by WLM / at a small collection of points; and these points are determined by the provisioner based on information about the utility functions of *other* WLMs. In this sense, such incremental elicitation techniques can truly be viewed as a form of collective negotiation between WLMs and the provisioner.

We are exploring error criteria other than minimax regret in our general model. Optimization w.r.t. specific utility constraints, and determining queries that reduce error, would both require tailoring to the specific criterion adopted; but the same incremental framework still applies. For instance, if distributional information over the U_i curves is available, Bayesian decision criteria could be used (e.g., an allocation that maximizes expected utility w.r.t. Pr(w)).

[2] Monotonicity of U_i seems natural, corresponding to a "free disposal" assumption.

References

[Boutilier, 2002] Craig Boutilier. A POMDP formulation of preference elicitation problems. In *Eighteenth National Conference on Artificial Intelligence,* pages 239-246, 2002.

[Kephart and Chess, 2003] Jeffrey O. Kephart and David M. Chess. The vision of autonomic computing. *Computer,* 36(I):41-52,2003.