Robotic Message Ferrying for Wireless Networks using Coarse-Grained Backpressure Control

Shangxing Wang
Dept. of Electrical Engineering,
University of Southern California,
Los Angeles, CA
Email: shangxiw@usc.edu

Andrea Gasparri
Department of Engineering
Roma Tre University
Rome, Italy
Email: gasparri@dia.uniroma3.it

Bhaskar Krishnamachari
Dept. of Electrical Engineering,
University of Southern California,
Los Angeles, CA
Email: bkrishna@usc.edu

***Abstract*—We formulate the problem of robots ferrying messages between statically-placed source and sink pairs that they can communicate with wirelessly. We first analyze the capacity region for this problem under both ideal (arbitrarily high velocity, long scheduling periods) and realistic conditions. We indicate how robots could be scheduled optimally to satisfy any arrival rate in the capacity region given prior knowledge about arrival rates. We then consider the setting where the arrival rates are unknown and present a coarse-grained backpressure message ferrying algorithm (CBMF) for it. In CBMF, the robots are matched to sources and sinks once every epoch to maximize a queue-differential-based weight. The matching controls both motion and transmission for each robot: if a robot is matched to a source, it moves towards that source and collects data from it; and if it is matched to a sink, it moves towards that sink and transmits data to it. We show through analysis and simulations the conditions under which CBMF can stabilize the network. We show that the maximum achievable stable throughput with this policy tends to the ideal capacity as the schedule duration and robot velocity increase.**

I. Introduction

Since the work by Tse and Grossglauser [1], it has been known that the use of delay tolerant mobile communications can dramatically increase the capacity of wireless networks by providing ideal constant throughput scaling with network size at the expense of delay. However, though the idea of message ferrying using controllable mobility nodes dates back to the work by Zhao and Ammar [2], nearly all the work to date has focused on message ferrying in intermittently connected mobile networks where the mobility is either unpredictable, or predictable but uncontrollable. With the rapidly growing interest in multi-robot systems, we are entering an era where the position of network elements can be explicitly controlled in order to improve communication performance.

This paper explores the fundamental limits of robotically controlled message ferrying in a wireless network. We consider a setting in which a set of K pairs of static wireless nodes act as sources and sinks that communicate not directly with each other (possibly because they are located far from each other and hence cannot communicate with each other at sufficiently high rates) but through a set of N controllable robots. We assume that there is a centralized control plane (which, because it collects only queue state information about all network entities, can be relatively inexpensively created either using infrastructure such as cellular / WiFi, or through a low-rate multi-hopping mesh overlay).

We mathematically characterize the capacity region of this system, considering both ideal (arbitrarily large) and realistic (finite) settings with respect to robot mobility and scheduling durations. This analysis shows that with $N = 2K$ robots the system could be made to operate at full capacity (effectively at the same throughput as if all sources and sinks were adjacent to each other). We indicate how any traffic that is within the capacity region of this network can be served stably if the data arrival rates are known to the scheduler. We then consider how to schedule the robots when the arrival rates are not known *a priori*. For this case, we propose and evaluate a queue-backpressure based algorithm for message ferrying that is coarse-grained in the sense that robot motion and relaying decisions are made once every fixed-duration epoch. We show that as the epoch duration and velocity of robots both increase, the throughput performance of this algorithm rapidly approaches that of the ideal case.

The primary benefit of the queue-differential based backpressure approach to message ferrying we propose in this work over the original proposal to use message ferrying nodes with statically-determined deterministic routes [2] is that it is much more adaptive to traffic conditions and particularly for large epoch durations, approaches the maximum possible stable throughput.

II. Problem Formulation

There are K pairs of static source and destination nodes located at arbitrary locations. Let the source for the i^{th} flow be denoted as $src(i)$, and the destination or sink for that flow be denoted as $sink(i)$. Source i receives packets at a constant rate denoted by λ_i.

There are $N \leq 2K$ mobile robotic nodes that act as message ferries, i.e. when they talk to a source node, they can collect packets from it, and when they talk to a sink node, they can transmit packets to it. Furthermore, for simplicity, we assume that the static nodes do not communicate directly with each other, but rather only through the mobile robots.

