A Fast and Frugal Method for Team-task Allocation in a Multi-robot Transportation System

Jens Wawerla[†] and Richard T. Vaughan[‡]

***Abstract*— In this paper we present two task-allocation strategies for a multi-robot transportation system. The first strategy is based on a centralized planner that uses domain knowledge to solve the assignment problem in linear time. In contrast in the second strategy, individual robots make rule based allocation decision using only locally obtainable information and single value communication. Both methods are tested and analysed in simulation experiments. We show that the rule based method performs well but the lack of information has to be paid for with energy.**

I. INTRODUCTION

Transportation of goods is an important part of today's global economy. The port of Vancouver alone handles 76.5 million metric tons annually [13]. This is a natural task for autonomous mobile robots, and in the future robots will perform more and more transportation tasks, from moving goods around in warehouses [3], delivering mail, bringing semi-finished products from one processing facility to the next, supplying factories to world-wide shipping. To achive this we need systems that assign transportation tasks to robots. Several versions of the problem can be stated, e.g. fixed or changing pickup or delivery points, changing rates at which goods have to be transported and varing fees.

In this paper we consider a multi-robot transportation system with fixed and known source and sink locations, and constant fees. Source produces goods at a certain, variable rate, but cannot store more than one unit at a time. Fig 1 shows an example of this case. The objective of the robots is to transport goods, considered abstractly as pucks, from a source to the corresponding sink. Pucks may not be transported to any other sink. To be economically viable the task-allocation has to be cost effective. We use energy expended by the robots as our unit of cost and pucks transported as our work metric. To reduce the cost, the task-allocation system can unallocate a robot temporarily by sending it to a depot, marked *D* in Fig.1. There the robot is on stand-by and does not require any energy.

We propose two task-allocations methods, (1) based on a central, omniscient planner and (2) governed by a heuristic, using only locally sensed information and minimal and infrequent communication between the robots. The challenge for both methods as well as any other task-allocation algorithms lies in correctly dealing with interference, inherent to all multi-robot systems. We show that a naive solution creates an unintended feedback loop due to interference, that leads to a drop in performance. We suggest a simple yet effective solution to this problem.

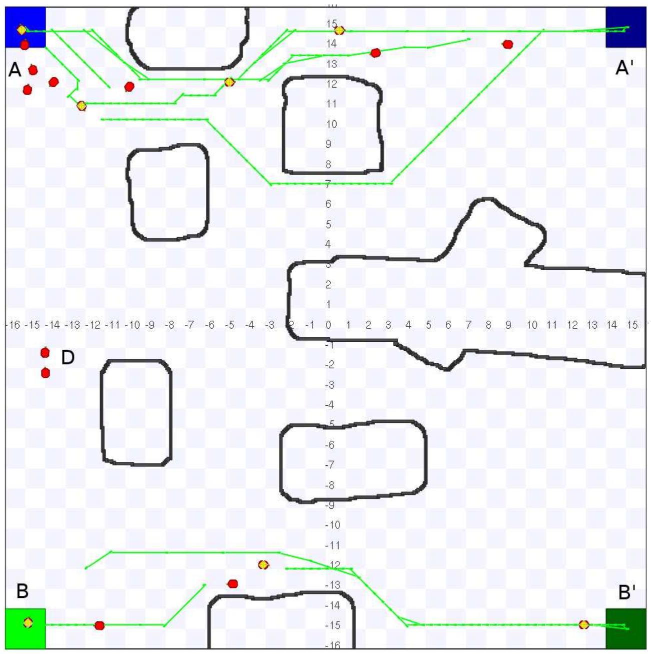


Fig. 1. Screenshot of a Stage simulation with two tasks, transporting pucks from A to A' and from B to B'. D marks the robot depot for unassigned robots. The thin (green) lines show the path planed by each robot. Robots carrying a puck are displayed with a (yellow) diamond on top.

A. Related Work

Robot task allocation is a widely studied field. It can be broadly classified into two classes: planner based systems and systems that rely on individual robots making individual task allocation decisions. Planners are often based on auction mechanisms in which robots bid for tasks. Gerkey's MURDOCH system [2] is one such example. As we will show, the problem we study does not require the sophistication of this class of planners.

