Ranking and Sorting in Unreliable Single Hop Radio Network*

Marcin Kik
Marcin.Kik@pwr.wroc.pl

Institute of Mathematics and Computer Science,
Wrocław University of Technology
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

Abstract. We propose simple and efficient sorting algorithm for unreliable single hop radio network. (In such network each listening station receives transmitted message with some probability $p < 1$.) We also propose a method of periodic transmission of a sorted sequence that allows for efficient and energetically safe ranking in this sequence.

1 Introduction

We consider the problems of sorting and ranking in unreliable single hop radio network. Such network consists of n stations $s_0, \ldots, s_{n-1}$ communicating with each other by exchanging short radio messages. The stations are synchronized. Time is divided into *slots*. Within a single time slot a single message can be broadcast. During each time slot each station is either listening or sending or idle. If it is sending or listening then it dissipates a unit of energy. We assume that the stations are powered by batteries. Therefore we want to minimize *energetic cost* of the algorithm, i.e. maximum over all stations of non-idle time slots. Each station is in the range of any other station (i.e. a *single hop* network). If two or more stations send messages simultaneously, then a *collision* occurs. In this paper we consider only collision-less algorithms. If during time slot t only one station sends a message and any other station (say s_i) is listening, then s_i receives the message with probability p (*probability of successful reception*). The special case $p = 1$ means *reliable* network. The previously proposed sorting algorithms for this model ([10], [6], [3], [4], [5]) were designed for reliable network. If any transmission failed then the whole output would be devastated. Since radio transmissions are vulnerable to many unpredictable external interferences, we believe that practical algorithms should be robust to occasional losses of received messages. A simple general strategy of increasing the robustness of the algorithm is to make each transmission robust by repeating it many times. If the transmission from single sender to single receiver is repeated r times then the probability of failure is reduced from q to q^r, where $q = 1 - p$. However, the

* This work has been supported by the ICT Programme of the European Union under contract number FP7-215270 (FRONTS)

energetic cost of sending is increased r times. (The receiver may stop listening as soon as it receives the message.) The situation is still worse if there are m receivers, $m > 1$. We should ensure that **all** receivers have received the message with high probability.

Any sorting algorithm consists of $\Omega(n)$ transmissions, and any of those transmissions may have a large number of receivers. It seems that constructing an algorithm with reasonably high probability of success requires a lot of energy. The number of repetitions for each step should be rather overestimated since the failure of any robust step makes all the previous and remaining computations useless. We propose sorting based on simple merge-sort presented in [4]. Because of the asymmetry between sending and receiving energetic costs in that algorithm, the asymptotic expected energetic cost of our robust algorithm is as low as that of the sorting algorithms with asymptotically lower costs (e.g. [1], [10]) with retransmissions of each step, while the low constants and simplicity make it preferable in practical implementations.

By *ranking* we mean the problem of locating the position of some key x in a sorted sequence of keys (i.e. the number of keys in the sequence that are less than x). One of the many applications of efficient sorting and ranking algorithms may be the routing of packets. The routing algorithms for single hop network ([2], [7], [8], [3]) typically consist of some preprocessing reservation phase that allows for subsequent energetically efficient delivery of the packets. Such preprocessing may consist of sorting the addresses of the packets, and ranking by each station its own address and the next address in the sorted sequence (see e.g. [3]). Then the packets are delivered according to the sorted sequence and each station knows the interval of time slots in which it should listen. Note that even the approximation of such interval (its superset) can be useful. The ranking algorithms proposed in this paper find the exact rank by updating its lower and upper bounds (until they meet each other) while listening to the iterated transmissions of the sorted sequence. The quality of these bounds after the first iteration depend solely on the probability p and can be used for approximation of such interval. We also consider the case when the ranking station may start at arbitrary time slot while the sequence is periodically transmitted. Here we propose that the sorted sequence is transmitted in *recursive bisection ordering* (rbo), which is easily computable permutation. In the case of a reliable network we formally prove in Section 3.1 that the energy used by the ranking station is then $O(\lg n)$.

In our algorithms each message contains only a single key of the input sequence.

2 Preliminaries.

We formulate the problem of sorting as follows: Each station s_i initially stores a key in its local variable $\mathsf{key}[s_i]$. The task of each station s_i is to compute the value $\mathsf{idx}[s_i]$ which is the index of $\mathsf{key}[s_i]$ in the sorted sequence of keys. (The indexes are numbered from 0 to $n-1$.)

In Section 3 we consider the problem of *ranking*: The sorted sequence is transmitted periodically (in some fixed ordering π). Each round requires n time slots. During any time slot any station may start the computation of the rank (or its approximation) of some key in the transmitted sequence. (By the *rank* of the *key* in the sequence s we mean the number of elements of s that are less or equal to the *key*).

