Wake up and Join Me! an Energy-Efficient Algorithm for Maximal Matching in Radio Networks

Varsha Dani ✉
Department of Computer Science, Rochester Institute of Technology, NY, USA

Aayush Gupta ✉
Department of Computer Science, University of New Mexico, Albuquerque, NM, USA

Thomas P. Hayes ✉
Dept. of Computer Science, University of New Mexico, Albuquerque, NM, USA

Seth Pettie ✉
Computer Science and Engineering, University of Michigan, Ann Arbor, MI, USA

── Abstract ──

We consider networks of small, autonomous devices that communicate with each other wirelessly. Minimizing energy usage is an important consideration in designing algorithms for such networks, as battery life is a crucial and limited resource. Working in a model where both sending and listening for messages deplete energy, we consider the problem of finding a maximal matching of the nodes in a radio network of arbitrary and unknown topology.

We present a distributed randomized algorithm that produces, with high probability, a maximal matching. The maximum energy cost per node is $O(\log^2 n)$, and the time complexity is $O(\Delta \log n)$. Here n is any upper bound on the number of nodes, and Δ is any upper bound on the maximum degree; n and Δ are parameters of our algorithm that we assume are known *a priori* to all the processors. We note that there exist families of graphs for which our bounds on energy cost and time complexity are simultaneously optimal up to polylog factors, so any significant improvement would need additional assumptions about the network topology.

We also consider the related problem of assigning, for each node in the network, a neighbor to back up its data in case of eventual node failure. Here, a key goal is to minimize the maximum *load*, defined as the number of nodes assigned to a single node. We present an efficient decentralized low-energy algorithm that finds a neighbor assignment whose maximum load is at most a polylog(n) factor bigger that the optimum.

1 Introduction

For networks of small computers, energy management and conservation is often a major concern. When these networks communicate wirelessly, usage of the radio transceiver to send or listen for messages is often one of the dominant causes of energy usage. Moreover, this has tended to be increasingly true as the devices have gotten smaller; see, for example, [13, 3, 10]. Motivated by these considerations, Chang et al. [4] introduced a theoretical model of distributed computation in which each send or listen operation costs one unit of energy, but local computation is free. Over a sequence of discrete timesteps, nodes choose whether

to sleep, listen, or send a message of $O(\log n)$ bits. A listening node successfully receives a message only when exactly one of its neighbors has chosen to send in that timestep; otherwise it receives no input.

It is not uncommon for research on sensor networks to make assumptions about the topology of the network, such as assuming the network is defined by a unit disk graph, or that each node is aware of its location using GPS. However, we will be interested in the more general setting where we make almost no assumptions about the network topology. We will assume that communication takes place via radio broadcasts, and that there is an arbitrary and unknown undirected graph G whose edges indicate which pairs of nodes are capable of hearing each other's broadcasts. We will, however, assume that each node is initialized with shared parameters n and Δ, which are upper bounds on, respectively, the total number of nodes, and the maximum degree of any node. By designing algorithms to operate without pre-conditions on, or foreknowledge of, the network topology, we potentially broaden the possible applications of our algorithms, and, by extension, of sensor networks. For instance, we can imagine a network of small sensors scattered rather haphazardly from an airplane passing over hazardous terrain; the sensors that survive their landing are unlikely to be placed predictably or uniformly.

In this model, [4] presented a polylog-energy, polynomial-time algorithm for the problem of one-to-all broadcast. A later paper, by Chang, Dani, Hayes, and Pettie [5], gave a sub-polynomial ($n^{o(1)}$) energy, polynomial-time algorithm for the related problem of breadth-first search.

In the present work, we will be concerned with another fundamental problem of graph theory, namely to find large sets of pairwise disjoint edges, or *matchings*. The problem of finding large matchings has been thoroughly studied in a wide variety of computational models dating back more than a century, to König [11]. For a fairly comprehensive review of past results, we recommend Duan and Pettie [8, Section 1].

The main goal of the present work is to present a polylog-energy, polynomial-time distributed algorithm that computes a maximal matching in the network graph. The term maximal here indicates that the matching intersects every edge of the graph, and therefore cannot be augmented without first removing edges. It is well-known that the cardinality of any maximal matching is within a factor two of the largest (or maximum) matching.

▶ **Theorem 1.** *Let G be any graph on at most n vertices, of maximum degree at most Δ. Then Algorithm 1 always terminates in $O(\Delta \log n)$ timesteps, at which point each node knows its partner in a matching, M. Furthermore,*

$$\mathbb{P}\left(M \text{ is a maximal matching and every node used energy } \leq 20C \log^2 n\right) \geq 1 - \frac{1}{n^2}.$$

Observe that the per-node energy use is polylog(n), which obviously cannot be improved by more than a polylog factor. Moreover, the time complexity bound, $O(\Delta \log n)$, is also nearly optimal, when one considers that G could contain a clique of size Δ, in which case, in order for all the nodes in that clique to get even one chance to send a message and have it received by the other nodes in the clique, there must be at least Δ timesteps, since our model does not allow a node the possibility to receive two or more messages in a single round. To put this another way, when Δ is small, a high degree of parallelism is possible, which our algorithm exploits; but, when Δ is large, there exist graphs for which this parallelism is impossible.