Local decision mechanisms are attractive because they are redundant and by definition do not rely on a centralized agency. Labella et al. [6] use a minimalistic rule originating from modelling ant behaviour. Each robot's probability to leave the nest is increased by a small δ if it found food on the last foraging trip and decreased by the same δ if the robot returned empty handed. Liu et al. [8] present a similar central place foraging system but with the explicit goal to minimize energy expenditure. The decision each robot makes

Simon Fraser University, School of Computing Science
Burnaby, BC, V5K 1S6, Canada
†jwawerla@sfu.ca
‡vaughan@sfu.ca
This work was supported by NSERC and DRDC in Canada.

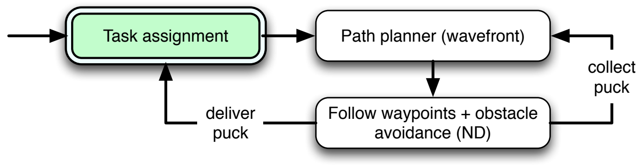


Fig. 2. Overview of the robot system

is based on two thresholds. One for the amount of time spent resting and one for the time spent foraging. The thresholds are adapted by certain cues. Colliding with other robots makes a robot more likely to rest and successfully retrieving food makes the robot more likely to keep foraging. In an experiment with up to 12 Khepera robots Krieger et al. [5] use a similar threshold based task-allocation mechanism to keep the swarm's energy reserve at a safe level. [5] also provides an extensive list of references to studies about task allocation in social insects. All three methods have been shown to arbitrate between work and rest periods but it is not clear how to select between two work tasks and resting.

Jones [4] proposes a probabilistic rule that is shown to work for at least two tasks. The robots calculate the probability of switching tasks based on a local estimate of the number of robot performing the same task and on an estimate of the available work load. This requires the robots to communicate what task they are currently engaged in. Jones explicitly prohibits an idle task. McFarlands's Cue$\times$Deficit-rule [9] allows robots to select between two types of resource depending on the robot's deficit and encounter rate of a particular resource. McFarland models situations in which the robot is in need of both resources, e.g. needing water and food. In the transportation problem considered in this paper a deficit of a certain task is not meaning full. McFarlands's method also requires encountering both resources at least every once in awhile in order to update the cue estimates. This cannot be guaranteed in a transportation problem.

Some work uses machine learning techniques to estimate the utility of tasks. A recent example is Dahl's vacancy chain scheduling [1]. Reinforcement learning and ε-greedy action selection is used to have robots choose between two transportation tasks. The scenario is similar to ours but less challenging due to the absence of fixed obstacles in the world. Also energy is not considered in the performance metric. The method we propose is significately less complex then that proposed by Dahl and does not suffer from any explicit trade-off between exploration and exploitation as Q-learning does.

II. ROBOT SYSTEM

The robots used in this work are generic models of a differential drive robot simulated in Stage [12]. Each robot is equipped with a 360° field of view laser range finder providing 64 samples with a maximum range of 5 meters. Each robot can localize itself with an abstract localization device. This models a GPS module or any form of SLAM implementation. To transport pucks each robot is capable of sensing pucks, picking them up and dropping them off. The cargo capacity is limited to one puck per robot. In order to communicate with other robots or with the centralized planner the robots are equipped with a generic communication device. We assume reliable communication, but as we will show communication used by our task allocation policies is minimal.

As energy expended by the robots is one of our performance measures the energy model of the robots is important. Two types of energy expenditure occur, (1) energy used for computation and sensing is expended at a constant rate and (2) energy used for locomotion is expended at a rate proportional to the speed of the robot. The first energy model is fairly accurate. The second model does not take acceleration into account and is thus not fully realistic. Acceleration and deceleration occur frequently in high interference situations, therefore our energy model does not penalize interference as much as one would expect. The actual energy rates are modelled after a typical Pioneer 3DX robot. Robots in the depot are in a stand-by mode and do not use any energy.

