Minimizing Flow Time in the Wireless Gathering Problem

Vincenzo Bonifaci, Peter Korteweg, Alberto Marchetti-Spaccamela, Leen Stougie

▶ **To cite this version:**

HAL Id: hal-00220388

https://hal.archives-ouvertes.fr/hal-00220388

Submitted on 28 Jan 2008

MINIMIZING FLOW TIME IN THE WIRELESS GATHERING PROBLEM

VINCENZO BONIFACI [1,3], PETER KORTEWEG [2], ALBERTO MARCHETTI-SPACCAMELA [3], AND LEEN STOUGIE [2,4]

[1] Technische Universität Berlin, Institut für Mathematik, Berlin, Germany

[2] Eindhoven University of Technology, Dept of Mathematics and Computer Science, Eindhoven, The Netherlands
E-mail address: `p.korteweg@tue.nl,l.stougie@tue.nl`

[3] University of Rome "La Sapienza", Dept of Computer and Systems Science, Rome, Italy
E-mail address: `bonifaci@dis.uniroma1.it,alberto@dis.uniroma1.it`

[4] CWI, Amsterdam, The Netherlands
E-mail address: `stougie@cwi.nl`

ABSTRACT. We address the problem of efficient data gathering in a wireless network through multi-hop communication. We focus on the objective of minimizing the maximum flow time of a data packet. We prove that no polynomial time algorithm for this problem can have approximation ratio less than $\Omega(m^{1/3})$ when m packets have to be transmitted, unless $\mathbf{P} = \mathbf{NP}$. We then use resource augmentation to assess the performance of a FIFO-like strategy. We prove that this strategy is 5-speed optimal, i.e., its cost remains within the optimal cost if we allow the algorithm to transmit data at a speed 5 times higher than that of the optimal solution we compare to.

1. Introduction

Wireless networks are used in many areas of practical interest, such as mobile phone communication, ad-hoc networks, and radio broadcasting. Moreover, recent advances in miniaturization of computing devices equipped with short range radios have given rise to strong interest in sensor networks for their relevance in many practical scenarios (environment control, accident monitoring etc.) [1, 16].

In many applications of wireless networks data gathering is a critical operation for extracting useful information from the operating environment: information collected from multiple nodes in the network should be transmitted to a sink that may process the data, or act as a gateway to other networks. We remark that in the case of wireless sensor networks

sensor nodes have limited computation capabilities, thus implying that data gathering is an even more crucial operation. For this reasons, data gathering in sensor networks has received significant attention in the last few years; we cite just a few contributions [1, 10]. The problem finds also applications in Wi-Fi networks when many users need to access a gateway using multi-hop wireless relay-routing [5].

In this paper we focus on the problem of designing and analysing *simple distributed* algorithms that have *good approximation guarantees* in *realistic scenarios*. Namely, we are interested in algorithms that are not only distributed but that are fast and can be implemented with limited overhead: sophisticated algorithms that require solving complex combinatorial optimization problems are impractical for implementations and have mainly theoretical interest.

In order to formally assess the performance of the proposed algorithms we focus on the minimization of the maximum flow, i.e. minimizing the maximum time spent in the system by a packet. Almost all of the previous literature considered the objective of minimizing the completion time (see for example [3, 4, 5, 10, 11, 13, 17]). Flow minimization is a largely used criterion in scheduling theory that more suitably allows to assess the quality of service provided when multiple requests occur over time [7, 8, 12, 15].

The problem of modelling realistic scenarios of wireless sensor networks is complicated by the many parameters that define the communication among nodes and influence the performance of transmissions (see for example [1, 18]). In the sequel we assume that stations have a common clock, hence time can be divided into rounds. Each node is equipped with a half-duplex interface; as a result it cannot send and receive during the same round. Typically, not all nodes in the network can communicate with each other directly, hence packets have to be sent through several nodes before they can be gathered at the sink; this is called *multi-hop* routing.

The key issue in our setting is *interference*. A radio signal has a *transmission* radius, the distance over which the signal is strong enough to send data, and an *interference* radius, the distance over which the radio signal is strong enough to interfere with other radio signals. If node i is transmitting data to node j we have *interference* (or *collision*) in communication if j also receives signals from other stations at the same time. Following Bermond et al. [5], we model the wireless network using a graph and a parameter d_I. An edge between nodes i and j represents the fact that stations i and j are within transmission range of each other. The parameter d_I models the interference radius: a node j successfully receives a signal if one of his neighbors is transmitting, and no other node within distance d_I from j is transmitting in the same round. The case $d_I = 1$ has been extensively considered earlier (see e.g. [4, 10, 11]); but we remark that assuming $d_I = 1$ or assuming that interferences/transmissions are modeled according to the well known unit disk graph model *does not* adequately represent interferences as they occur in practice [18].