1.1 Application: Neighbor Assignment

One possible motivation for finding large matchings, apart from their intrinsic mathematical interest, comes from the desire to **back up data in case of node failures**. Suppose we had a perfect matching (that is, one whose edges contain every node) on the n nodes of our network. Then the matching could be viewed as pairing each node with a neighboring node that could serve as its backup device. This would ensure that each device has a load of one node to back up, and that each node is directly adjacent to its backup device.

Since perfect matchings are not always available, we consider a more general scheme, in which each node is assigned one of its neighbors to be its backup device, but we allow for loads greater than one. Such a function can be visualized as a directed graph, with a directed edge from each node to its backup device. In this case, each node has out-degree 1, and load equal to its in-degree. We would like to minimize the maximum load over all vertices.

In Section 6, we will show that, if one is willing to accept a maximum load that is $O(\log n)$ times the optimum, this problem can be simply reduced to the maximal matching problem. In light of our main result, this means that, if there exists a neighbor assignment with $\mathrm{polylog}(n)$ maximum load, then we can find one on a radio network, while using only $\mathrm{polylog}(n)$ energy.

1.2 Techniques

Our matching algorithm can be thought of as a distributed and low-energy version of the following greedy, centralized algorithm. Randomly shuffle the m edges. Then, processing the edges in order, accept each edge that is disjoint from all previous edges. Note that this always results in a maximal matching.

To make this into a distributed algorithm, we make each node, in parallel, try to establish contact with one of its unmatched neighbors to form an edge. Since a node can only receive a message successfully if exactly one of its neighbors is sending, we limit the probability for each node to participate in a given round, by setting a participation rate that is, with high probability, at most the inverse of the maximum degree of the residual graph induced by the unmatched nodes. It turns out that this can be accomplished using a set schedule, where the participation rate is a function of the amount of elapsed time.

The main technical obstacle in the analysis is proving that the maximum degree of the graph decreases according to schedule (or faster). This is achieved by noting that, if not, the first vertex to have its degree exceed the schedule would have to have been failed to be paired by our algorithm, despite going through a long sequence of consecutive rounds in which its chance to be paired was relatively high.

1.3 Related Work

Multi-hop radio network models have a long history, going back at least to work in the early 1990's by Bar-Yehuda, Goldreich, Itai [1, 2] among others. The particular model of energy-aware radio computation we are using was introduced by Chang et al. [4].

A recent result by Chatterjee, Gmyr, and Pandurangan [6] considered the closely related problem of Maximal Independent Set in another model, called the "Sleeping model." Although it has some interesting similarities to our work, there are several important differences. Firstly, we note that although matchings of G are nothing more than independent sets on the line graph of G, in distributed computing, we cannot just convert an algorithm designed to run on the line graph of G into an algorithm to run on G. Secondly, we note that the Sleeping model is based on the CONGEST model, and so, when a node is awake, it is allowed to send

a different message to each of its neighbors at a unit cost. By contrast, in our model, one node can only send one message in a timestep, and it may collide with messages sent by other nodes.

Moscibroda and Wattenhofer [12] considered the problem of finding a Maximal Independent Set in a radio network. Their work also has some interesting similarities to ours, although they are assuming a unit-disk topology, and listening for messages is free in their model. On the other hand, their algorithm works even when the nodes wake up asynchronously at the start of the algorithm.

2   Preliminaries

2.1   Matchings

A *matching* is a subset of the edges of a graph G, such that no two of the edges share an endpoint. We say a matching is *maximum* if it has at least as many edges as any other matching for G. We say a matching is *maximal* if it is not contained in a larger matching for G. Equivalently, a matching is maximal if every edge of G shares at least one endpoint with an edge from the matching.

For $\alpha > 1$, we say a matching is α-approximately maximum if its cardinality is at least $1/\alpha$ times the cardinality of a maximum matching. It is an immediate consequence of the definitions that any maximal matching is 2-approximately maximum.

2.2   Low-Energy Radio Networks

We work in the Radio Network model, where we have a communication network on an underlying graph G. Each node in G is an identical processor equipped with a transmitter and receiver to communicate with other nodes. There is an edge between nodes u and v in the graph if u and v are within transmission range of each other. Time is divided into discrete timesteps. In each time step a processor can choose to do one of three actions: transmit, listen, or sleep. A message travels from a node u to a neighbor v of u at time t if

- u decides to transmit at time t,
- v decides to listen at time t and
- No other neighbor of v decides to transmit at time t.

Thus when a node u decides to send a message, that message is heard by **all** of its neighbors who happen to be listening, and for whom none of their other neighbors are sending.