The robot control software is split in two parts, (1) the common part that handles navigation, obstacle avoidance, picking up and delivering pucks etc. and (2) the task allocation part that decides which task to perform next. An overview is shown in Fig. 2.

Path-planning is done using a wavefront planner (WP). This widely used technique discretizes the world into a grid and builds a gradient of shortest distances starting at the goal location. The shortest path is then given by gradient decent from the robot's current position. (see [7] for details). In our implementation we augment the map data with sensor data from the past n time steps. This enables the planner to incorporate knowledge about temporary changes in the world such as congestion. Figure 1 illustrates this feature, one robot has planned a seemingly longer path around the south side of the obstacle at (0,10). While four other robots planned the shorter path along the often more congested north side of the same obstacle. Emphasis was put on reliability of the planner and not so much on interference reduction. For example the introduction of simple traffic rules such as "always stay on the right side of corridors" may reduce interference noticeably. But as highways in any major city during rush-hour are evidence, interference cannot be avoided as the number of agents increases. In fact to investigate our task-allocation methods we want to explore the configuration space under low and high interference conditions.

The planned trajectory is used by a sensor-based obstacle avoidance routine to navigate the robot along the trajectory to the goal. We use Minguez's [10] Nearness Diagram (ND) an extension of the Vector Field Histogram approach. Minguez's implementation is available on his homepage[1].

The key contributions of this paper are the task-allocation methods. We investigate and compare two methods, both using the same navigation and control routines described

[1]http://webdiis.unizar.es/~jminguez/code/sources_ND_english.zip

above.

A. Allocation Re-Planner

The allocation re-planner is a centralized planner that has full knowledge of the world, the production rate of the sources and the current robot task assignments. An optimal assignment is achieved if the puck production rate of a task equals the sum of the transport rate of all assigned robots. Only under this condition is there no unused transport capacity nor are pucks waiting at the source. The optimal number of robots is thus given by

$$N_i = \frac{\dot{p}_i T_i}{c} \tag{1}$$

where the index i refers to the i-th task, $\dot{p}_i$ is the production rate of the task, T_i the round trip time the robot needs to drive from the source to the sink and back, and c is the puck carrying capacity of the robot ($c = 1$ in our case). It is fair to assume that a central planner has accurate information about the robot's carrying capacity and the production rate for each task. But knowledge of the round trip time is difficult to obtain. The round trip time can be estimated from the distance between source and sink and the speed of the robot. The distance between source and sink maybe accurately obtained from a path-planner e.g. the wavefront planner. The actual speed of the robot is a function of the robots hardware, the control software, as well as the interference between the robot and other robots or fixed obstacles in the environment. Modelling this interference and the resulting change in speed is difficult. For example, in the environment in Fig. 1 task A is subject to a higher degree of interference then task B because of the narrow corridor between the obstacles at (0,10) and (-8,14). Modelling interference is generally a difficult endeavour. A simple model is proposed by Seth [11]. Given an interference factor Seth's model uses this factor and the number of agents to calculate the change in performance. This approach has two limitations, (1) we do not know the interference factor and (2) since this model calculates the effect of interference on the average performance and not on a sensori-motor level it is too high level for our purpose. Instead of modelling interference we take the embodied approach. A naive solution is to average over the actual round trip time, as experienced by the robots, and base the planners allocation on this time estimate. Unfortunately this method has an undesired feedback loop. Randomly occurring interference will increase the estimate of the round trip time, causing the planner to allocate more robots to the task and thus eventually increasing the interference. This in turn results in even more allocated robots.

What is needed, is a way to distinguish between the "true" round trip time and delays caused by interference due to randomly occuring congestion. As a simple measure for interference we use the speed of the robot. We sum up the time steps during which the robot's speed is below a certain threshold. The robots then report the experienced round trip time corrected for the time wasted due to interference and the time spent waiting at the source and sink $\hat{T}_i$.

Once a robot completes a task the planner may assign it a new task using Eq. 1 and a low-pass filtered estimate of the, interference corrected experienced round trip time. Alternatively the robot might be send to the depot if it is no longer needed. The complexity of this planning step is $O(k)$ where k is the number of tasks. In the depot, robots are waiting in a queue and only the head of the queue queries the planner for a new assignment.