Time is divided into discrete time steps of unit duration. The locations of the sources and sinks for flow i are denoted by $x_{src(i)}$ and $x_{sink(i)}$ respectively, and the location of robot j at time t is denoted as $x_j(t)$. Let the distance between a source for flow i and a robot j be denoted as $d(x_{src(i)}, x_j(t))$ (similarly for the sink). When in motion, the robotic nodes move with a uniform velocity v directly to the destination (there are no obstacles), so that if robot j is moving towards

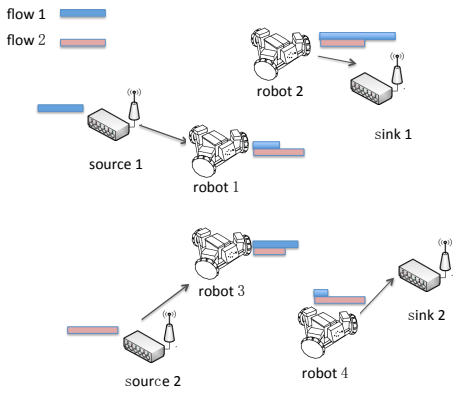

Fig. 1. A network containing 2 pairs of source and sink nodes and 4 robots

the source for flow i, its position $x_j(t)$ is updated so that it moves along the vector between its previous position and the source location to be at the following distance:

$$d(x_{src(i)}, x_j(t+1)) = max\{d(x_{src(i)}, x_j(t)) - v, 0\} \quad (1)$$

We assume that the rate at which a source for flow i can transmit to a robot j, denoted by $R_{src(i),j}(t)$ is always strictly positive, and decreases monotonically with the distance between them, and similarly for the rate at which a robot j can transmit to the sink for flow i, denoted by $R_{j,sink(i)}(t)$. We assume that when the robot is at a location of a particular source or sink, (i.e., the distance between them is 0), the corresponding throughput between the mobile robot and that source or sink is R_{max}

The queue at the source for flow i is denoted as $Q_{src(i)}$. It is assumed that there is no queue at the sinks as they directly consume all packets intended for them. Each robot j maintains a separate queue for each flow i, labelled Q_j^i. Figure 1 shows an illustration of this system with $K = 2$ flows and $N = 4$ robots.

Every T time steps there is a new epoch. At the start of each epoch, it is assumed that the information about queue states of all source and sink nodes as well as all queues at each of the robots is made available to a centralized scheduler. At that time this centralized scheduler can use this information to match each robot to either a source or sink. The matching is represented by an allocation matrix A such that $A(i,j)$ is 0 if the robot j is not allocated to either source or sink for flow i, 1 if it is allocated to $src(i)$, and -1 if it is allocated to $sink(i)$. When a robot is allocated to a given source (or sink), for the rest of that epoch it moves closer to that node until it reaches its position. At at all time steps of that epoch that robot will communicate exclusively with that source (or sink) to pick up (or drop, in case of the sink) any available packets betweent the corresponding queues at a rate dependending on its current distance to that node.

If a robot j is communicating with $src(i)$ at time t, the update equations for the corresponding queue of the robot and the source queue will be as follows:

$$\begin{aligned} n_p(t) &= \min\{R_{src(i),j}(t), Q_{src(i)}(t)\} \\ Q_j^i(t+1) &= Q_j^i(t+1) + n_p(t) \\ Q_{src(i)}(t+1) &= Q_{src(i)}(t+1) - n_p(t) + \lambda_i \end{aligned} \quad . \quad (2)$$

Similarly, if the robot j is communicating with $sink(i)$ at time t, the queue update equation for the robot's corresponding queue will be:

$$\begin{aligned} n_p(t) &= \min\{R_{j,sink(i)}(t), Q_j^i(t)\} \\ Q_j^i(t+1) &= Q_j^i(t+1) - n_p(t) \end{aligned} \quad . \quad (3)$$

III. Capacity Analysis

We define an open region $\mathbf{\Lambda}$ of arrival rates as follows:

$$\mathbf{\Lambda} = \left\{ \lambda | 0 \le \lambda_i < R_{\max}, \quad \forall\ i,\ \sum_{i=1}^{K} \lambda_i < \frac{R_{\max} N}{2} \right\}. \quad (4)$$