So what happens if node v happens to listen and more than one of its neighbors is sending a message? There are several different models for this situation. In the most permissive of these, the LOCAL model, v receives all the messages sent by its neighbors. As already specified, we are not working in this model. A more restrictive model is the Collision Detection model (CD) where, when a listening node does not receive a message, it can can tell the difference between silence (no neighbors sending) and a collision (more than one neighbor sending). Our results hold in the even more restrictive "No Collision Detection" model, where a listening node receives a message when exactly one of its neighbors is sending, but it cannot distinguish between silence and colliding messages.

What about message sizes? The LOCAL model allows nodes to send messages of arbitrary size in a single timestep. In our work we assume that the messages sent in a single timestep are of size that is logarithmic in the size of the network. That is, when there are n nodes, the message sizes are $O(\log n)$.

We measure the cost of our algorithms in terms of their energy usage. We assume that a node incurs a cost of 1 energy unit each time that it decides to send or listen. When the node is sleeping there is no energy cost. We also assume that local computation is free. The goal of energy aware computation is to design algorithms where the nodes can schedule sleep and communication times so that the energy expenditure is small, ideally $\text{polylog}(n)$, without compromising the time complexity too much, i.e., the running time remains $\text{poly}(n)$.

3 Notation

3.1 The network

As mentioned earlier, $G = (V, E)$ is the graph defining our radio network. We denote $n = |V|$, and refer to the nodes as "processors." Although the processors are identical, and run identical code, we will assume each node has a unique ID that it knows and uses as its "name" in communication. We make the standard observation that, if each node were to generate an independently random string of $C \log n$ bits as its ID, the probability that all n nodes have distinct IDs is at least $1 - 1/n^{C-1}$, which can be made overwhelmingly likely.

When we present our pseudocode, it will be written from the perspective of a single processor. However, most of our analysis will be written from the "global" perspective of the entire graph.

3.2 Measuring time

To begin with, we define two units of time that will be used throughout the paper. The smaller unit of time is called a *timestep*, and refers to the basic time unit of our radio network model: in each timestep the nodes that choose to transmit are allowed to send a single message.

The larger unit of time, is called a *round*, and consists of three timesteps of the form $3t - 2, 3t - 1, 3t$, where $1 \leq t \leq t_{\max}$ is the round number. As shall be seen, rounds have the property that at the end of each round, the aggregate state of the network *encodes a matching*. More precisely, each node has a variable, `partner`, and at the end of each round, this variable is either the ID of one of its neighbor nodes, or has the value `null`; moreover, whenever, at the end of a round, `partner`$(v) = w \neq$ `null`, we also have `partner`$(w) = v$.

3.3 The Evolving Matching

For $t \geq 0$, we denote by $M(t)$ the matching encoded by the network at the end of round t; this is a random variable whose value is always a pairwise disjoint set of edges of the graph. As discussed earlier, $M(t)$ is well defined because, at the end of every round, all vertices have a mutually consistent view of whom they are paired to.

We mention that our matching will be non-decreasing over time, that is, for all $t < t'$, $M(t) \subseteq M(t')$ with probability one. Next we define a few useful notations, all defined in terms of G and $M(t)$.

By $G(t)$, we denote the *residual graph at the end of round t*; that is, the graph induced by all unmatched vertices of G. Thus $G(t) = (V(t), E(t))$, where $V(t) = V \setminus \bigcup M(t)$ and $E(t) = E \cap \binom{V(t)}{2}$. Similarly, for a vertex v, we will refer to its residual neighbor set, $N(v, t) = N(v) \cap V(t)$, and its residual degree, $d(v, t) = |N(v, t)|$. We will denote the maximum degree in the residual graph by $\Delta(t) = \max_{v \in V(t)} d(v, t)$, taking this value to be zero if $V(t)$ is empty.

4   Maximal Matching Algorithm

The basic idea of our algorithm is, starting with the empty matching, to greedily add disjoint edges until a maximal matching is achieved. The challenge is to keep each node's energy cost low. We achieve this by having nodes wake up at random times, and try to recruit one of their neighbors to pair with them. If this succeeds, and is not hampered by additional, redundant, neighbors that also happen to wake up, then an edge is added to the matching.

To ensure that both endpoints of the edge agree about who they are paired with, the nodes execute a three-step "handshake" protocol, with the property that, if it succeeds, both nodes know that the other node has only been in communication with them, and was not, for instance, trying to form an edge with another, different, endpoint.

To keep the energy costs low, it is essential that nodes wake up with approximately the correct frequency. If the rate is too high (significantly greater than the inverse of the degree), too many nodes will wake up at once, causing collisions. Even if we get around these collisions by some device, having too many nodes wake up at once seems likely to lead to excessive energy consumption, since at most one neighbor of a node can get a message through in a single round.

If the rate is too low (significantly less than the inverse of the degree), then too few nodes will wake up at once, again leading to an excessive waste of energy, since a node whose neighbors are all asleep cannot get anywhere.