B. Allocation Heuristic

The allocation heuristic has no access to any information other than the location of sources and sinks. Since the production rates are unknown, robots initially select a task at random with equal probabilities. Every time a robot returns from delivering a puck it faces one of three possible situations:

1) robots are already waiting in a queue to pick up a puck
2) no robots are waiting and a puck is available for pick up
3) no robots are waiting and no puck is available yet

Situation (1) can be evidence that the number of robots assigned to the task is too high. It can also be a result of an unfortunate spatial clustering of robots. Situation (2) might indicate that the number of robots assigned to the task is too low since the production rate appears to be higher then the transportation rate. But again this could also be a consequence of a brief congestion somewhere along the transportation route. Situation (3) is, at least from this particular robots point of view, the ideal case. On one hand the absence of a robot waiting queue indicates that there are not too many robots assigned to the task. On the other hand pucks not already waiting indicates that the task has recently been serviced by another robot.

The allocation heuristic is entirely based on these three situation, which are easily identifiable by the robots. A robot in situation (1) randomly chooses a maximal time it is willing to wait in the queue according to

$$t_{wait} = max(t_{minwait}, \hat{T}_i \cdot r) \tag{2}$$

where $t_{minwait}$ is a minimal waiting time, $\hat{T}_i$ the corrected round trip time and r is a uniform random number in the interval $[0,1]$. If the robot is able to enter the loading bay before the waiting time is up, it will continue with the current task. Otherwise it stops performing any task and returns to the depot.

A robot in situation (2) broadcasts a worker recruitment message for the current task with a probability p_b. The first robot in the depot will respond to the allocation call by starting to perform the task in need of an additional worker. In the case that there are no robots in the depot, the allocation message remains unanswered. Situation (3) does not require the robot to alter the allocation, thus it simply continues to perform the task. Because production rates may drop to zero, a robot in situation (3) will only wait for a puck to be produced for a maximal duration. We choose the last experienced round trip time to be the maximal waiting time before returning to the depot.

Note at no point are the robots obtaining any information about the tasks production rates, the number of assigned robots per task or the number of unassigned robots. The group allocation is entirely based on locally observed information and a single value broadcast message, used infrequently.

III. EXPERIMENTS

All experiments reported in this paper were conducted with 18 simulated robots in Stage [12] a sensori-motor level multi robot simulator. The robot controller was the same during all experiments, just the task selection component was swapped. The simulation environment is shown in Fig. 1. Initial starting positions of the robots, the location of source and sink as well as the total sum of production rates were fixed. We just altered the ratio of the production rate between the two tasks. Performance was measured in two dimensions, total number of pucks transported in a fixed amount of time and total energy expenditure during this time. These values are not directly convertible into a single performance criterion. The result of any form of linear combination is dependent on the weighting and is thus application dependent and not suitable for a general comparison. Due to the cost of interference, ratios like pucks transported per unit energy, generally favour allocations where only one robot is performing the task while the remaining robots are in stand-by. So we report the results of our experiments in a energy-work graph, similar to a precision-recall graph in machine learning. Results closer to the upper left corner of a energy-work graph should be considered better. Less energy is then expended and more work performed. All experiments were run for 2 simulated hours.

A. Constant Production Ratios

In a first set of experiments we kept the production rate constant at 1:1, 2:1 and 10:1 respectively. Each configuration was tested 20 times. Fig. 3 shows the results in energy-work graphs for the re-planner and the heuristic allocation for a selection of broadcast probabilities. We tested probabilities with 0.1 increments, but due to space constraints only show the most interesting ones.

In order to not only compare our two task-allocation methods with each other and to allow the read to see where in the possible energy work space our solutions lie we also characterize the possible energy work space. To do so we assigned a fixed number of robots to each task and kept this assignment constant during the course of a 2 hour simulation run. We repeated this process for all possible ways to assign up to 18 robots to two tasks, where the unassigned robots return to the depot, just like in the case of the other two policies (a total of 189 experiments).