Kumar et al. [14] give an overview of other interference models, including the so-called *distance-2 interference model*. The distance-2 interference model is similar to our interference model with $d_I = 1$, plus the extra constraint that no two transmitting nodes should be adjacent; we observe that this requirement might pose unnecessary conditions.

The Wireless Gathering Problem. An instance of the *Wireless Gathering Problem* (WGP) is given by a static wireless network which consists of several stations (nodes) and one base station (the sink), modeled as a graph, together with the interference radius d_I; over time data packets arrive at stations that have to be gathered at the base station.

A feasible solution of an instance of WGP is a schedule without interference which determines for each packet both route and times at which it is sent. As we will see in Section 2 this can be modeled as a clean combinatorial optimization problem. We consider two different objectives. One is to minimize completion time, i.e., the time needed to gather all the packets. Another, perhaps more natural, objective is minimization of maximum flow time of packets, i.e., the maximum difference between release time and arrival time at the sink of a packet. We call these two problems C-WGP and F-WGP, respectively.

Related work. The Wireless Gathering Problem was introduced by Bermond et al. [5] in the context of wireless access to the Internet in villages. The authors proved that the problem of minimizing the completion time is **NP**-hard and presented a greedy algorithm with asymptotic approximation ratio at most 4. In [6] we considered arbitrary release times and proposed an on-line greedy algorithm with the same approximation ratio.

Bar-Yehuda et al. [4] considered distributed algorithms for C-WGP. Their model is a special case of our model, where $d_I = 1$ and there are no release dates. Kumar et al. [13] studied the more general *end-to-end* transmission problem, where each of the packets may have a different source *and* destination in the network. Under the assumption of a distance-2 interference model, Kumar et al. considered the objective of minimizing the maximum completion time of the schedule, and presented hardness results and approximation algorithms for arbitrary graphs and disk graphs. They presented distributed algorithms for packet scheduling over fixed routing paths, and used a linear program in order to determine the paths. By contrast, we use a shortest paths tree to fix the routing paths, which is easier to implement in a distributed setting.

Florens et al. [10] considered the minimization of the completion time of data gathering in a setting with unidirectional antennas. They provided a 2-approximation algorithm for tree networks and an optimal algorithm for line networks. Gargano and Rescigno [11] gave a polynomial time algorithm for the special case of the same model in which each node has exactly one packet to send.

Another related problem is to compute the throughput of a wireless network. This has been studied for example in [14]. We also observe that many papers study broadcasting in wireless networks [3, 17]. However, we stress that data gathering and broadcasting are substantially different tasks in the context of packet networks. In particular, the idea of reversing a broadcast schedule to obtain a gathering schedule (which works when data can be aggregated) cannot be used.

Main results. In Section 3 we give inapproximability results for F-WGP. We prove that F-WGP on m packets cannot be approximated within $\Omega(m^{1/3})$, unless $\mathbf{P} = \mathbf{NP}$. We also show that any algorithm using shortest paths in order to route the packets to the sink is no better than an $\Omega(m)$-approximation.

In Section 4 we present a polynomial time resource augmented approximation algorithm for F-WGP which is in fact an on-line algorithm. We use resource augmentation because F-WGP is hard to approximate within a reasonable factor.

Resource augmentation was introduced in the context of machine scheduling in [12]: the idea is to study the performance of on-line algorithms which are given processors faster than the adversary. Intuitively, this has been done to compensate an on-line scheduler for its lack of future information. Such an approach has led to a number of interesting results showing that moderately faster processors are sufficient to attain satisfactory performance guarantee for various scheduling problems, e.g. [8, 12]

Surprisingly, in the case of F-WGP a modest resource augmentation allows to compensate not only the lack of future information but also the approximation hardness of the problem: we present a σ-speed optimal algorithm for F-WGP and C-WGP; σ depends on d_I and is at most 5.

We remark that our algorithm can be implemented using local information only: in particular, it suffices that a node is informed about the state of nodes within distance $d_I + 1$. On the other hand, our lower bounds hold for centralized algorithms as well.

2. Mathematical preliminaries

We formulate WGP as a graph optimization problem. The model we use can be seen as a generalization of a well-known model for packet radio networks [3, 4].

An instance of WGP consists of a graph $G = (V, E)$, a *sink* node $s \in V$, a positive integer d_I, and a set of *data packets* $J = \{1, 2, \ldots, m\}$. Each packet $j \in J$ has an *origin* $o_j \in V$ and a *release date* $r_j \in \mathbb{Z}_+$.