In this paper "lg" denotes "$\log_2$". For simplicity of description we assume that n (number of keys and stations) is a power of two (i.e. $\lg n$ is integer). By $Pr(\mathcal{E})$ we denote probability of the event $\mathcal{E}$. By $E[X]$ we denote expected value of random variable X. By $|S|$ we denote the size of the set S. Whenever we define a permutation π of $\{0,\ldots,n-1\}$, π^{-1} denotes the permutation reverse to π and we settle that $\pi(NIL) = \pi^{-1}(NIL) = NIL$, where NIL is a special constant distinct from all numbers.

3 Ranking.

Let $n = 2^k$, where k is positive integer. The *generic ranking algorithm* is defined as follows: Let π_k be a permutation of the elements $\{0,\ldots,n-1\}$. Let $b_0,\ldots,b_{n-1}$ be a sorted sequence of keys. The sequence permuted by π_k is transmitted periodically, i.e. for $t \geq 0$, b_i such that $\pi_k(i) = t \bmod n$ is transmitted in time slot t. Let a be a station that wants to compute the rank of $\mathsf{key}[a]$ in the sorted sequence. a can start in arbitrary time slot. It knows permutation π_k and the numbering of time slots. Station a contains variables $\mathsf{minR}[a]$ and $\mathsf{maxR}[a]$ that are updated during successful receptions. Initially $\mathsf{minR}[a] = 0$ and $\mathsf{maxR}[a] = n$. In time slot t (i.e. when $key = b_{\pi_k^{-1}(t \bmod n)}$ is transmitted), a does:

> Let $t' = t \bmod n$. If $\mathsf{minR}[a] \leq \pi_k^{-1}(t') < \mathsf{maxR}[a]$ then a listens. If a received the *key*, then
>
> > if $\mathsf{key}[a] < key$ then a sets $\mathsf{maxR}[a]$ to $\pi_k^{-1}(t')$, otherwise (i.e. if $key \leq \mathsf{key}[a]$) it sets $\mathsf{minR}[a]$ to $\pi_k^{-1}(t') + 1$.

Note the following invariant: The rank of $\mathsf{key}[a]$ is in the interval $[\mathsf{minR}[a], \mathsf{maxR}[a]]$. Thus as soon as $\mathsf{minR}[a] = \mathsf{maxR}[a]$ the exact rank of $\mathsf{key}[a]$ is computed. Station a participates in the algorithm as long as it needs or some limit imposed on time or its *listening* energy is exceeded.

Lemma 1 can be used for estimating the time needed for exact ranking with high probability.

Lemma 1. *Let c be a positive integer. After $c \cdot n$ time slots* $\mathsf{minR}[a] = \mathsf{maxR}[a]$ *with probability at least* $1 - 2 \cdot (1-p)^c$.

Proof. Let r be the exact rank of $\mathsf{key}[a]$. To have $\mathsf{minR}[a] = \mathsf{maxR}[a] = r$ the station needs successful reception of the keys b_{r-1} and b_r. The probability that during the c trials a fails to receive the key is $(1-p)^c$. Thus the probability that a fails to receive from both b_{r-1} and b_r is not greater than $2 \cdot (1-p)^c$. □

Lemma 2 estimates the size of the interval $[\mathsf{minR}[a], \mathsf{maxR}[a]]$ after n time slots.

Lemma 2. *The expected value of* $\Delta = \mathsf{maxR}[a] - \mathsf{minR}[a]$ *after* n *time slots is not greater than* $2/p - 2$.

Proof. Let r be the exact rank of $\mathsf{key}[a]$. In the n time slots all the keys b_i have been transmitted. We may think as follows: each transmission was successful with probability p, and whenever station a actually listened it simply observed this transmission. Let u be minimal integer such that $u = n$ or $r \leq u < n$ and the transmission of b_u was successful. It follows from the construction of the algorithm that a observes this transmission and ends up with $\mathsf{maxR}[a] = u$. Let $X_1 = u - r$. If u had not been limited by n, then $X = u - r + 1$ would have been random variable with geometric distribution: $Pr(X = m) = (1-p)^{m-1} \cdot p$ with the expected value $E[X] = 1/p$. Since $X_1 = \min\{X - 1, n - r\}$, we have $E[X_1] \leq 1/p - 1$. It follows (by symmetry) that, for $X_2 = r - \mathsf{minR}[a]$, $E[X_2] \leq 1/p - 1$. Thus, $E[\Delta] = E[X_1 + X_2] = 2/p - 2$. □