As mentioned earlier it is application dependent whether we wish to minimize energy expenditure or maximize puck transportation or find some middle ground. But in general we wish to minimize wasting energy. Therefore control policies that achieve the same number of pucks transported while spending less energy have to be considered better policies. From the fixed assignment we can see the line of good performance, it is the left hand side of the fixed assignment energy-work tuples. From Fig. 3 we conclude that the re-planner performs very well, since it's results are on the line of good performance for all configurations tested.

The results from the allocation heuristic are subject to a higher degree of variance. This is to be expected due to the stochastic nature of the heuristic. Further we observe that a low broadcast probability yields results closer to the line of good performance. The reason is that a higher probability causes a higher frequency of re-assignments and these cost energy as the newly allocated robot travels to the worksite. The key to the heuristic proposed is to trade-off re-assignments and adaptation to the task configuration. From the graphs it is apparent that this adaptation is more difficult in the 2:1 case. The 1:1 case is easier, because of the initial random task selection, the ratios of robots already closely matches the production rates and "only" a reduction of workers has to be achieved by the system. The cues used by the heuristic are more pronounced in the 10:1 case, because the ratios are so much more different.

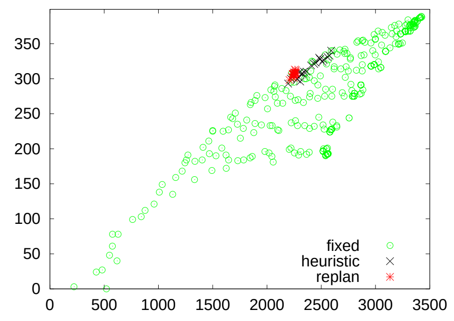
Summarizing, the allocation heuristic performs remarkably well considering that no information about the production rates or the allocation of other robots is known to the heuristic.

B. Changing Production Ratios

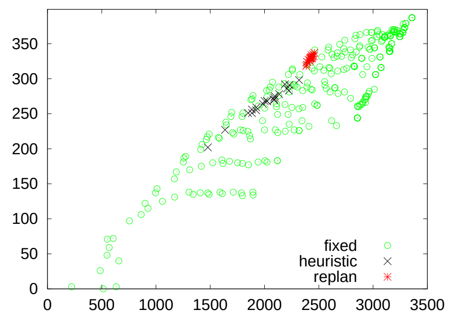
TABLE I
AVERAGE NUMBER OF ROBOTS ASSIGNED BETWEEN TWO TASKS WITH STEPWISE CHANGE OF PRODUCTION RATES FROM 3:1 TO 1:3

Ratio	re-planner	heuristic with broadcast probability			
		0.0	0.3	0.7	1.0
3:1	8.0 : 3.0	7.4 : 3.6	9.1 : 4.1	10.1 : 4.1	10.6 : 4.4
1:3	3.1 : 8.0	3.4 : 3.6	3.7 : 8.1	3.8 : 10.4	3.8 : 11.7

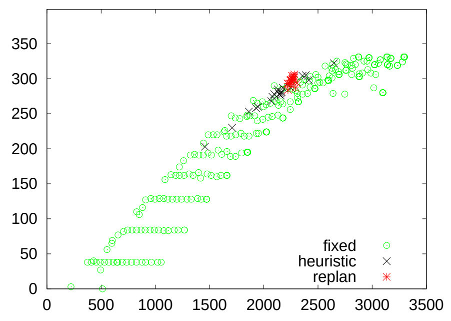
Next we investigate how the two policies perform in situations of changing production rates. Therefore we conducted an experiment in which we set the production ratio to 3:1 for the first hour and then reversed it to 1:3 for the second hour. Unlike in the constant rate situation, a fixed assignment obviously does not show us the possible energy-work space. So we restricted our analysis to comparing the re-planner with the heuristic. The results are summarized in energy-work graphs (Fig. 4) and the average robot assignment numbers in Tab. I. The assignment numbers are the average number of robots assigned to a task during a production rate period. As to be expected, with zero broadcast probability the heuristic just de-allocates robots once the production rate of the first task drops. With probability 0.3 there is enough recruitment so that the ratio of assigned robots matches that of the re-planner. The number of assigned robots is generally higher, because of the overhead of constant re-assignment. As the probability increases we observe an increase in overhead and thus higher assignment numbers but the ratio is similar to that of the re-planner. Disabled recruitment (prob. 0.0) means fewer workers after the ratio change and that results in fewer



