Market-based Coordination Strategies for Physical Multi-Agent Systems *

MyungJoo Ham Gul Agha
Open Systems Laboratory, University of Illinois at Urbana-Champaign
{ham1,agha}@cs.uiuc.edu

Abstract

Market-based mechanisms for real-time dynamic coordinated task assignment of physical multi-agent systems are studied. Specifically, a number of auction mechanisms and bidding strategies, as well as swapping, are considered. The physical robot experiments require addressing real-time and fault-tolerance issues in a dynamic environment. We provide a small scale validation for work with software simulations.

1. Introduction

Robot systems have been proposed for use in *search and rescue* (SR) applications. A simplified version of the SR problem, namely the vehicle routing problem [2], is NP-hard; thus, an optimal solution is computationally intractable. For the SR problem, two types of *fully-distributed* and *globally asynchronous* mechanisms are studied: *auctions* and *swapping*. These mechanisms have reasonable computational cost: $O(|agents|+|targets|)$ time for each assignment. We assume that the environment is dynamic; thus, the system needs to be real-time and agents need to act before the information and decisions become obsolete.

In [3], we described experiments studying different parameters and strategies through large-scale software simulations. By necessity, physical experiments involve fewer agents and cannot explore as many possibilities. However, physical experiments are necessary to validate the methods in a more realistic real-time environment. Several results reported in [3] are tentatively supported by our experiments with physical agents.

2. Methods

We assume that robots (agents) and targets move on a bounded rectangular plane and that robots have global knowledge. Servicing a mobile target t requires $req_t>1$ robots, has utility $util_t>0$, and results in $util_t/req_t$ pay-off per robot. When the dedicated req_t robots approach t, the target t is served. A mission is finished if all targets are served. Because robots experience real-time delays in communication, computation, as well as mechanical delays, and targets move, the system has (soft) real-time requirements.

Auctions are used to allocate targets to robots. After targets are allocated, robots pursue the targets. We briefly describe the mechanisms (see [3] for details).

Non-cooperative heuristic (N/C): Robots choose targets with the smallest cost. N/C is not a market-based method, but resembles swarm-intelligence [4].

Forward auction: Robots compete for targets raising prices.

Reverse auction: Targets compete for robots reducing prices.

Forward/reverse auction (F/R): Both forward and reverse auction are used to reduce converging costs.

F/R and sealed-bid auction (S/B): Target price is hidden from robots running F/R auctions. Because there are no fund transactions, target price may be not important.

Utility functions calculate the value of a specific target for the auction methods.

Static: The pay-off $util_t/req_t$ is used.

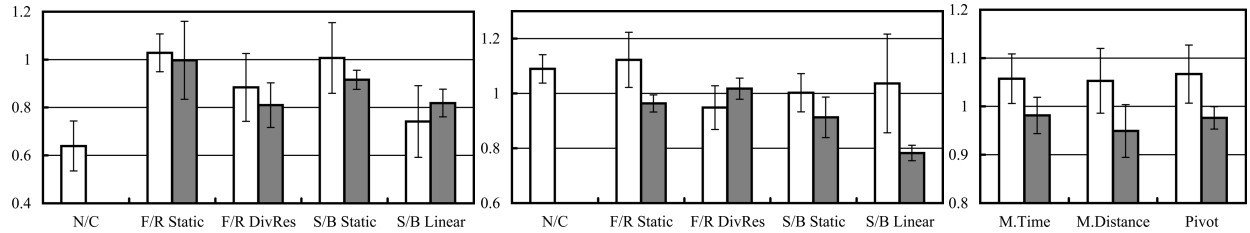
Division: $util_t/(req_t-min(asn_t,req_t-1))$, where asn_t is the number of t's bidders. The value is increased when t has more bidders so that the auction may end soon.

Division-Restricted: The utility increases only if t has insufficient bidders or the robot has already bidden for t.

Linear: The utility increase per bidder is constant.

Because of the dynamic and asynchronous environment, dynamic reallocation may improve the performance. Robots may *swap* tasks after their auctions.