The choice of permutation π_k has great influence on the energy used by a. If π_k is identity, a starts listening in time slot 0 and rank of $\mathsf{key}[a]$ is n, then a is forced to listen in all n time slots. Much better option is to use *bisection ordering* (denoted by bo): first (on level 0) transmit the median x of the sequence, then (on level 1) transmit the two medians of the sub-sequences neighboring to x, and so on. For $n = 2^k$, we define precisely bo_k by selecting upper median whenever we have to choose. There is binary tree of depth $k+1$ corresponding to bisection ordering (see Figure 1). On Figure 1 each argument x is joined by vertical dotted

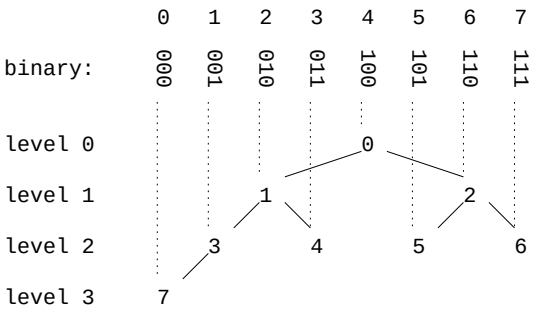

Fig. 1. The tree of bo_3.

line with its corresponding node labeled by $\mathsf{bo}_k(x)$. Note the dependence between binary representation of x and its position in the tree: The level of positive x is determined by the position of its rightmost one and the digits left to this one form the position of x within the level. $x = 0$ is the only argument placed on level k. For $x > 0$, let $\mathsf{irmo}(x) = \min\{j \geq 0 \mid \lfloor x/2^j \rfloor \bmod 2 = 1\}$ (index of rightmost one), and let $\mathsf{irmo}(0) = -1$. Let $\mathsf{lbo}_k(x) = k - 1 - \mathsf{irmo}(x)$ (level of x in bo_k).

Now we can define bo_k as follows: $\mathsf{bo}_k(x) = 2^{\mathsf{lbo}_k(x)} - 1 + \lfloor x/2^{\mathsf{irmo}(x)+1} \rfloor$. (There are $2^{\mathsf{lbo}_k(x)} - 1$ nodes above the level $\mathsf{lbo}_k(x)$ and $\lfloor x/2^{\mathsf{irmo}(x)+1} \rfloor$ is the position of x on the level $\mathsf{lbo}_k(x)$.) The permutation reverse to bo_k can be computed as follows: Let $\mathsf{lev}(y) = \lfloor \lg(y+1) \rfloor$. Then $\mathsf{bo}_k^{-1}(y) = \left(y - \left(2^{\mathsf{lev}(y)} - 1\right)\right) \cdot 2^{k-\mathsf{lev}(y)} + \lfloor 2^{k-\mathsf{lev}(y)-1} \rfloor$. (For $y = 2^k - 1$, the result is zero, and, for $y < 2^k - 1$, the first component of the sum is the position of y within the level $\mathsf{lev}(y)$ multiplied by $2^{k-\mathsf{lev}(y)}$ and the second component settles the rightmost one.)

In reliable networks the ordering of transmissions $\pi_k = \mathsf{bo}_k$ guaranties that if a starts in time slot t such that $t \bmod n = 0$, then a has to listen at most once on each level and therefore uses no more than $k + 1$ units of energy. However, if we let a start its computation in arbitrary time slot, then a may be forced to listen in many time slots. For example, if a starts in time slot $n/2 - 1$ and the rank of $\mathsf{key}[a]$ is n, then a must listen in all $n/2$ time slots on level $k - 1$. On the other hand, forcing a to wait until time slot t such that $t \bmod n = 0$ may cause serious delays. Therefore we propose slightly more "sophisticated" permutation that to a large extent eliminates this problem. The permutation *recursive bisection ordering* (rbo_k) is defined as follows: First we permute the elements according to bisection ordering and then we permute each level (except the first and the last one) according to recursive bisection ordering. The permutations rbo_k and rbo_k^{-1} can be computed by Algorithms 1 and 2, respectively. The permutation π_k can

function $\mathsf{rbo}_k(x)$
begin
  if $x = 0$ **then return** $2^k - 1$;
  $y \leftarrow \mathsf{bo}_k(x)$;
  if $y = 0$ **then return** 0;
  $above \leftarrow 2^{\mathsf{lev}(y)} - 1$;
  return $above + \mathsf{rbo}_{\mathsf{lev}(y)}(y - above)$;
end

Algorithm 1: Computation of $\mathsf{rbo}_k(x)$.

function $\mathsf{rbo}_k^{-1}(y)$
begin
  if $y = 2^k - 1$ **then return** 0;
  if $y = 0$ **then return** $\mathsf{bo}_k^{-1}(0)$;
  $above \leftarrow 2^{\mathsf{lev}(y)} - 1$;
  return $\mathsf{bo}_k^{-1}\left(above + \mathsf{rbo}^{-1}_{\mathsf{lev}(y)}(y - above)\right)$;
end