We shall show that this arrival rate region $\mathbf{\Lambda}$ can be served by a convex combination of configurations in which robots are allocated to serve distinct flows. Let $\tilde{\Gamma}$ be a finite set of vectors defined epoch is::

$$\tilde{\Gamma} = \left\{ \gamma | \gamma_i = \frac{a_i R_{\max}}{2}, \quad \forall\ i, a_i \in \{0,1,2\}, \sum_{i=1}^{K} a_i \le N \right\}. \quad (5)$$

For each element of this set $\tilde{\Gamma}$ the corresponding integer vector $\mathbf{a}$ corresponds to a "basis" allocation of robots to distinct sources and sinks that can service each flow at rate γ_i. Specifically, a_i refers to the number of robots allocated to service flow i. If $a_i = 1$, this means exactly one robot is allocated to flow i, and can service this flow maximally by spending half its time near the source and half the time near the sink (ignoring for now the time spent in transit), yielding a maximum service rate of $\gamma_i = R_{max}/2$. If two robots are allocated to a flow i, we have that $a_i = 2$, in which case two robots take turns spending time at the source and sink of the flow respectively for half the time each, yielding a net rate of $\gamma_i = R_{max}$. The constraints on a_i ensure that the total number of robots allocated does not exceed the available number N.

Let us refer to the convex hull of $\tilde{\Gamma}$ as $\mathcal{H}(\tilde{\Gamma})$ or, for readability, simply $\mathcal{H}$.

***Lemma** 1:* $\mathcal{H} \supset \Lambda$

Proof: First, note that the convex hull of $\tilde{\Gamma}$ can be written as follows:

$$\mathcal{H} = \left\{ \gamma | \gamma_i = \frac{a_i R_{max}}{2}, a_i \in [0,2] \quad \forall\ i, \sum_{i=1}^{K} a_i \le N \right\}. \quad (6)$$

In other words, the convex hull of the set $\tilde{\Gamma}$ is obtained by allowing a_i to vary continuously. Now using the relationship

$a_i = \frac{2\gamma_i}{R_{max}}$, we can re-express $\mathcal{H}$ as follows:

$$\mathcal{H} = \left\{ \gamma | \frac{2\gamma_i}{R_{max}} \in [0,2] \ \forall \ i, \sum_{i=1}^{K} \frac{2\gamma_i}{R_{max}} \leq N \right\}$$
$$= \left\{ \gamma | 0 \leq \gamma_i \leq R_{max} \ \forall \ i, \sum_{i=1}^{K} \gamma_i \leq \frac{R_{max} N}{2} \right\} \supset \Lambda$$

■

Each basis allocation corresponding to the elements of $\tilde{\Gamma}$ can actually be expressed as two distinct but symmetric allocations of robots to sources/sinks over two successive epochs. For the i^{th} flow, if $a_i = 0$, there is no robot allocated to either the source or sink in either of these two slots; if $a_i = 1$, a particular robot is assigned to be at the source at the first slot and at the sink at the second slot; if $a_i = 2$, two robots are assigned (call them $R1$ and $R2$) such that $R1$ is at the source at the first epoch and at the sink at the second epoch while $R2$ is at the sink at the first epoch and at the source at the second epoch.

The set $\mathcal{H}$ describes all possible robot service rates that can be obtained by a convex combination of these basis allocations. Consider a rate vector $\gamma \in \mathcal{H}$. Since it lies in the convex hull of the set $\tilde{\Gamma}$ it can be described in terms of a vector of convex coefficients α each of whose elements corresponds to a basis allocation of robots. We can therefore[1] identify n_i such that $n_i / \sum_i n_i = \alpha_i$. The given rate vector γ can then be scheduled by allocating n_i epochs each for the two parts of the i^{th} basis allocation. And after a total of $\sum_i 2n_i$ epochs, the whole schedule can be repeated. This schedule will provide the desired service rate vector γ.