We assume that time is discrete; we call a time unit a *round*. The rounds are numbered $0, 1, 2, \ldots$. During each round a node may either be *sending* a packet, be *receiving* a packet or be *inactive*. If two nodes u and v are adjacent, then u can send a packet to v during a round. If node u sends a packet j to v in some round, then the pair (u, v) is said to be a *call* from u to v. For each pair of nodes $u, v \in V$, the *distance* between u and v, denoted by $d(u, v)$, is the length of a shortest path from u to v in G. Two calls (u, v) and (u', v') *interfere* if they occur in the same round and either $d(u', v) \leq d_I$ or $d(u, v') \leq d_I$; otherwise the calls are *compatible*. For this reason, the parameter d_I is called the *interference radius*. The special case of a unit interference radius corresponds to the above cited model of Bar-Yehuda et al. [3].

For every packet $j \in J$, the release date r_j specifies the time at which the packet enters the network, i.e. packet j cannot be sent before round r_j. In the off-line version the entire instance is completely known at time 0; in the on-line version information about a packet becomes known only at its release date.

A solution for a WGP instance is a schedule of compatible calls such that all packets are ultimately sent to the sink. Notice that while in principle each radio transmission can broadcast the same packet to multiple destinations, in the gathering problem having more than one copy of the same packet does not help, as it suffices to keep the one that will arrive first at the sink. Thus, we assume that at any time there is a unique copy of each packet. Also, in the model we consider, packets cannot be aggregated.

Given a schedule, let x_j^t be the unique node holding packet j at time t. The integer $C_j := \min\{t : x_j^t = s\}$ is called the *completion time* of packet j, while $F_j := C_j - r_j$ is the *flow time* of packet j. In this paper we are interested in the minimization of $\max_j F_j$ (F-WGP). As an intermediate step in the analysis of F-WGP, we also study the minimization of $\max_j C_j$ (C-WGP).

Some auxiliary notation, we denote by $\delta_j := d(o_j, s)$ the minimum number of calls required for packet j to reach s. We also define $\gamma := d_I + 2$, and $\gamma_0 := \lfloor (d_I + 1)/2 \rfloor$.

We analyze the performance of our algorithms using the standard worst case analysis techniques of approximation ratio analysis, as well as resource augmentation. Given a WGP instance $\mathcal{I}$ and an algorithm ALG, we define $\mathcal{C}(\mathcal{I})$ as the cost of ALG and $\mathcal{C}^*(\mathcal{I})$ as the cost of the optimal solution on $\mathcal{I}$. A polynomial-time algorithm is called an α-approximation if for any instance $\mathcal{I}$ we have $\mathcal{C}(\mathcal{I}) \leq \alpha \cdot \mathcal{C}^*(\mathcal{I})$.
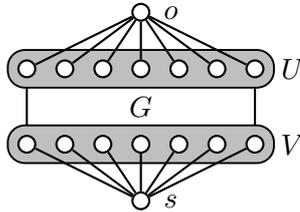
Figure 1: The construction in the proof of Theorem 3.2

In the resource augmentation paradigm, the algorithm is allowed to use more resources than the adversary. We consider resource augmentation based on speed, in which we assume that the algorithm can schedule compatible calls with higher speed than the optimal algorithm. For any $\sigma \geq 1$, we call an algorithm a σ-speed algorithm if the time used by the algorithm to schedule a set of compatible calls is $1/\sigma$ time units. See [2] for more information on approximation algorithms, and [12] for more on resource augmentation.

3. Inapproximability

In this section we prove an inapproximability result for F-WGP. To prove this result we consider the so-called *induced matching* problem. A matching M in a graph G is an *induced matching* if no two edges in M are joined by an edge of G. The following rather straightforward relation between compatible calls in a bipartite graph and induced matchings will be crucial in the following.

Proposition 3.1. *Let $G = (U, V, E)$ be a bipartite graph with node sets (U, V) and edge set E. Then, a set $M \subseteq E$ is an induced matching if and only if the calls corresponding to edges of M, directed from U to V, are all pairwise compatible, assuming $d_I = 1$.* ∎

INDUCED BIPARTITE MATCHING (IBM)
Instance: a bipartite graph G and an integer k.
Question: does G have an induced matching of size at least k?

We will use the fact that the optimization version of IBM is hard to approximate: there exists an $\alpha > 1$ such that it is **NP**-hard to distinguish between graphs with induced matchings of size k and graphs in which all induced matchings are of size at most k/α. The current best bound for α is $6600/6599$ [9].

Theorem 3.2. *Unless $\mathbf{P} = \mathbf{NP}$, no polynomial-time algorithm can approximate F-WGP within a ratio better than $\Omega(m^{1/3})$.*

Proof. Let (G, k) be an instance of IBM, $G = (U, V, E)$. We construct a 4-layer network with a unique source o (first layer), a clique on U and a clique on V (middle layers), and a sink s (last layer). Source o is adjacent to each node in U, and s to each node in V. The edges between U and V are the same as in G (see Figure 1). We set $d_I = 1$.