Algorithm 2: Computation of $\mathsf{rbo}_k^{-1}(y)$.

be imagined as a set of parallel vertical blades cutting of parts of horizontal interval containing the rank of $\mathsf{key}[a]$ as it falls downwards. To appreciate the difference between bo_k and rbo_k see Figure 2. Even if many highest blades of rbo

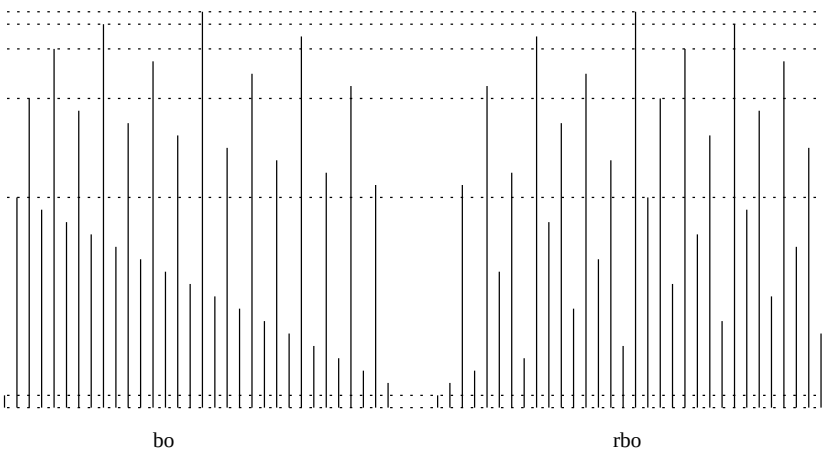

Fig. 2. Permutations bo_5 and rbo_5. (Dotted lines denote borders between levels of bo_5.)

are missing, the remaining ones perform (less exact) bisection.

Next we show that if the station a starts in time slot 0 and we use permutation bo or rbo, then the expected energy used by a during the first iteration is very low.

Lemma 3. *Let π_k be bo_k or rbo_k. Let the station a start in time slot 0. The expected energy used by a during the first $n = 2^k$ time slots is at most $1 + k \cdot (2/p - 1)$.*

Proof. Let $B_l = \langle b_{l,0}, \ldots, b_{l,2^{l+1}-2} \rangle$ be a sequence of keys $\{b_j \mid \mathsf{lbo}_k(j) \leq l\}$ (i.e. the sub-tree of bisection tree from level 0 to l) sorted by j. Station a has to listen to $b_{0,0}$ (the root of bisection tree). Thus the energy used by a on level 0 is 1. For $l \geq 0$, let $r_l = |\{b_{l,i} \mid b_{l,i} \leq \mathsf{key}[a]\}|$ (i.e. the rank of a in the sequence B_l). Let d be maximal integer such that $d = -1$ or $0 \leq d < r_l$ and the transmission of $b_{l,d}$ was successful. Let u be minimal integer such that $u = 2^{l+1} - 1$ or $r_l \leq u < 2^{l+1} - 1$ and the transmission of $b_{l,u}$ was successful. As in the proof of Lemma 2, we can show that the expected value of $u - (d+1)$ is not greater than $2/p - 2$. It follows from the construction of the algorithm, that the station a will not listen to any keys that are before $b_{l,d}$ or after $b_{l,u}$ on the following levels of bisection tree. The sequence B_{l+1} consists of the keys from level $l+1$ on even positions and of the keys from B_l on odd positions. Thus a will have to listen to at most $X_l = u - d$ keys on level $l+1$ and $E[X_l] \leq 2/p - 1$. □

3.1 Ranking with rbo in a reliable network

In this subsection we assume that probability of successful reception is $p = 1$ (i.e. a reliable network), and the station a can start in arbitrary time slot t_0 (w.l.o.g. we assume that $0 \leq t_0 < n$) and continues until it learns its rank r_a. We also assume that the used permutation is $\pi_k = \mathsf{rbo}_k$, where $k = \lg n$ is positive integer.

Note that a listens until the last b_j with $j \in \{r_a - 1, r_a\}$ has been transmitted. This happens within n time slots. For given subset of indexes $S \subseteq \{0, \ldots, n-1\}$ and rank $r \in \{0, \ldots, n\}$, let $l(S, r) = \max\{i + 1 \mid i \in S \cup \{-1\} \wedge i < r\}$ and $u(S, r) = \min\{i \mid i \in S \cup \{n\} \wedge r \leq i\}$. For $t \geq t_0$, let S_t be a set of indexes of keys that have been transmitted during time slots $t_0, \ldots, t$. Then, just after t, $\mathsf{minR}[a] = l(S_t, r_a)$ and $\mathsf{maxR}[a] = u(S_t, r_a)$ and, for any $S' \subseteq S_t$, $l(S', r_a) \leq \mathsf{minR}[a] \leq \mathsf{maxR}[a] \leq u(S', r_a)$.