We define the "participation rate," $r(t)$ by the function

$$r(t) = \frac{3C \log(n)}{4C \log(n) + t_{\max} - t}$$

where $t_{\max} = C\Delta \log(n)$. This function gives a schedule for gradually raising the participation rate from $\Theta(1/\Delta)$ up to $\Theta(1)$. The constant C will be specified in the proof of Theorem 1.

■ **Algorithm 1** Main Algorithm: A Low-Energy Distributed implementation of Greedy Maximal Matching in a Radio Network.

 1: $t \leftarrow 1$
 2: `partner`← `null`
 3: **while** `partner`= `null` and $t \leq t_{\max}$ **do**
 4:     Sample x uniformly from $[0, 1]$.
 5:     **if** $x \leq r(t)/2$ **then**
 6:         Do RECRUIT_PROTOCOL this round.
 7:     **else if** $r(t)/2 < x \leq r(t)$ **then**
 8:         Do ACCEPT_PROTOCOL this round.
 9:     **else**
10:         Sleep this round.
11:     **end if**
12:     $t \leftarrow t + 1$
13: **end while**
14: Sleep for the remaining $t_{\max} - t$ rounds.

◾ **Algorithm 2** RECRUIT_PROTOCOL: Try to form an edge as initial sender.

1: At timestep 1, Send my_ID ▷ "My name is my_ID and I am available"
2: At timestep 2, Listen
3: **if** message received **then**
4:   Interpret the message as an ordered pair of integers (x, y)
5:   **if** $x = my\_ID$ **then**   ▷ Match found
6:    partner$\leftarrow y$
7:    At timestep 3, send (x, y) ▷ "x and y are paired"
8:   **end if**
9: **else**
10:   Sleep for timestep 3.
11: **end if**

◾ **Algorithm 3** ACCEPT_PROTOCOL: Try to form an edge as initial listener.

1: At timestep 1, Listen
2: **if** message received **then**
3:   Interpret the message as an integer x
4:   At timestep 2, Send (x, my_ID) ▷ "Hello, lets match up, x and my_ID"
5:   At timestep 3, Listen
6:   **if** message received **then**
7:    Interpret the message as an ordered pair of integers (x, y)
8:    **if** $(y == my\_ID)$ **then**   ▷ x and y are matched
9:     partner$\leftarrow x$
10:    **end if**
11:   **end if**
12: **end if**
13: Sleep for any timesteps remaining in the round.

---

5   Maximal Matching Analysis

▶ **Lemma 2.** *With probability one, at the end of every round $t \geq 0$, the* partner *variables of the n nodes encode a well-defined matching $M(t)$.*

Proof. Initially, all the vertices are unmatched, with null partners, so $M(0) = \emptyset$. Later, we observe that the only circumstances under which the partner variables have their values reassigned is when a vertex v has chosen to participate in that round as recruiter, a neighboring vertex w has chosen to participate in that round as accepter, and furthermore, both v and w receive a message each time they Listen during their respective protocols. Since a message is received if and only if exactly one neighbor Sends in that timestep, the messages v receives must come from w, and vice-versa. Therefore v stores the id of w in its partner variable, and vice-versa.

Furthermore, since v and w would not have participated in round t unless their partner variables were both null beforehand, we know by induction that no other vertices have v or w as their partners. Since this applies for all vertices and all rounds, the pairing is one-to-one, as desired. ◀

▶ **Lemma 3.** *Let $t \geq 1$, let $\{v, w\} \in E$, and let $X_{v,w,t}$ be the indicator random variable for the event that v and w get matched to each other in round t. Then*

$$\mathbb{E}\left(X_{v,w,t} \mid M(t-1) \right) \geq \frac{r(t)^2}{2} \left(1 - r(t) \right)^{\Delta(t-1)-1} \mathbf{1}(v, w \in V(t-1)).$$

Proof. In order for an edge to form between v and w in round t, it is necessary and sufficient for the following four events all to occur:

$E_0 = \{v, w \in V(t-1)\}$,

$E_1 = \{v, w \text{ both participate in round } t\}$,

$E_2 = \{\text{exactly one of } v, w \text{ participates as recruiter, the other as accepter, in round } t\}$,

$E_3 = \{E_2, \text{ and } v \text{ and } w \text{ receive each others messages without any collision}\}$.

Note that $E_3 \subset E_2 \subset E_1 \subset E_0$. For $0 \leq i \leq 3$, let $X_i = \mathbf{1}(E_i)$. We now compute expectations, conditioned on the matching at the end of the previous round. We will prove, below, that

$$\mathbb{E}\left(X_1 \mid M(t-1) \right) = r(t)^2 X_0, \tag{1}$$

$$\mathbb{E}\left(X_2 \mid X_1, M(t-1) \right) = \frac{1}{2} X_1, \tag{2}$$

$$\mathbb{E}\left(X_3 \mid X_2, X_1, M(t-1) \right) \geq \left(1 - r(t) \right)^{\Delta(t-1)-1} X_2. \tag{3}$$

It follows by the Law of Total Probability that

$$\mathbb{E}\left(X_3 \mid M(t-1) \right) \geq \frac{r(t)^2}{2} \left(1 - r(t) \right)^{\Delta(t-1)-1} X_0,$$

which is equivalent to the statement of the lemma, noting that $X_3 = X_{v,w,t}$ and $X_0 = \mathbf{1}(v, w \in V(t-1))$.