*It was supported in part by NSF grant CNS 05-09321, CMS 06-00433 and by ONR/DoD MURI award N0014-02-1-0715. We thank Liping Chen and Rajesh Karmani at UIUC and Tom Brown at Google for their help.



(a) Mission Time (b) Movement Distance (c) Swapping

Figure 1. Experimental results. Confidence intervals of 90%. White: Simple / Solid: Complex

3. Experiments

6 robots and 9 targets are used. A robot agent runs on an iPAQ h4100 mounted on an Acroname Garcia robot. Targets are virtual and run on a PC. 4 cameras, 4 vision servers, and a master vision server process localization information. A simulation management agent controls missions and handles environments. The system is based on [1]. Two setups are experimented with: req_t=2 in *Simple* and req_t=[2,5] in *Complex*. The setups have different initial positions as well.

Fig. 1 shows part of the experimental results. The controls, whose values are 1, are forward auction with static utility for (a),(b) and no-swapping for (c). In *Simple*, where the problem is trivial, N/C works efficiently; it has the shortest mission time because auctions have communication overhead. N/C suffers from deadlock in *Complex*. However, F/R with division-restricted auction has the shortest distance traveled in *Simple*. In Simple, little or negative improvement is shown by F/R, S/B over forward and by swapping over no-swapping. In contrast, the performance improves more in *Complex*: S/B performs better than F/R, and F/R performs better than forward. Unlike [3], negative effects of dynamic (non-static) utility to the auction quality (movement distance) are not clear. Mission time is reduced significantly enough to show that dynamic utility reduces auction delay; the worse auction quality of dynamic utility can be compensated by the faster auctions. Forward auctions with dynamic utility suffer from deadlock and reverse auctions perform no better than forward auctions as they did in [3]. Ping-pong swapping–robots swap repeatedly without progress–happens more frequently than in software simulations due to larger delays.

Unlike [3], robots need to endure longer and varying delay of sensing (approx. 300–600ms) and communication (20–200ms). Moreover, if a message requires a reply, the reply may take longer due to scheduling delays (sometimes 1 or 2 seconds!). Such delays can be critical; e.g., 1 stands for 267.5sec in Complex of Fig. 1(a).

While auctions are executing, bidding robots do not move. Because the communication delay can be long, auction timeout is long (10 seconds), which also makes auction pause long. Swapping requires multiple steps of negotiations, which incur many messages and replies. Fault-tolerance is important for multi-robot environments; the more robots we have, the more failures may occur. Although we ignore cases with failures, the system works with failures (dead batteries, out-of-bound or frozen robots, and heavy collisions) and other run-time errors (communication or vision errors). When there are failing robots, remaining robots continue to work and try to finish the mission.

4. Conclusions

Physical robot agent experiments support many of the results in previous software simulation results in a soft real-time system. The mechanisms synchronize asynchronous robots for each task. However, the system remains globally asynchronous and the local synchrony appears for a team of robots sharing a task. Although the system does not have firm deadlines, scheduling (task-agents assignments) should be done before the given information becomes obsolete. Besides, a robot team should service a task, before the assignment becomes inefficient (obsolete), or reallocate it (swap) to keep the assignment up-to-date.

Although the mechanisms work in the given implicit and soft real-time constraints, the effect of failures has not been measured under different scenarios, such as slower communication or robot movement, and time constraints of tasks. An analysis on how the performance degradation and system failures with different scenarios remains to be done.

References

[1] A. Ahmed et al. Task assignment for a physical agent team via a dynamic forward/reverse auction mechanism. In *Proc' of KIMAS*, pages 311–317, 2005.

[2] G. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science*, 6(1):80–91, oct 1959.

[3] M. Ham and G. Agha. Market-based coordination strategies for large-scale multi-agent systems. *System and Information Sciences Notes*, 2(1):126–131, 2007.

[4] M. J. B. Krieger, J.-B. Billeter, and L. Keller. Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406:992–995, 2000.