For a sequence of non-negative integers α we define subset of indexes $L(\alpha)$ as follows: $L(\langle\rangle) = \{0, \ldots, n-1\}$, and $L(\alpha \cdot \langle l \rangle)$ is the lth level of bisection tree formed from $L(\alpha)$ ("$\cdot$" denotes concatenation). Note that if $|L(\alpha)| = 2^{l'} \geq 1$ then, for $l < l'$, $|L(\alpha \cdot \langle l \rangle)| = 2^l$ (full levels), and for $l = l'$, $|L(\alpha \cdot \langle l \rangle)| = 1$ (the last level is singleton), and for $l > l'$, $|L(\alpha \cdot \langle l \rangle)| = 0$ (empty levels below the tree). In $\mathsf{rbo}_{\lg n}$ the sequence of subsets of indexes is: $L(\langle 0 \rangle), \ldots, L(\langle \lg n \rangle)$, and within each $L(\alpha)$ the sequence is: $L(\alpha \cdot \langle 0 \rangle), \ldots, L(\alpha \cdot \langle \lg |L(\alpha)| \rangle)$.

Lemma 4. *Let $|L(\alpha)| = 2^{l'} \geq 2$, and $1 \leq l < l'$. Let $t = \max\{\mathsf{rbo}_{\lg n}(x) \mid x \in L(\alpha \cdot \langle l \rangle)\}$. If, just after time slot t, $\mathsf{minR}[a] \geq l(L(\alpha \cdot \langle l \rangle), r_a)$ and $\mathsf{maxR}[a] \leq u(L(\alpha \cdot \langle l \rangle), r_a)$ then during each of the levels $L(\alpha \cdot \langle l+1 \rangle), \ldots, L(\alpha \cdot \langle l' \rangle)$ the station a listens at most twice.*

Proof. During transmission of the level $L(\alpha \cdot \langle l+1 \rangle)$ the station a listens only to the keys b_j with $\mathsf{minR}[a] \leq j \leq \mathsf{maxR}[a] - 1$. Since there are no such nodes in $L(\alpha \cdot \langle l \rangle)$, the only such nodes in $L(\alpha \cdot \langle l+1 \rangle)$ possibly are: the right child of $\mathsf{minR}[a] - 1$ (if $\mathsf{minR}[a] - 1 \in L(\alpha \cdot \langle l \rangle)$) and the left child of $\mathsf{maxR}[a]$ (if $\mathsf{maxR}[a] \in L(\alpha \cdot \langle l \rangle)$) in the tree $L(\alpha)$. After transmission of $L(\alpha \cdot \langle l+1 \rangle)$, we have $\mathsf{minR}[a] \geq l(L(\alpha \cdot \langle l+1 \rangle), r_a)$ and $\mathsf{maxR}[a] \leq u(L(\alpha \cdot \langle l+1 \rangle), r_a)$. Thus we can repeat the same reasoning for each following level in the tree $L(\alpha)$. □

Lemma 5. *Let $|L(\alpha)| = 2^{l'} \geq 16$, and $2 \leq l < l'$. Let $t = \max\{\mathsf{rbo}_{\lg n}(x) \mid x \in L(\alpha \cdot \langle l \rangle)\}$. If, just after time slot t, $|\{x \in L(\alpha \cdot \langle l \rangle) \mid \mathsf{minR}[a] - 1 < x < \mathsf{maxR}[a]\}| \leq 1$ then during the transmissions of $L(\alpha \cdot \langle l+1 \rangle)$ the station a listens at most four times.*

Proof. In the worst case a listens in $L(\alpha \cdot \langle l+1 \rangle)$ to some subset of: right child of $\mathsf{minR}[a] - 1$, both children of the single x between $\mathsf{minR}[a] - 1$ and $\mathsf{maxR}[a]$, and left child of $\mathsf{maxR}[a]$ in the tree $L(\alpha)$. □

Theorem 1. *If the assumptions formulated in the first paragraph of this subsection hold then the station a listens at most $4 \lg n$ times before it learns its rank.*

Proof. Let $U(\alpha, l) = \bigcup_{i=0}^{l} L(\alpha \cdot \langle i \rangle)$ (i.e. the uppermost $l+1$ levels of the tree $L(\alpha)$). Let $D(\alpha, l) = \bigcup_{i=l}^{\lg |L(\alpha)|} L(\alpha \cdot \langle i \rangle)$ (i.e. the lowest $\lg |L(\alpha)| - l + 1$ levels of the tree $L(\alpha)$).