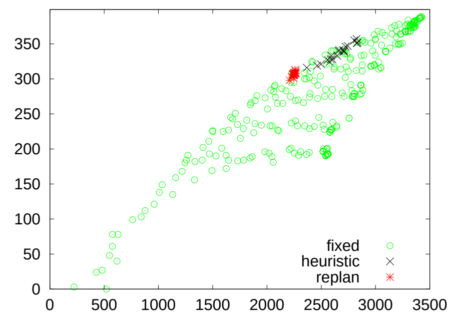
(a) Ratio 1:1, broadcast probability 0.0



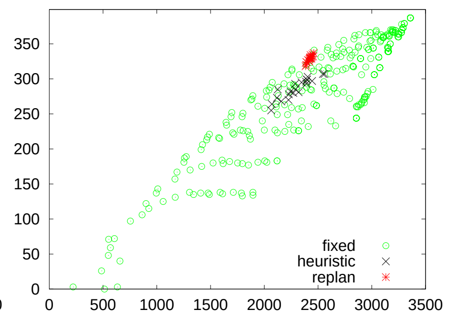
(b) Ratio 2:1, broadcast probability 0.0



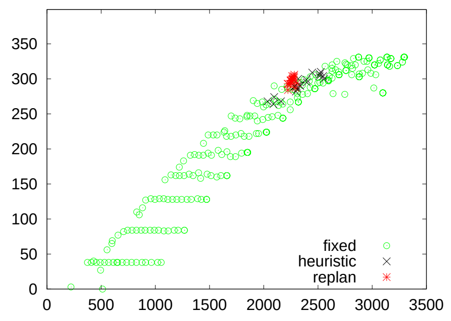
(c) Ratio 10:1, broadcast probability 0.0



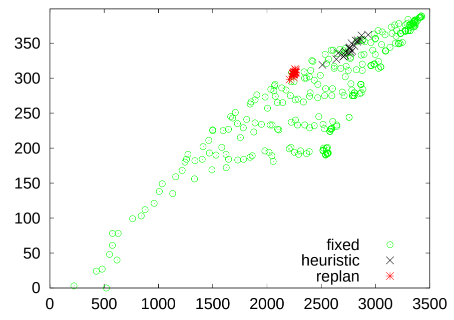
(d) Ratio 1:1, broadcast probability 0.1



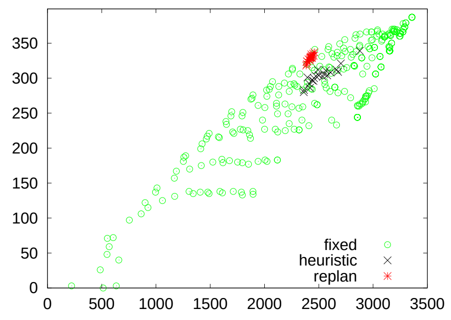
(e) Ratio 2:1, broadcast probability 0.1



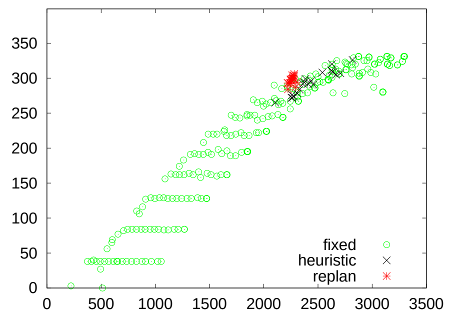
(f) Ratio 10:1, broadcast probability 0.1