Thus far the schedules have been derived under the assumption of instantaneous robot movements. Now we consider the effect of transit time. It is possible to choose T to be sufficiently large to bound the fraction of time spent in transit by ϵ, i.e. $\frac{d}{vT} < \epsilon$. Thus even while taking into account time wasted in transit, we can scale either time period of the epochs T or the velocity v so as to provide a service rate vector γ' that is arbitrarily close to any ideal service rate γ in the sense that $\gamma_i - \gamma_i' < \epsilon \ \forall \ i$.

We now state one of our main results:

***Theorem** 1:* $\mathbf{\Lambda}$ is the achievable capacity region of the network.

Proof: By construction, $\mathcal{H}$ represents the boundary of all feasible robot service rates, and as we have discussed time spent in transit can be accounted for by increasing T or v so that any arrival rate that is in the interior of $\mathcal{H}$ can be served. Since in lemma 1, we have already shown that $\Lambda \subset \mathcal{H}$, any arrival rate in Λ can be stably served.

Furthermore, $\mathcal{H}$ represents the closure of the open set Λ. Thus any arrival rate vector that is a bounded distance outside of Λ cannot be served stably (as it would also be outside of $\mathcal{H}$).

[1]Here, for ease of exposition, we are assuming that α_i is rational, otherwise it can be approximated by an arbitrarily close rational number which will not affect the overall result.

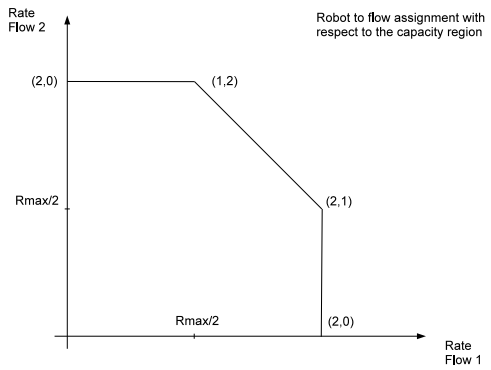

Fig. 2. Capacity region for a problem with 3 robots and 2 flows

Together, these imply that Λ is the achieveable capacity region of the network. ■

A. An Example

Figure 2 shows the capacity region when $K = 2$, $N = 3$. The labels such as (x, y) are given to the basis allocations on the Pareto boundary to denote that they can be achieved by allocating an integer number of robots x to flow 1 and y to flow 2. Note in particular that the point (R_{max}, R_{max}) is outside the region in this case because the only way to serve that rate is to allocate two robots full time to each of the two flows, and we have only 3 robots. The vertices on the boundary of the region , which represent basis allocations, are all in the set $\tilde{\Gamma}$; the convex hull of $\tilde{\Gamma}$ completely describes the region.

B. Capacity Region under finite velocity and epoch duration

The analysis thus far assumes that either the velocity of the robot or the epoch duration can be chosen to be arbitrarily large. Next, motivated by practical considerations we consider the case when v and T are finite. In particular, the restriction of T to be finite is useful for two reasons: a) it fixes the overhead of scheduling and b) it can be used to enforce a deterministic upper bound on delay (the time between generation and delivery of a given packet). As may be expected, these constraints reduce the capacity region.

The fraction of time spent in transit, is bounded by $\frac{d}{vT}$, where d is the maximum distance between the static nodes. We assume that $\frac{d}{vT} < 1$, which implies that a robot can always reach its destination (source or sink) within an epoch.

This directly yields the following inner-bound on the capacity region when v and T are finite and fixed:

$$\mathbf{\Lambda_{IB}}(\mathbf{v}, \mathbf{T}) = \left\{ \lambda | 0 \leq \lambda_i < R_{max}(1 - \frac{d}{vT}), \right.$$
$$\left. \forall \ i, \ \sum_{i=1}^{K} \lambda_i < \frac{R_{max}(1 - \frac{d}{vT})\, N}{2} \right\}. \quad (7)$$

IV. Coarse-Grained Backpressure Control

From the previous discussion, we know that if the arrival rate of each flow is known, and within the ideal capacity region of the system, the epoch duration and a service schedule for the

robots can be designed in such a way that the rate is served in a stable manner (maintaining the average size of each queue to be bounded). We consider now the case when the arrival rates are within the capacity region but not known to the scheduling algorithm, and the v and T parameters are kept fixed. Is it still possible to schedule the movement and communications of the robots in such a way that all queues remain stable?