Now, (1) and (2) follow immediately from the definitions of E_1, E_2, and the fact that every vertex in $V(t-1)$ has probability $r(t)/2$ to participate as recruiter in round t, and the same probability to participate as accepter. To establish (3), we first note that, assuming E_2 occurs, for E_3 to occur it is sufficient that no other neighbor of w decides to participate in round t in the same role as v, and no other neighbor of v decides to participate in the same role as w. Since each vertex makes its participation decision independently, this probability equals

$$\left(1 - r(t) \right)^A \left(1 - \frac{r(t)}{2} \right)^B \geq \left(1 - r(t) \right)^{A+B/2},$$

where $A = |N(v, t-1) \cap N(w, t-1)|$ and $B = |N(v, t-1) \oplus N(w, t-1) \setminus \{v, w\}|$. But since $2A + B = |N(v, t-1) \setminus \{w\}| + |N(w, t-1) \setminus \{v\}| \leq 2(\Delta(t-1) - 1)$, this establishes (3), completing the proof.   ◀

We now introduce a random variable that will be used crucially in the remainder of our analysis.

▶ **Definition 4.** *Let v be a vertex, and $1 \leq t \leq t_{\max}$. $Z(v, t)$ denotes the indicator random variable for the event $\{d(v, t) \geq 1/(2r(t)) \text{ and } \Delta(t-1) \leq 1/r(t)\}$.*

This definition is motivated by the observation that, if t were minimal such that there exists a v such that $d(v, t) \geq 1/(2r(t))$, then $Z(v, t)$ would have to have value 1 for many immediately preceding rounds, which we can prove is highly unlikely.

▶ **Lemma 5.**

$$\mathbb{E}\left(Z(v,t) \mid M(t-1)\right) \leq 1 - \frac{r(t)}{4e^2}.$$

Proof. Recall that, by definition, $Z(v,t) = 1$ if and only if: $v \in V(t)$ and $d(v,t) \geq 1/2r(t)$ and $\Delta(t-1) \leq 1/r(t)$. Now, since the degrees at time $t-1$ are determined by $M(t-1)$, and there is nothing to prove when the conditional information implies $Z(v,t)$ is identically zero, we may assume $d(v,t-1) \geq 1/2r(t)$ and $\max_w d(w,t-1) \leq 1/r(t)$.

Also, note that $v \notin V(t)$ will occur if and only if $X_{v,w,t} = 1$ for some $w \in N(v,t-1)$. Since these events are disjoint, we may sum their probabilities, obtaining

$$\mathbb{E}\left(Z(v,t) \mid M(t-1)\right) \leq 1 - \sum_w \mathbb{E}\left(X_{v,w,t} \mid M(t-1)\right)$$

$$\leq 1 - d(v,t-1)\frac{r(t)^2}{2}\left(1 - r(t)\right)^{\Delta(t-1)-1} \qquad \text{by Lemma 3.}$$

$$\leq 1 - d(v,t-1)\frac{r(t)^2}{2}\left(1 - r(t)\right)^{1/r(t)}$$

Additionally, since $Z(v,t) = 0$ unless $d(v,t) \geq 1/2r(t)$, we have

$$\mathbb{E}\left(Z(v,t) \mid M(t-1)\right) \leq 1 - \frac{1}{2r(t)}\frac{r(t)^2}{2}\left(1 - r(t)\right)^{1/r(t)}$$

$$\leq 1 - \frac{r(t)}{4}\left(1 - r(t)\right)^{1/r(t)}$$

$$\leq 1 - \frac{r(t)}{4e^2}.$$

Here the last inequality follows because on the interval $[0, 3/4]$ we have $1 - x > e^{-2x}$, and $r(t) \leq 3/4$ for all t. ◀

We are now ready to present the proof of Theorem 1.

Proof of Theorem 1. We focus on the probability that M is maximal. If not, then there exists a vertex whose degree is positive after round $t_{\max}$. Since $r(t_{\max}) = 3/4$, it follows that the set

$$S = \left\{(v,t)\colon v \in V, 0 < t \leq t_{\max}, d(v,t) \geq \frac{3}{4r(t)}\right\}$$

is non-empty. Let t be the smallest time such that there exists a vertex v with $(v,t) \in S$.

Let I be the set of all times $t' \leq t$ such that $2r(t)/3 < r(t') \leq r(t)$. By our assumption that t is minimal, we have

$$d(w, t'-1) < \frac{3}{4r(t'-1)} < \frac{1}{r(t')}$$

for all vertices $w \in V$ and times $t' \in I$. Hence $\Delta(t'-1) < 1/r(t')$ for all $t' \in I$. On the other hand, since $d(v,t')$ is a non-increasing function of t', we also have

$$d(v,t') \geq d(v,t) \geq \frac{3}{4r(t)} > \frac{1}{2r(t')},$$

for all $t' \in I$, where the last inequality is by the definition of I. It follows that

for all $t' \in I$, $Z(v,t') = 1$.