For $t \geq 0$, let $\gamma(t)$ be the shortest sequence such that $x = \mathsf{rbo}_{\lg n}^{-1}(t \bmod n)$ is the root of the tree $L(\gamma(t))$ and let $\delta(t) = \min\{\delta \geq 0 \mid |L(\gamma(t+\delta))| \geq 2\}$. For $0 \leq i < \delta(t)$, $L(\gamma(t+i))$ is the last level (singleton) of some tree T_i. Hence, $L(\gamma(t+i+1))$ is a level of some tree T_{i+1} such that $|T_{i+1}| \geq 2 \cdot |T_i|$, or of the whole tree $L(\langle\rangle)$ if $(t+i+1) \bmod n = 0$ (in this case: $i+1 = \delta(t)$). T_i is predecessor of the level $L(\gamma(t+i+1))$ in T_{i+1}. $L(\gamma(t+\delta(t)))$ is a full level, thus its size is at least $2 \cdot |T_{\delta(t)-1}|$. Hence we have:

Claim. $|L(t+\delta(t))| \geq 2^{\delta(t)}$.

Let $\beta = \gamma(t_0 + \delta(t_0))$. If β is empty sequence, then a listens at most $\delta(t_0)$ times and then at most $\lg n + 1$ times starting from the root of the whole tree $L(\langle\rangle)$. By the Claim: $\delta(t_0) + \lg n + 1 \leq 2 \lg n + 1$.

Otherwise, let $\beta = \langle l_1, \ldots, l_R \rangle$ and let $\beta_i = \langle l_1, \ldots, l_i \rangle$. Note that R (the length of β) is the level of recursion on which the bisection tree of $L(\beta)$ is formed and that $l_R \geq 1$, since $|L(\beta)| = 2^{l_R} \geq 2$.

First a listens $\delta(t_0)$ times. By the Claim: $\delta(t_0) \leq l_R$, since $|L(\beta)| = 2^{l_R}$. Then a listens to $L(\beta)$ starting from the root of $L(\beta)$. Thus it listens at most $l_R + 1$ times and, after that, $\mathsf{minR}[a] \geq l(L(\beta), r_a)$ and $\mathsf{maxR}[a] \leq u(L(\beta), r_a)$. Then rbo steps back one recursion level. Then it listens to (possibly empty) sequence of sets $L(\beta_{R-1} \cdot \langle l_R + 1 \rangle), \ldots, L(\beta_{R-1} \cdot \langle l_{R-1} \rangle)$. By Lemma 4, a listens at most twice in each of these sets. After that $\mathsf{minR}[a] \geq l(D(\beta_{R-1}, l_R), r_a)$ and $\mathsf{maxR}[a] \leq u(D(\beta_{R-1}, l_R), r_a)$. Then rbo steps back one recursion level. Such stepping back is repeated $R-1$ times, for i taking values $R-1$,...,1. For each such i, initially $\mathsf{minR}[a] \geq l(D(\beta_i, l_{i+1}), r_a)$ and $\mathsf{maxR}[a] \leq u(D(\beta_i, l_{i+1}), r_a)$, and the following (possibly empty) sequence of sets is transmitted: $L(\beta_{i-1} \cdot \langle l_i + 1 \rangle)$, $\ldots$, $L(\beta_{i-1} \cdot \langle l_{i-1} \rangle)$. Note that, since $L(\beta_{i+1}) = L(\beta_i \cdot \langle l_{i+1} \rangle)$ is a full (i.e. not last) level of $L(\beta_i)$, for each $x \in U(\beta_i, l_{i+1} - 1)$, the predecessor and the successor of x in $L(\beta_i)$ are in $D(\beta_i, l_{i+1})$ and, hence, $|\{x \in L(\beta_i) \mid l(D(\beta_i, l_{i+1}), r_a) - 1 < x < u(D(\beta_i, l_{i+1}), r_a))\}| \leq 1$. Thus, by Lemma 5, since $L(\beta_i) = L(\beta_{i-1} \cdot \langle l_i \rangle)$, the station a has to listen at most four times in $L(\beta_{i-1} \cdot \langle l_i + 1 \rangle)$ and, by Lemma 4, at most twice in each of the remaining sets.

Consider the sequence of **all** the sets mentioned above. For each set $L(\beta_i \cdot \langle j \rangle)$ in the sequence (except $L(\beta_{R-1} \cdot \langle l_R \rangle) = L(\beta)$ – the first one) its index "j" is greater than the index of its predecessor. The greatest possible value of j is $\lg n$ (in the set $L(\langle\rangle \cdot \langle \lg n \rangle)$). Thus the number of the sets following $L(\beta)$ is at most $\lg n - l_R$ and in each of them a listens at most four times. Each set in which a has to listen more than twice (which is a full level) must be followed by at least one set (e.g. the last level) in which a has to listen at most two times. Thus the energy used by a while listening to the sequence of sets is at most $(4 \cdot \frac{1}{2} + 2 \cdot \frac{1}{2})(\lg n - l_R) + (l_R + 1) \leq 3 \lg n - 2 l_R + 1$.