The answer to this question turns out to be yes, using the notion of Backpressure scheduling first proposed by Tassiulas and Ephremides [3]. We propose an algorithm for scheduling message ferrying robots that achieves throughput-optimal performance for finite v and T parameters, which we refer to as coarse-grained backpressure-based message ferrying (CBMF). The CBMF algorithm works as follows.

At the beginning of each epoch:

- compute the weights $w_{src(i),j} = (Q_{src(i)} - Q^i_j)$ and $w_{sink(i),j} = (Q^i_j)$.
- If the allocation $A(i,j) = 1$, denote $w_{i,j} = w_{src(i),j}$. If $A(i,j) = -1$, denote $w_{i,j} = w_{sink(i),j}$. Else, if $A(i,j) = 0$, $w_{i,j} = 0$.
- Find the allocation A that maximizes $\sum_{i,j} |A(i,j)| w_{i,j}(A(i,j))$ subject to the following three constraints:

  (1) $\sum_i |A(i,j)| = 1$,

  (2) $\sum_j \mathcal{I}\{A(i,j) = 1\} \leq 1$,

  (3) $\sum_j \mathcal{I}\{A(i,j) = -1\} \leq 1$.

  The first constraint ensures that each robot is allocated to exactly one source or sink. The second constraint ($\mathcal{I}\{\}$ represents the indicator function) ensures that no source is allocated more than one robot, while the third constraint ensures that no sink is allocated more than one robot.

***Theorem** 2:* For any arrival that is strictly within $\mathbf{\Lambda_{IB}(v, T)}$, the CBMF algorithm ensures that all source and robot queues are stable (always bounded by a finite value).

Due to space constraints, we do not present a full proof of this theorem here, however the proof follows from bounding the Drift of a quadratic Lyapunov function and deriving a control policy that minimizes this bound, following closely the approach pioneered by Tassiulas and Ephremides [3]. The only technical complication in this setting compared to traditional backpressure as applied to static wireless networks is that the average rate obtained over the course of an epoch for each matching can be slightly different depending on the starting position of the robot with respect to the node it is being matched to. CBMF treats the rate for each matching the to be the same in its weight calculation, and as a result it is not provably stable for all arrival rates up to the outer-bound (which as discussed before is in fact achievable using scheduling with prior knowledge of the arrival rates, for a given v, T); this theoretical guarantee can only be provided for all arrival rates upto the inner-bound. However, as v and T increase, the inner-bound approaches the ideal capacity region.

V. Simulations

We present numerical simulation results for two flows and four robots. We use $\eta = 2$, $C = 1$, $d_1 = 25, d_2 = 100$, v and T and λ_i are varied as shown in the figures. In figure 3 we see the average end-to-end delay (time taken for a packet generated at the source to reach the sink; it is obtained by measuring the average total queue size for each flow in the simulations and dividing by the arrival rate, as per Little's Theorem [4]) versus arrival rate for each flow, plotted wherever CBMF results in stable queues. Also marked on the figure is the lower (inner) bound of capacity, for rates below which CBMF is provably stable. We find that we are able to get converging, bounded delays (indicative of stability) up to a point even beyong the lower (inner) bound. We see that as the velocity increases, so does the capacity, and at the same time the delay decreases. Thus improvement in robot velocity benefits both throughput and delay performance of CBMF, as may be expected. Figure 4, in which the velocity is kept constant across curves but the epoch duration is varied, is somewhat similar but with one striking difference, however, as the epoch duration increases, so does the capacity; but at the same time, the average delay also increases (for the same arrival rates, so long stability is maintained). Thus, increasing the scheduling epoch duration improves throughput but hurts delay performance.