The F-WGP instance consists of $m := (1 - 1/\alpha)^{-1}(1 + k/\alpha)(2k + 1)k = \Theta(k^3)$ packets with origin o. They are divided into m/k groups of size k. Each packet in the hth group has release date $(k + 1)h$, $h = 0, \dots, m/k - 1$. Rounds $(k + 1)h$ till $(k + 1)(h + 1) - 1$ together are a *phase*.

We prove that if G has an induced matching of size k, there is a gathering schedule of cost $2k + 1$, while if G has no induced matching of size more than k/α, every schedule has cost at least $(2k + 1)k = (2k + 1)\Theta(m^{1/3})$. The theorem then follows directly.

Assume G has an induced matching M of size k, say (u_i, v_i), $i = 0 \ldots k - 1$. Then consider the following gathering schedule. In each phase, the k new packets at o are transmitted, necessarily one-by-one, to layer U while old packets at layer V (if any) are absorbed at the sink; then, in a *single* round, the k new packets move from U to V via the matching edges. More precisely, each phase can be scheduled in $k + 1$ rounds as follows:
1. for $i = 0, \ldots, k - 1$ execute in the ith round the two calls (o, u_i) and $(v_{i+1 \bmod k}, s)$;
2. in the kth round, execute simultaneously all the calls (u_i, v_i), $i = 0, \ldots, k - 1$.
The maximum flow time of the schedule is $2k + 1$, as a packet released in phase h reaches the sink before the end of phase $h + 1$.

In the other direction, assume that each induced matching of G is of size at most k/α. By Proposition 3.1, at most k/α calls can be scheduled in any round from layer U to layer V. We ignore potential interference between calls from o to U and calls from V to s; doing so may only decrease the cost of a schedule. As a consequence, we can assume that each packet follows a shortest path from o to s. Notice however that, due to the cliques on the layers U and V, no call from U to V is compatible with a call from o to U, or with a call from V to s.

Let m_o and m_U be the number of packets at o and U, respectively, at the beginning of a given phase. Also, let $\beta := 1 + k/\alpha$. We associate to the phase a potential value $\psi := \beta m_o + m_U$, and we show that at the end of the phase the potential will have increased proportionally to k. Let c_o and c_U denote the number of calls from o to U and from U to V, respectively, during the phase. Since a phase consists of $k + 1$ rounds, and in each round at most k/α calls are scheduled from U to V, we have $c_o + c_U/(k/\alpha) \leq k + 1$, or, equivalently since $k/\alpha = \beta - 1$,

$$(\beta - 1)c_o + c_U \leq (\beta - 1)(k + 1). \tag{3.1}$$

If m_o', m_U' are the number of packets at o and U at the beginning of the next phase, and $\psi' = \beta m_o' + m_U'$ is the new potential, we have

$$
\begin{aligned}
m_o' &= m_o + k - c_o \\
m_U' &= m_U + c_o - c_U \\
\psi' - \psi &= \beta(m_o' - m_o) + m_U' - m_U \\
&= \beta(k - c_o) + c_o - c_U \\
&= \beta k - (\beta - 1)c_o - c_U \\
&\geq \beta k - (\beta - 1)(k + 1) \\
&= k - (\beta - 1) \\
&= (1 - 1/\alpha)k
\end{aligned}
$$

where the inequality uses (3.1).

Thus, consider the situation after m/k phases. The potential has become at least $\Psi := (1 - 1/\alpha)m$. By definition of the potential, this implies that at least $\Psi/\beta = (1 - 1/\alpha)(1 + k/\alpha)^{-1}m = (2k+1)k$ packets reside at either o or U; in particular, they have been released but not yet absorbed at the sink. Since the sink cannot receive more than one packet per round, this clearly implies a maximum flow time of $(2k+1)k = (2k+1)\Theta(m^{1/3})$ for one of these packets.   ∎

In cases where the packets are routed via shortest paths to the sink – a behavior common to many gathering protocols – the result of Theorem 3.2 can be strengthened further.

Theorem 3.3. *No algorithm that routes packets along shortest paths can approximate* F-WGP *within a ratio better than* $\Omega(m)$.

Proof. Consider the instance in Figure 2. The adversary releases a message at each of the nodes u_1, u_2, u_3 at times $5i$, $i = 0, \ldots, m/3$. Any shortest paths following algorithm sends all messages via u, yielding $\max_j C_j \geq 3m$. As $r_j \leq 5m/3$ for each message j, we have $\max_j F_j \geq 3m - 5m/3 = 4m/3$.

The adversary sends each message over the path which does not contain u. We claim that it is possible to do this so that all messages released at time $5i$ arrive at the sink in round $5(i+1)+1$ latest. If the claim holds, then we have $\max_j F_j^* \leq 5(i+1)+1-5i = 6$, from which the theorem will follow.