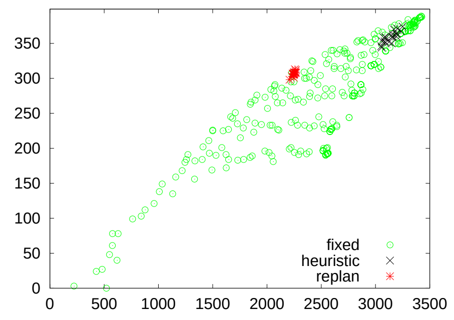
(g) Ratio1:1, broadcast probability 0.2



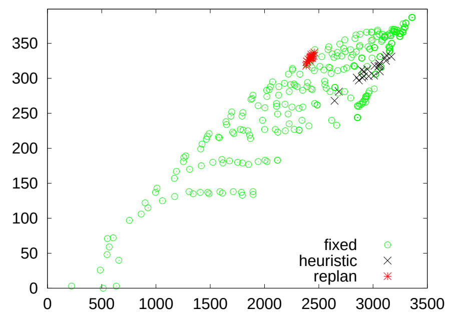
(h) Ratio 2:1, broadcast probability 0.2



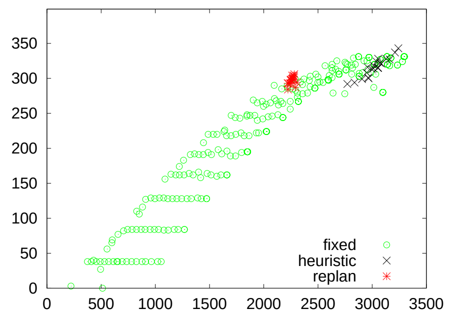
(i) Ratio 10:1, broadcast probability 0.2



(j) Ratio 1:1, broadcast probability 1.0



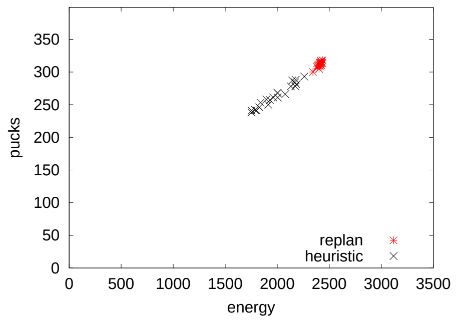
(k) Ratio 2:1, broadcast probability 1.0



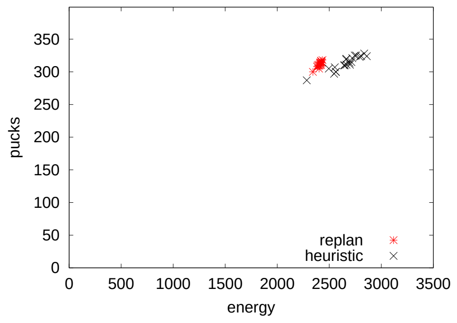
(l) Ratio 10:1, broadcast probability 1.0

Fig. 3. Energy-Work graphs for different production rate ratios and broadcasting probabilities. Energy expended is plotted on the x-axis and pucks transported on the y-axis. The green circles mark results from fixed robot assignments to visualize the possible performance space. The performance of 20 trials of the re-planner is shown in red asterixes and the performance from 20 trials of the allocation heuristic is shown in black crosses.

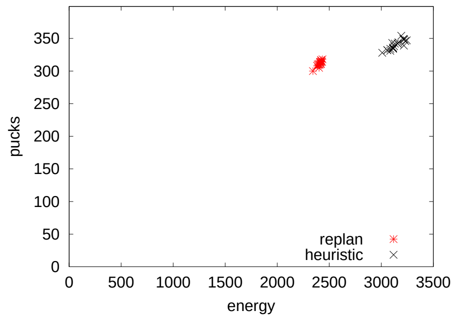
pucks transported (Fig. 4(a)) and less energy spent. Using a broadcasting probability of 1.0 does not yield more pucks transported but results in more energy wasted on frequent re-allocation (Fig. 4(c)), this coincides with our analysis of the assignment numbers. More interesting is Fig. 4(b), which shows that the work rate achieved by both policies is about the same (broadcast probability of 0.3), but the heuristic requires more energy. As in the first experiments, lack of information has to be paid for with energy.