Let $\mathcal{E}_{v,t}$ be the above event, *i.e.,* that $\prod_{t' \in I} Z(v, t') = 1$. We want to compute the probability of $\mathcal{E}_{v,t}$. Let t_0 be the smallest time in I. Then

$$\mathbb{P}\left(\mathcal{E}_{v,t}\right) = \mathbb{E}\left(\prod_{t'=t_0}^{t} Z(v, t')\right)$$

$$= \mathbb{E}\left(\mathbb{E}\left(\prod_{t'=t_0}^{t} Z(v, t') \,\middle|\, M(t-1)\right)\right) \qquad \text{by the Law of Total Expectation}$$

$$= \mathbb{E}\left(\left(\prod_{t'=t_0}^{t-1} Z(v, t')\right) \mathbb{E}\left(Z(v, t) \mid M(t-1)\right)\right) \qquad (*)$$

$$\leq \mathbb{E}\left(\prod_{t'=t_0}^{t-1} Z(v, t')\right)\left(1 - \frac{r(t)}{4e^2}\right) \qquad \text{by Lemma 5}$$

Line (*) follows since $M(t-1)$ determines $Z(v, t_0), \ldots, Z(v, t-1)$. By repeating the steps from using the Law of Total Expectation to applying Lemma 5 on the remaining expectation of a product, we see by induction that

$$\mathbb{P}\left(\mathcal{E}_{v,t}\right) \leq \prod_{t' \in I}\left(1 - \frac{r(t')}{4e^2}\right)$$

$$\leq \exp\left(-\frac{1}{4e^2}\sum_{t' \in I} r(t')\right) \qquad \text{since for all } x, \, 1 - x \leq e^{-x}$$

To get a handle on the expression on the right hand side, we need a lower bound on the sum of the participation rates. Let $t^* \in \mathbb{R}$ be such that $r(t^*) = \frac{2}{3}r(t)$. Then $t_0 - 1 < t^* \leq t_0$, and we have

$$\sum_{t'=t_0}^{t} r(t') \geq \int_{t_0-1}^{t} r(t') \mathrm{d}t' \qquad \text{upper Riemann sum}$$

$$\geq \int_{t^*}^{t} r(t') \mathrm{d}t'$$

$$\geq \int_{t^*}^{t} \frac{3C \log n}{4C \log n + t_{\max} - t'} \mathrm{d}t'$$

$$= 3C \log n [\log(4C \log n + t_{\max} - t^*) - \log(4C \log n + t_{\max} - t)]$$

$$= 3C \log n \log\left(\frac{r(t)}{r(t^*)}\right)$$

$$= 3C \log n \log(3/2)$$

Plugging this back into the probability calculation,

$$\mathbb{P}\left(\mathcal{E}_{v,t}\right) \leq \exp\left(-\frac{1}{4e^2}\sum_{t' \in I} r(t')\right)$$

$$\leq \exp\left(-\frac{3C \log n \log(3/2)}{4e^2}\right)$$

$$\leq \frac{1}{n^{4+\epsilon}}.$$

where the last inequality holds for suitably large values of C, *e.g.,* when $C = 100$.

Taking a union bound over the $nt_{\max} = O(n^2 \log n)$ events $\mathcal{E}_{v,t}$ completes the proof of correctness.

The upper bound on energy use comes from a simple analysis of the number of rounds each vertex participates in. Clearly, the energy use is at most 3 times the number of rounds the vertex participates in, which is at most the number of heads that would be flipped in $t_{\max}$ independent coin flips, with probabilities of heads $r(1), r(2), \ldots, r(t_{\max})$. Hence the expected energy use is at most the sum, which we calculated above as approximately $3 \cdot 3C \log n \log \left(\frac{r(t_{\max})}{r(0)} \right)$, which is at most $9C(\log n)^2 = O(\log^2 n)$. Chernoff's bound, together with with a union bound over the n vertices, implies the high-probability upper bound on expected energy cost. ◄

6 Neighbor Assignment Functions

Motivated by the problem of assigning nodes to backup data from their neighbors in a sensor network, we introduce the following definition. As we shall see later, it is extremely closely connected to the established concept of matching covering number.

▶ **Definition 6.** *Given graph $G = (V, E)$, a* neighbor assignment function *(NAF) is a function $f : V \to V$ such that for all $v \in V$, $\{v, f(v)\} \in E$. Equivalently, we may think of this as an oriented subgraph of G, in which each vertex has out-degree 1. The* load *of the assignment is the maximum in-degree of this digraph. Equivalently, load is $\max_{v \in V} |f^{-1}(v)|$. The* minimum NAF load *of G is the minimum load among all NAFs for G.*

Note. In the case when G is bipartite, NAFs are also known as "semi-matchings." (See, for example, [9, 7].) However, since we are particularly concerned with the non-bipartite case, we preferred to introduce a different term.