We prove the claim by induction. Suppose the claim holds for messages released in round $5(i-1)$. Then, the last message released at time $5(i-1)$ latest is sent to the sink in round $5i$. This message does not block any message released in round $5i$. Now, the adversary sends the messages released in round $5i$ to a node adjacent to s in 3 rounds, i.e. in the rounds $5i$, $5i+1$ and $5i+2$. Then, it requires another 3 rounds to send all 3 messages to the sink, i.e. the rounds $5i+3$, $5i+4$, and $5(i+1)$. This proves the theorem, since $\max_j F_j / \max_j F_j^* \geq (4m/3)/6 = 2m/9$. ∎
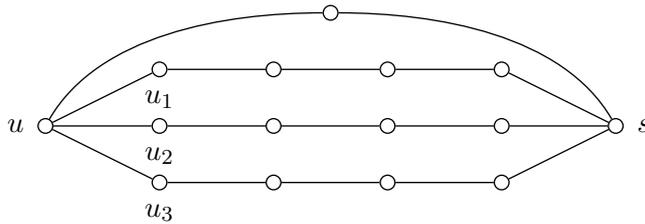

Figure 2: No shortest path based algorithm is better than $\Omega(m)$-approximate ($d_I = 1$).

4. Approximation Algorithms

In this section we present and analyze a FIFO algorithm for WGP. First, we show that FIFO is a 5-approximation for C-WGP. Note that the best approximation algorithm known is 4-approximate; the main interest in analyzing FIFO is that we use it as a subroutine in an algorithm for F-WGP which uses resource augmentation. Next, we prove that this algorithm with resource augmentation is a σ-speed optimal algorithm, for any $\sigma \geq 5$, for both C-WGP and F-WGP.

4.1. An approximation algorithm for C-WGP

We will present an approximation algorithm for C-WGP. The algorithm we consider is actually a special case of a general scheme for which we can prove an upper bound on the completion time [6]. In this scheme, called PRIORITY GREEDY, each packet is assigned a unique priority based on some algorithm-specific rules. Then, in each round, packets are considered in order of decreasing priority and are sent towards the sink as long as there is no interference with higher priority packets.

Algorithm 4.1 (PRIORITY GREEDY)**.** In every round, consider the available packets in order of decreasing priority, and send each next packet along a shortest path from its current node to s, as long as this causes no interference with any higher-priority packet.

We first derive upper bounds on the completion time C_j of each packet j in a PRIORITY GREEDY solution.

We say that packet j is *blocked* in round t if $t \geq r_j$ but j is not sent in round t. Note that in a PRIORITY GREEDY algorithm a packet can only be blocked due to interference with a higher priority packet. We define the following *blocking relation* on a PRIORITY GREEDY schedule: $k \prec j$ if in the last round in which j is blocked, k is the packet closest to j that is sent in that round and has a priority higher than j (ties broken arbitrarily). The blocking relation induces a directed graph $F = (J, A)$ on the packet set J with an arc (k, j) for each $k, j \in J$ such that $k \prec j$. Observe that, for any PRIORITY GREEDY schedule, F is a directed forest and the root of each tree of F is a packet which is never blocked. For each j let $T(j) \subseteq F$ be the tree of F containing j, $b(j) \in J$ be the root of $T(j)$, and $P(j)$ the set of packets along the path in F from $b(j)$ to j. Finally, define $\pi_j := \min\{\delta_j, \gamma_0\}$ and $R_j := r_j + \delta_j - \pi_j$.

We have upper and lower bounds on the completion time of a packet.

Lemma 4.2 ([6]). *For each packet $j \in J$, $C_j \leq R_{b(j)} + (\gamma/\gamma_0) \cdot \sum_{i \in P(j)} \pi_i$.*

Lemma 4.3 ([6]). *Let $S \subseteq J$ be a nonempty set of packets, and let C_i^* denote the completion time of packet i in some feasible schedule. Then there is $k \in S$ such that $\max_{i \in S} C_i^* \geq R_k + \sum_{i \in S} \pi_i$.*

Our algorithm is based on a version of the PRIORITY GREEDY scheme, in which a higher priority is given to packets with earlier release dates (ties broken arbitrarily). We call this algorithm FIFO after the famous *first-in-first-out* algorithm in scheduling and service systems, though in our case packets do not necessarily arrive in order of their priority at the sink.

Theorem 4.4. FIFO *is a $(1 + \gamma/\gamma_0)$-approximation algorithm for* C-WGP.