(a) Ratio 3:1 → 1:3, broadcast prob. 0.0



(b) Ratio 3:1 → 1:3, broadcast prob. 0.3



(c) Ratio 3:1 → 1:3, broadcast prob. 1.0

Fig. 4. Stepwise change of production rates from 3:1 to 1:3

IV. DISCUSSION

For energy to be used in a performance analysis of simulation experiments an accurate model of the robots energy consumption is required. Mainly this model must be able to capture possible differences in energy requirements of the sensori-motor activity caused by different control policies. To test our methods under more realistic circumstances we are currently working on real robot experiments. An issue not addressed in this work and left for the future is refuelling. Our current implementation can handle refuelling in principle but more and especially longer experiments are required for a thorough investigation.

The sources of the transportation task used throughout this paper were only able to store one puck at a time. In the future we plan to investigate systems in which pucks accumulate at the source if the robots do not pick them up in time. Another question left for the future is how does task priority, e.g. expressed in terms of task-dependent rewards, influence the allocation policies?

The paper introduced two task-allocation strategies for a multi-robot transportation system. The computational complexity of both strategies is small. In simulation experiments we compared the strategies in form of energy-work graphs. It is not possible to generally say which method performs better because we lack a unifying metric. But we can conclude that the re-planner's performance is on the line of good performance. The performance of the heuristic allocation is often slightly below the line of good performance. Yet the heuristic still performs well, considering that it uses no information about the production rates nor about the task allocation of other robots. This simple rule based allocation policy yields remarkable results entirely relying on local information and minimalistic broadcasting.

EXPERIMENTAL DATA

In accordance with the Autonomy Lab's policy on code publication, the source code and analysis scripts of the experiments are made available online at `git://github.com/rtv/autolab-fasr.git`. The exact data that led to the presented results can be accessed via the commit hash `ed47f9212cf3b72f6a083b48bc2d39a1b79006af`.

REFERENCES

[1] T. S. Dahl, M. J. Matarić, and G. S. Sukhatme. Multi-robot task allocation through vacancy chain scheduling. *Robotics and Autonomous Systems*, 57(6):674–687, 2009.

[2] B. P. Gerkey and M. J. Matarić. Sold!: Auction methods for multi-robot coordination. *IEEE Transactions on Robotics and Automation, Special Issue on Multi-Robot Systems*, 18(5):758–768, 2002.

[3] E. Guizzo. Three engineers, hundreds of robots, one warehouse. *IEEE Spectrum*, July 2008.

[4] C. V. Jones and M. J. Matarić. Adaptive division of labor in large-scale minimalist multi-robot systems. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1969–1974.

[5] M. J. B. Krieger and J.-B. Billeter. The call of duty: Self-organised task allocation in a population of up to twelve mobile robots. *Robotics and Autonomous Systems*, 30:65–84, 2000.

[6] T. H. Labella, M. Dorigo, and J.-L. Deneubourg. Self-organised task allocation in a group of robots. In *Proceedings of the 7th International Symposium on Distributed Autonomous Robotic Systems*, pages 371–380, 2004.

[7] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[8] W. Liu, A. F. T. Winfield, J. Sa, J. Chen, and L. Dou. Towards energy optimization: Emergent task allocation in a swarm of foraging robots. *Adaptive Behavior*, 15(3):289–305, 2007.

[9] D. J. McFarland and E. Spier. Basic cycles, utility and opportunism in self-sufficient robots. *Robotics and Autonomous Systems*, 20:179–190, June 1997.

[10] J. Minguez and L. Montano. Nearness diagram navigation (nd): Collision avoidance in troublesome scenarios. *IEEE Transactions on Robotics and Automation*, 20(1):45–59, 2004.

[11] A. K. Seth. Competitive foraging, decision making, and the ecological rationality of the matching law. In *From Animals to Animats 7, 7th International Conference on Simulation of Adaptive Behavior, SAB*, pages 359–368, 2002.

[12] R. T. Vaughan. Massively multi-robot simulations in Stage. *Swarm Intelligence*, 2(2-4):189–208, 2008.

[13] Wikipedia. Port of Vancouver - Wikipedia, The Free Encyclopedia, 2009. `http://en.wikipedia.org/wiki/Port_of_vancouver` [Online; accessed 15-Sept-2009].