In the context of backing up data, we think of the assigned node $f(v)$ as the node who will store a backup copy of v's data. Our goal for this section is to find a NAF whose load is small. In the energy-aware radio network setting, we also want to ensure that the per-node energy use is small.

Our next result establishes a close connection between the load of the best NAF for a graph and the minimum number of matchings needed to cover all of its vertices.

▶ **Definition 7.** *The* matching cover number *of a graph G, denoted $\mathrm{mc}(G)$, is the minimum integer k such that there exists a set of k matchings of G, whose union contains every vertex of G.*

▶ **Theorem 8.** *For every graph G, the minimum NAF-load of G equals the matching cover number of G, unless the NAF-load of G equals 1. If the NAF-load of G equals 1, the matching cover number of G can be 1 or 2.*

Proof. Suppose $V = V(G)$ is covered by the union of matchings $M_1, \ldots, M_L$. Then assigning each vertex v to its partner in the first matching that contains v is an NAF with maximum load at most L. This establishes that the NAF-load is always at most the matching covering number.

Before we begin the proof for the reverse implication, we make the following general observation about digraphs with out-degree 1. By considering the unique walk obtained by starting at any vertex v, and repeatedly following the edge $\{v, f(v)\}$, we can see that each weakly connected component consists of one oriented cycle (of length ≥ 2), together with one or more "tributary" trees, each rooted at a node of this cycle, and oriented towards that root. See Figure 1.
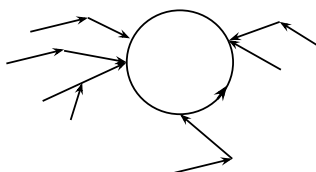
Figure 1 Sketch of the digraph of a NAF with only one component. Each component can be seen as a directed cycle, along with zero or more "tributary trees."
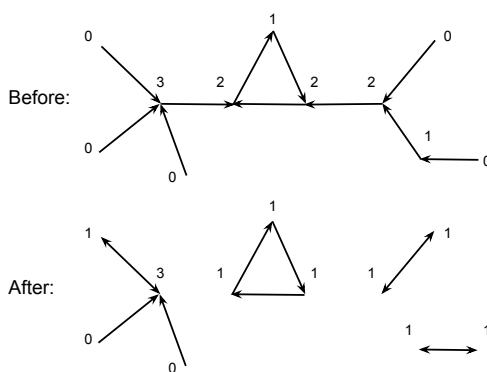
Figure 2 Example of the conversion of a given NAF by decreasing the number of load-zero nodes. The number by each node indicates its load. Note that, after the conversion, the maximum load did not increase, and the connected components of the NAF are now all directed cycles and/or stars with one bi-directed edge. (The components of size 2 are both.)

Now, if f has any leaf, that is, a node, v, whose load is zero, we can obtain a new NAF by reassigning $f(v)$ to point back to v. This increases the load at v to 1, decreases the load by 1 at $f(f(v))$, and does not change any other vertex loads. Repeated application of this rule to all leaves in turn, eventually leads to a NAF whose components are all either (a) directed cycles, which do not have any leaves, or (b) stars with one bi-directed edge. See Figure 2. In case (b), the component consists of one node, r, of in-degree $i \leq L$, i nodes, $x_1, \ldots, x_i$, each with an edge directed to r, and one edge from r to x_1.

It is easy to see that, for a directed cycle, whose edges are $e_1, \ldots, e_\ell$, a single matching consisting of the even edges, $e_2, e_4, \ldots,$, will cover all the vertices if ℓ is even, and all but one vertex if ℓ is odd. Therefore, one matching covers the component if ℓ is even, and two if ℓ is odd.

For the star with bi-directed edge, the maximum load equals the degree, d, of the center vertex. And a matching cover consists of the d single edges that make up the star.

In this way, we can build up our matching cover component by component, noting that if every component has a matching cover of size at most k, then so does the entire graph. Since the only case when our matching cover was smaller than the maximum load for the component was when $L = 1$, the proof is complete.   ◀

Wang, Song, and Yuan [14] have given an $O(n^3)$-time centralized algorithm for finding the minimum number of matchings needed to cover a graph. In light of Theorem 8, their result implies an $O(n^3)$ algorithm for finding the minimum-load NAF for any graph.

In the distributed and low-energy setting, it is unlikely that we can achieve such an ambitious goal. For instance, a node cannot determine its exact degree without receiving that many messages successfully, which may require linear energy. Instead, we aim for the less ambitious goal of finding a NAF whose maximum load is well within our energy budget. Our next result shows that this is possible, assuming one exists.

■ **Algorithm 4** Low-Energy Distributed algorithm to compute a NAF in a Radio Network.