Proof. Let j be the packet having maximum C_j, and consider $T(j)$, the tree containing j in the forest induced by the blocking relation. We can apply Lemma 4.3 with $S = T(j)$ to obtain

$$\max_{i \in T(j)} C_i^* \geq r_k + \delta_k + \sum_{\substack{i \in T(j) \\ i \neq k}} \pi_i \tag{4.1}$$

where k is some packet in $T(j)$. On the other hand, by using Lemma 4.2,

$$
\begin{aligned}
C_j &\leq R_{b(j)} + \frac{\gamma}{\gamma_0} \sum_{i \in P(j)} \pi_i \tag{4.2} \\
&= r_{b(j)} + \delta_{b(j)} - \pi_{b(j)} + \frac{\gamma}{\gamma_0} \sum_{i \in P(j)} \pi_i \\
&\leq r_{b(j)} + \frac{\gamma}{\gamma_0} \min\{\delta_k, \gamma_0\} + \frac{\gamma}{\gamma_0} \sum_{\substack{i \in P(j) \\ i \neq k}} \pi_i + \delta_{b(j)} \\
&\leq \frac{\gamma}{\gamma_0} \left(r_k + \delta_k + \sum_{\substack{i \in T(j) \\ i \neq k}} \pi_i \right) + \delta_{b(j)}.
\end{aligned}
$$

where we used the fact that, by definition of FIFO, we have $r_{b(j)} \leq r_k$. Equations (4.1) and (4.2), and observation $\max_{i \in T(j)} C_i^* \geq \delta_{b(j)}$ prove the theorem. ∎

It is straightforward to verify that $2 \leq \gamma/\gamma_0 \leq 4$ for all d_I, while $\gamma/\gamma_0 = 3$ for $d_I = 1$.

Corollary 4.5. FIFO *is a* 5-*approximation algorithm for* C-WGP. *When* $d_I = 1$, FIFO *is a* 4-*approximation for* C-WGP.

The bound on the approximation ratio of FIFO is slightly worse than that of a PRIORITY GREEDY algorithm based on R_j, which is a γ/γ_0-approximation. In fact, we also have an example on which FIFO is strictly worse than a γ/γ_0-approximation (we omit the example here due to space limitations). However, we remark that FIFO is both natural and simple; and, perhaps more importantly, Theorem 4.4 will be instrumental in proving good bounds for the minimization of maximum flow time, where we will use FIFO as a subroutine of our algorithm.

4.2. A resource augmentation bound for F-WGP

Motivated by the hardness result of Section 3, we study algorithms under resource augmentation. In this context we study σ-speed algorithms, in which data packets are sent at a speed that is σ times faster than the solution we compare to.

Algorithm 4.6 (σ-FIFO).
1. Create a new instance $\mathcal{I}'$ by multiplying release dates: $r_j' := \sigma r_j$;
2. Run FIFO on $\mathcal{I}'$;
3. Speed up the schedule thus obtained by a factor of σ.

The schedule constructed by σ-FIFO is a feasible σ-speed solution to the original problem because of step 1. We will show that σ-FIFO is optimal for both C-WGP and F-WGP, if $\sigma \geq \gamma/\gamma_0 + 1$. The following Lemma is crucial.

Lemma 4.7. *If* σ-FIFO *is a* σ-*speed optimal algorithm for* C-WGP, *then it is also a* σ-*speed optimal algorithm for* F-WGP.

Proof. Let F_j^* and $F_{j,\sigma}$ be the flow time of data packet j in an optimal solution and in a σ-FIFO solution, respectively, to F-WGP and let C_j^* and $C_{j,\sigma}$ be the completion time of data packet j in the same solutions. Suppose σ-FIFO is a σ-speed optimal algorithm for C-WGP, hence we have $\max_{j \in J} C_{j,\sigma} \leq \max_{j \in J} C_j^*$. We show that this inequality implies, for any time t,

$$\max_{j \in J, \, r_j = t} C_{j,\sigma} \leq \max_{j \in J, \, r_j \leq t} C_j^*. \tag{4.3}$$

We prove inequality (4.3) by contradiction. Suppose it is false, then there is an instance $\mathcal{I}$ of minimum size (number of data packets) for which it is false. Also, let t_0 be the first round in such an instance for which it is false. By definition, σ-FIFO schedules each data packet j definitively in round r_j; no data packet is rescheduled in a later round. I.e., the algorithm determines the completion time $C_{j,\sigma}$ of data packet j in round r_j. If the inequality is false, then we must have

$$C_{i,\sigma} > \max_{j \in J, \, r_j \leq t_0} C_j^*, \tag{4.4}$$

for some data packet i with $r_i = t_0$, and because $\mathcal{I}$ is a minimum size instance the instance does not contain any data packets released after round t_0. But then (4.4) contradicts