This procedure is finished, for the last $i = 1$, just before time slot n that starts the next round. In this round a listens no more than $\lg n$ times (it does

not need to listen in the last level again). Adding $\delta(t_0) \leq l_R$ initial slots, we have upper bound $4 \lg n - l_R + 1 \leq 4 \lg n$ on the energy used by a. □

The estimation $4 \lg n$ of Theorem 1 seems to be very pessimistic. (In our tests a never had to listen more than $2 \lg n$ times.) Nevertheless, it shows that the station a can safely start its ranking at any time slot. (This reduces the upper bound on ranking time from $2n - 1$ to n.) The simulations indicate that rbo is also energetically efficient on unreliable network (i.e. when $p < 1$).

4 Sorting.

We assume that $\lg n$ is positive integer. Each station s_i contains variables: $\mathsf{idx}_0[s_i], \ldots, \mathsf{idx}_{\lg n}[s_i], \mathsf{minR}_0[s_i], \ldots, \mathsf{minR}_{\lg n-1}[s_i], \mathsf{maxR}_0[s_i], \ldots, \mathsf{maxR}_{\lg n-1}[s_i]$. The variables are initialized by the procedure init (see Algorithm 3). The ultimate goal for each s_i is to compute $\mathsf{idx}_{\lg n}[s_i]$, which is the index of $\mathsf{key}[s_i]$ in the sorted sequence of keys. Our algorithm is designed to perform *stable* sorting (i.e. the initial ordering between equal keys is preserved). The basic building block of our algorithm is the procedure $\mathsf{rank}(k, l, d, \pi_k)$, where $d \in \{0, 1\}$ and $0 \leq l < n/2^{k+1}$, (see Algorithm 4) that tries to find the rank of each key from the stations $s_{l \cdot 2^{k+1}+d \cdot 2^k}, \ldots, s_{l \cdot 2^{k+1}+d \cdot 2^k+2^k-1}$ in the sorted sequence of keys from the stations $s_{l \cdot 2^{k+1}+(1-d) \cdot 2^k}, \ldots, s_{l \cdot 2^{k+1}+(1-d) \cdot 2^k+2^k-1}$. Once the station s_i knows its rank r in the neighboring sequence and its index idx in its own sorted sequence, it can compute its index $(r + \mathsf{idx})$ in the sequence merged from the two sequences. The permutation π_k is either rbo_k or bo_k (defined in Section 3). For any k, $0 \leq k < \lg n$, all procedures $\mathsf{rank}(k, l, d, \pi_k)$ are used to produce indexes for sorted sub-sequences of length 2^{k+1}. This is done by procedure $\mathsf{levelRanking}(k, \pi_k)$ (see Algorithm 5). We refer to k as a *level*.

```
procedure init
Each s_i does (in parallel):
begin
    idx_0[s_i] ← 0;
    for k ← 1 to lg n do idx_k[s_i] ← NIL;
    for k ← 0 to lg n − 1 do
        minR_k[s_i] ← 0;
        maxR_k[s_i] ← 2^k;
end
```

Algorithm 3: procedure init.

Sorting algorithms can be built by composing sequences of levelRanking for various levels. For $n > 0$ and $0 < q, q' < 1$, let $c(q, q', n) = \lceil \log_{1/q} \left(\frac{2n \lg n}{q'} \right) \rceil = \lceil (1 + \lg n + \lg \lg n + \lg(1/q'))/\lg(1/q) \rceil$. We propose and analyze a simple procedure $\mathsf{sorting}_{q'}$ (see Algorithm 6) that successfully sorts with probability $1 - q'$

procedure rank(k, l, d, π_k)
for $0 \leq i < 2^k$:

- let a_i denote $s_{l \cdot 2^{k+1} + d \cdot 2^k + i}$, and
- let b_i denote $s_{l \cdot 2^{k+1} + (1-d) \cdot 2^k + i}$.