1: Run our Maximal Matching algorithm on G.
2: For each edge $\{u, v\}$ in the matching, mark u, v as assigned, and assign them to each other.
3: **for** $i \leftarrow 1$ to k **do**
4:     Run the maximal matching algorithm on G, modified so that only unassigned nodes are allowed to recruit, and only assigned nodes are allowed to accept.
5:     For each edge $\{u, v\}$ in the matching, mark u, v as assigned, and (re-)assign them to each other.
6: **end for**

▶ **Theorem 9.** *Suppose G has a NAF with maximum load L. Then Algorithm 4, run with parameter $k = O(L \log n)$, will, with high probability, output a NAF with maximum load $O(L \log n)$. Its per-vertex energy usage will be $O(L \operatorname{polylog} n)$.*

Proof. Since G has a NAF with maximum load L, by Theorem 8, it has a vertex cover by L matchings. This implies that the maximum matching covers at least N/L vertices. Hence every maximal matching covers at least $N/2L$ vertices. So the first call to the maximal matching algorithm will assign neighbors to at least $N/2L$ vertices.

In subsequent rounds, the modification to the maximal matching algorithm has the effect of making it run on the bipartite graph where the bipartition is into the assigned and unassigned vertices. Now, consider the vertex cover by the L matchings in G. By pigeonhole principle, at least one of these matchings, M, must cover at least a $1/L$ fraction of the unassigned vertices. Since the first matching was maximal, no edges in G have both endpoints unassigned; therefore, M is a matching within the bipartite graph being fed into our maximal matchings algorithm. Therefore, the maximal matching that is found must cover at least a $1/2L$ fraction of the unassigned vertices. It follows that after k iterations, at most

$$\left(1 - \frac{1}{2L}\right)^k n \leq e^{-k/2L} n$$

nodes will remain unassigned.

Setting $k = 2L \log(n)$ gives a good chance that no nodes remain unassigned, and so the algorithm produces a NAF. Clearly, since the digraph for the NAF is contained within the union of k matchings, this is an upper bound on the maximum load. Since each run of the maximal matching algorithm succeeds with probability $1 - O(1/n^2)$, a union bound establishes the high-probability bound.

More generally, if a partial NAF exists, that covers a $(1 - \varepsilon)$ fraction of the nodes, with maximum load L, and we run Algorithm 4 with $k = 2L \log(1/\delta)$, it should, with high probability, produce a partial NAF that covers at least $(1 - \varepsilon)(1 - \delta)n$ nodes, and has maximum load k ◀

References

1   Reuven Bar-Yehuda, Oded Goldreich, and Alon Itai. Efficient emulation of single-hop radio network with collision detection on multi-hop radio network with no collision detection. *Distributed Computing*, 5(2):67–71, 1991.

2   Reuven Bar-Yehuda, Oded Goldreich, and Alon Itai. On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization. *Journal of Computer and System Sciences*, 45(1):104–126, 1992.

3   Matthew Barnes, Chris Conway, James Mathews, and DK Arvind. Ens: An energy harvesting wireless sensor network platform. In *2010 Fifth International Conference on Systems and Networks Communications*, pages 83–87. IEEE, 2010.

4   Yi-Jun Chang, Varsha Dani, Thomas P Hayes, Qizheng He, Wenzheng Li, and Seth Pettie. The energy complexity of broadcast. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, pages 95–104, 2018.

5   Yi-Jun Chang, Varsha Dani, Thomas P Hayes, and Seth Pettie. The energy complexity of BFS in radio networks. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, pages 273–282, 2020.

6   Soumyottam Chatterjee, Robert Gmyr, and Gopal Pandurangan. Sleeping is efficient: Mis in $o(1)$-rounds node-averaged awake complexity. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, pages 99–108, 2020.

7   Andrzej Czygrinow, Michał Hanćkowiak, Edyta Szymańska, and Wojciech Wawrzyniak. On the distributed complexity of the semi-matching problem. *Journal of Computer and System Sciences*, 82(8):1251–1267, 2016.

8   Ran Duan and Seth Pettie. Linear-time approximation for maximum weight matching. *J. ACM*, 61(1):1–23, 2014.

9   Nicholas JA Harvey, Richard E Ladner, László Lovász, and Tami Tamir. Semi-matchings for bipartite graphs and load balancing. *Journal of Algorithms*, 59(1):53–78, 2006.

10   Wendi R. Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, pages 10–pp. IEEE, 2000.

11   Dénes König. Über graphen und ihre anwendung auf determinantentheorie und mengenlehre. *Mathematische Annalen*, 77(4):453–465, 1916.

12   Thomas Moscibroda and Roger Wattenhofer. Maximal independent sets in radio networks. In *Proceedings of the 24th Annual ACM Symposium on Principles of Distributed Computing*, pages 148–157, 2005.

13   Joseph Polastre, Robert Szewczyk, and David Culler. Telos: Enabling ultra-low power wireless research. In *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks*, pages 364–369. IEEE, 2005.

14   Xiumei Wang, Xiaoxin Song, and Jinjiang Yuan. On matching cover of graphs. *Mathematical Programming*, 147(1):499–518, 2014.