$\max_{j \in J} C_{j,\sigma} \leq \max_{j \in J} C_j^*$. Using (4.3) we have

$$
\begin{aligned}
\max_{j \in J} F_{j,\sigma} &= \max_t \left(\max_{j \in J, r_j = t} C_{j,\sigma} - t \right) \leq \max_t \left(\max_{j \in J, r_j \leq t} C_j^* - t \right) \\
&\leq \max_t \left(\max_{j \in J, r_j \leq t} F_j^* \right) = \max_{j \in J} F_j^*.
\end{aligned}
$$

∎

Theorem 4.8. *For $\sigma \geq \gamma/\gamma_0 + 1$, σ-FIFO is a σ-speed optimal algorithm for both C-WGP and F-WGP.*

Proof. By Lemma 4.7, it suffices to prove that σ-FIFO is σ-speed optimal for C-WGP.

Let C_j be the completion time of any data packet j in the σ-FIFO solution on instance $\mathcal{I}$, and let C_j' be the completion time of j in the FIFO solution on the instance $\mathcal{I}'$ (see the algorithm description). By construction $C_j = C_j'/\sigma$. Let $R_j' := \sigma r_j + \delta_j - \pi_j$. Then the upper bound of Lemma 4.2 applied to instance $\mathcal{I}'$ implies $C_j' \leq R_{b(j)}' + (\sigma - 1) \sum_{i \in P(j)} \pi_i$. Hence,

$$
C_j = C_j'/\sigma \leq \frac{1}{\sigma} R_{b(j)}' + \frac{\sigma - 1}{\sigma} \sum_{i \in P(j)} \pi_i \quad \leq \quad r_{b(j)} + \frac{1}{\sigma} \delta_{b(j)} + \frac{\sigma - 1}{\sigma} \sum_{i \in P(j)} \pi_i. \quad (4.5)
$$

Since in any solution $b(j)$ has to reach the sink we clearly have

$$
\max_{i \in P(j)} C_i^* \geq C_{b(j)}^* \geq r_{b(j)} + \delta_{b(j)}. \quad (4.6)
$$

Also, by Lemma 4.3, for some $k \in P(j)$,

$$
\max_{i \in P(j)} C_i^* \geq R_k + \sum_{i \in P(j)} \pi_i \geq r_k + \sum_{i \in P(j)} \pi_i \geq r_{b(j)} + \sum_{i \in P(j)} \pi_i, \quad (4.7)
$$

where the last inequality follows from $b(j)$ having lowest release time in $P(j)$, by definition of FIFO. Combining (4.5), (4.6) and (4.7), we obtain

$$
\begin{aligned}
\max_{i \in P(j)} C_i^* &= \frac{1}{\sigma} \max_{i \in P(j)} C_i^* + \frac{\sigma - 1}{\sigma} \max_{i \in P(j)} C_i^* \\
&\geq \frac{1}{\sigma} \left(r_{b(j)} + \delta_{b(j)} \right) + \frac{\sigma - 1}{\sigma} \left(r_{b(j)} + \sum_{i \in P(j)} \pi_i \right) \\
&= r_{b(j)} + \frac{1}{\sigma} \delta_{b(j)} + \frac{\sigma - 1}{\sigma} \sum_{i \in P(j)} \pi_i \geq C_j.
\end{aligned}
$$

∎

Corollary 4.9. *5-FIFO is a 5-speed optimal algorithm for C-WGP and F-WGP.*

4.3. Another upper bound for FIFO

As we have seen in Section 3, F-WGP is extremely hard to approximate without resource augmentation – no bound better than $\Omega(m^{1/3})$ is possible. Moreover, algorithms that route along shortest paths cannot do better than $\Omega(m)$ (recall Theorem 3.3). In this section we show that FIFO is in fact an $O(m)$-approximation for F-WGP. Thus, apart from constant factors, FIFO is best possible among algorithms that use shortest paths.

Theorem 4.10. FIFO *is an $O(m)$-approximation for* F-WGP.

Proof. Since every packet must be gathered at the sink, clearly $\max_j F_j^* \geq \max_j \delta_j \geq \max_j \pi_j$. Now let j be the packet incurring the maximum flow time in the schedule obtained by FIFO. Since $r_j \geq r_{b(j)}$ (by definition of FIFO), we have

$$R_{b(j)} - r_j = r_{b(j)} + \delta_{b(j)} - \pi_{b(j)} - r_j \leq \delta_{b(j)} \tag{4.8}$$

Using Lemma 4.2 and (4.8), we get

$$
\begin{aligned}
F_j = C_j - r_j &\leq R_{b(j)} - r_j + \frac{\gamma}{\gamma_0} \sum_{i \in P(j)} \pi_i \\
&\leq \delta_{b(j)} + \frac{\gamma}{\gamma_0} \sum_{i \in P(j)} \pi_i \\
&\leq \max_i F_i^* + \frac{\gamma}{\gamma_0} \cdot |P(j)| \cdot \max_i F_i^* \\
&\leq \left(1 + \frac{\gamma}{\gamma_0} m\right) \max_i F_i^*.
\end{aligned}
$$

■

5. Conclusion

We considered the wireless gathering problem with the objective of minimizing the maximum flow time of data packets (F-WGP). We showed that the simple on-line algorithm FIFO has favorable behavior: although the problem is extremely hard to approximate in general, augmenting the transmission rate by a factor of 5 allows FIFO to remain within the cost of an optimal solution for the problem without augmentation.