VI. Conclusions

This paper has addressed two fundamental questions in robotic message ferrying for wireless networks: what is the throughput capacity region of such systems? How can they be scheduled to ensure stable operation, even without prior knowledge of arrival rates? There are a number of open directions suggested by the present work. The first is to improve the CBFM algorithm to support the entire capacity region (our present work only establishes its stablity within the stated inner-bound, obtained by neglecting the time spent by robots in transit); this may require a slight modification to the algorithm itself. Furthermore, we are interested in improving the delay properties of this algorithm, possibly by adapting the schedule length based on observed delay or by considering finer-grained motion control. Finally, we are interested in developing decentralized scheduling mechanisms that the robots can implement in a distributed fashion.

References

[1] M. Grossglauser and D. Tse, "Mobility Increases the Capacity of Ad Hoc Wireless Networks," *IEEE/ACM Trans. on Networking*, vol 10, no 4, August 2002.

[2] W. Zhao and M.H. Ammar, "Message ferrying: proactive routing in highly-partitioned wireless ad hoc networks," The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems, May 2003

[3] L. Tassiulas, A. Ephremides, "Stability properties of constrained queueing systems and scheduling for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, Vol. 37, No. 12, pp. 1936-1949, December 1992.

[4] A. Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, Addison-Wesley, 1993.

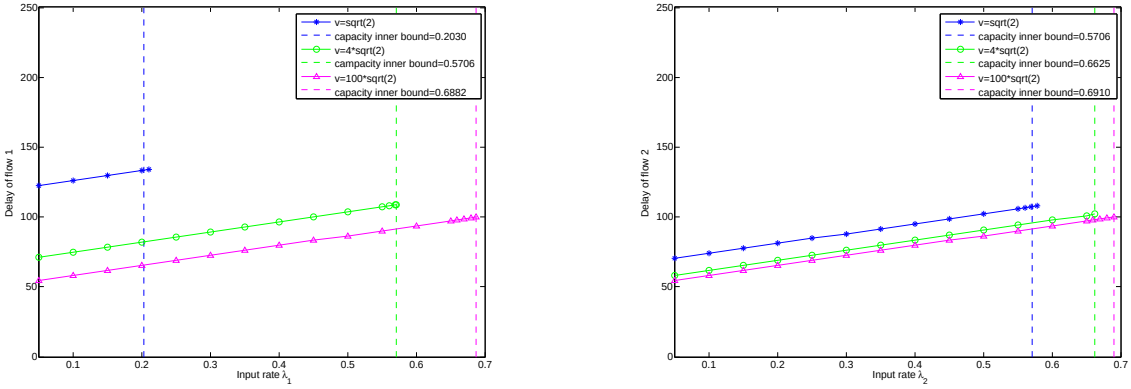

Fig. 3. Delay of flows 1 (left) and 2 (right) as we vary v for $T = 100$

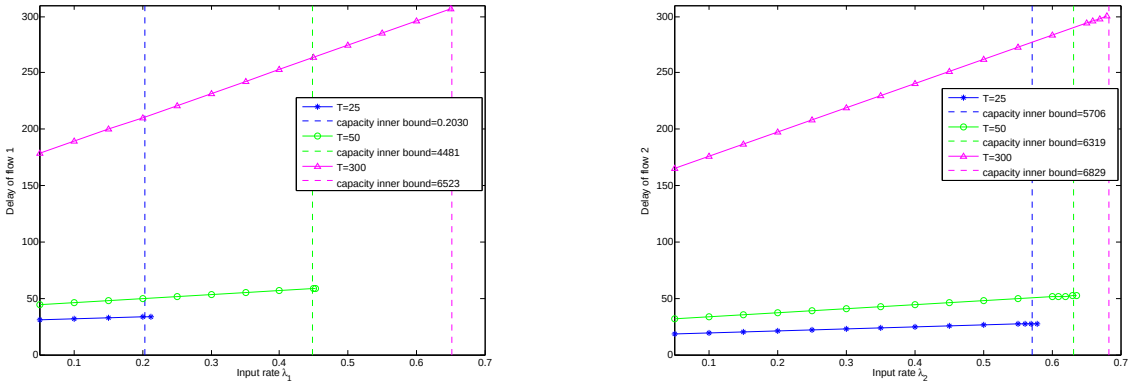

Fig. 4. Delay of flows 1 (left) and 2 (right) as we vary T for $v = 4 * \sqrt{2}$