It is an open question whether optimality can be achieved by augmenting the transmission rate by a factor smaller than 5, and whether an efficient algorithm exists that matches the $\Omega(m^{1/3})$ lower bound on the approximability of F-WGP.

Another interesting set of questions concerns resource augmentation by allowing the algorithms to use extra frequencies, meaning that more than one data packet can be sent simultaneously over the same channel. For instance, does there exist a 5-frequency optimal FIFO-type algorithm?

For the minimization of the completion time (C-WGP), the existence of a polynomial time approximation scheme is still open. It is known that no algorithm that uses shortest paths to route the data packets to the sink can give an improvement over the currently best approximation ratio [6]. It is a challenge to design and analyze congestion avoiding algorithms with better ratios.

Acknowledgments

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.

[2] G. Ausiello, M. Protasi, A. Marchetti-Spaccamela, G. Gambosi, P. Crescenzi, and V. Kann. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer-Verlag, 1999.

[3] R. Bar-Yehuda, O. Goldreich, and A. Itai. On the time-complexity of broadcast in multi-hop radio networks: an exponential gap between determinism and randomization. *Journal of Computer and Systems Sciences*, 45(1):104–126, 1992.

[4] R. Bar-Yehuda, A. Israeli, and A. Itai. Multiple communication in multihop radio networks. *SIAM Journal on Computing*, 22(4):875–887, 1993.

[5] J. Bermond, J. Galtier, R. Klasing, N. Morales, and S. Pérennes. Hardness and approximation of gathering in static radio networks. *Parallel Processing Letters*, 16(2):165–183, 2006.

[6] V. Bonifaci, P. Korteweg, A. Marchetti-Spaccamela, and L. Stougie. An approximation algorithm for the wireless gathering problem. In *Proc. 10th Scandinavian Workshop on Algorithm Theory*, pages 328–338, 2006.

[7] H.-L. Chan, T. W. Lam, and K.-S. Liu. Extra unit-speed machines are almost as powerful as speedy machines for competitive flow time scheduling. In *Proc. 17th Symp. on Discrete Algorithms*, pages 334–343, 2006.

[8] C. Chekuri, A. Goel, S. Khanna, and A. Kumar. Multi-processor scheduling to minimize flow time with epsilon resource augmentation. In *Proc. 36th Symp. on Theory of Computing*, pages 363–372, 2004.

[9] W. Duckworth, D. Manlove, and M. Zito. On the approximability of the maximum induced matching problem. *Journal of Discrete Algorithms*, 3(1):79–91, 2005.

[10] C. Florens, M. Franceschetti, and R. J. McEliece. Lower bounds on data collection time in sensory networks. *IEEE Journal on Selected Areas in Communications*, 22:1110–1120, 2004.

[11] L. Gargano and A. A. Rescigno. Optimally fast data gathering in sensor networks. In *Proc. 31st Symp. on Mathematical Foundations of Computer Science*, pages 399–411, 2006.

[12] B. Kalyanasundaram and K. Pruhs. Speed is as powerful as clairvoyance. *Journal of the ACM*, 47(4):617–643, 2000.

[13] V. S. Anil Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. End-to-end packet-scheduling in wireless ad-hoc networks. In J. I. Munro, editor, *Proc. 15th Symp. on Discrete Algorithms*, pages 1021–1030, 2004.

[14] V. S. Anil Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. Algorithmic aspects of capacity in wireless networks. In *Measurement and Modeling of Computer Systems*, pages 133–144, 2005.

[15] J. McCullough and E. Torng. SRPT optimally utilizes faster machines to minimize flow time. In J. I. Munro, editor, *Proc. 15th Symp. on Discrete Algorithms*, pages 350–358, 2004.

[16] K. Pahlavan and A. H. Levesque. *Wireless information networks*. Wiley, New York, 1995.

[17] A. Pelc. Broadcasting in radio networks. In *Handbook of Wireless Networks and Mobile Computing*, pages 509–528. Wiley and Sons, 2002.

[18] S. Schmid and R. Wattenhofer. Algorithmic models for sensor networks. In *Proc. 20th Int. Parallel and Distributed Processing Symposium*, 2006.