for *time slot* $t \leftarrow 0$ **to** $2^k - 1$ **do**
  the (at most one) b_j with $\pi_k(\mathsf{idx}_k[b_i]) = t$ broadcasts $key = \mathsf{key}[b_j]$;
  let $x = \pi_k^{-1}(t)$;
  each a_i with $\mathsf{minR}[a_i] \leq x < \mathsf{maxR}_k[a_i]$ does:
  begin
    a_i listens;
    if a_i *received key* **then**
      (* comparison for *stable* ranking *)
      if *(*$d = 0$ *and* $\mathsf{key}[a_i] \leq key$*) or (*$d = 1$ *and* $\mathsf{key}[a_i] < key$*)* **then**
        $\mathsf{maxR}_k[a_i] \leftarrow x$;
      else
        $\mathsf{minR}_k[a_i] \leftarrow x + 1$;
      (* cascading computation of indexes *)
      $k' \leftarrow k$;
      while $k' < \lg n$ *and* $\mathsf{idx}_{k'}[a_i] \neq NIL$ *and* $\mathsf{minR}_{k'}[a_i] = \mathsf{maxR}_{k'}[a_i]$ **do**
        $\mathsf{idx}_{k'+1}[a_i] \leftarrow \mathsf{idx}_{k'}[a_i] + \mathsf{minR}_{k'}[a_i]$;
        $k' \leftarrow k' + 1$;
  end

Algorithm 4: Procedure rank.

procedure levelRanking(k, π_k)
for $l \leftarrow 0$ **to** $n/(2^{k+1}) - 1$ **do**
  rank$(k, l, 1, \pi_k)$;
  rank$(k, l, 0, \pi_k)$;

Algorithm 5: Procedure levelRanking.

procedure sorting$_{q'}$
init;
Let $q = 1 - p$, where p is probability of successful reception;
for $k \leftarrow 0$ **to** $\lg n - 1$ **do**
  repeat $c(q, q', n)$ times levelRanking(k, rbo_k);

Algorithm 6: Procedure sorting.

by repeating levelRanking $c(q, q', n)$ times on each level. The output consists of the final values of $\mathsf{idx}_{\lg n}$ in the stations.

Theorem 2. *For $0 < q' < 1$, the procedure $\mathsf{sorting}_{q'}$ (Algorithm 6) sorts any input sequence with probability greater or equal $1 - q'$.*

Proof. Let $q = 1 - p$ and $c = c(q, q', n)$. Let $\mathcal{Q}$ be the event that $\mathsf{sorting}_{q'}$ failed to sort (i.e. some indexes remained uncomputed). For $0 \leq k < \lg n$, let $\mathcal{Q}_k$ be the event that the first failure occurred at level k (i.e. some $\mathsf{idx}_{k+1}[s_i]$ remained uncomputed, while all values $\mathsf{idx}_{k'}[s]$, for $0 \leq k' \leq k$, for each station s, are computed.) Thus $Pr(\mathcal{Q}) = \sum_{k=0}^{\lg n - 1} Pr(\mathcal{Q}_k)$ ($\mathcal{Q}$ is disjoint union of all events $\mathcal{Q}_k$). Let $\mathcal{F}_k$ be the event that repeating c times $\mathsf{levelRanking}(k, \mathsf{rbo}_k)$ fails to compute all indexes on level $k+1$ under the condition that all indexes on levels up to k have been computed. Let $q = 1 - p$. By Lemma 1, the probability that some given index remains uncomputed is not greater than $2 \cdot q^c$. Thus $Pr(\mathcal{F}_k) \leq 2nq^c$, as we have to compute n indexes. Let $\mathcal{E}_k$ be the event that all indexes on levels up to k has been properly computed in $\mathsf{sorting}_{q'}$. Then $Pr(\mathcal{Q}_k) = Pr(\mathcal{E}_k) \cdot Pr(\mathcal{F}_k) \leq Pr(\mathcal{F}_k)$ and, hence, $Pr(\mathcal{Q}) \leq 2nq^c \cdot \lg n$. It is easy to verify that $c \geq \min\{c \mid q^c \cdot 2n \lg n \leq q'\}$. This completes the proof. □

Theorem 3. *For any input, the expected energy used for listening by any single station in $\mathsf{sorting}_{q'}$ is at most $\lg n \cdot (1 + (c-1)(2/p - 2) + (2/p - 1)(\lg n - 1)/2) + c(n-1)q'$, where $c = c(q, q', n)$. The energy used for sending by any single station is $c \lg n$. Time of $\mathsf{sorting}_{q'}$ is $cn \lg n$.*

Proof. Let the input sequence be arbitrary and let s be any of the stations. Let X be random variable that is the energy used for listening by s. Let X_k be random variable that is the energy used by s in all c levelRankings on level k. Thus $X = \sum_{k=0}^{\lg n - 1} X_k$. Let Ω be the set of all elementary events (i.e. of all possible computations). Note that $E[X] = \sum_{\omega \in \Omega} X(\omega) \cdot Pr(\omega)$, where $X(\omega)$ is the energy used by s in the computation ω and $Pr(\omega)$ is the probability of this computation. Let the events $\mathcal{Q}$, $\mathcal{E}_k$ be defined as in the proof of Theorem 2. Let $\mathcal{E} = \mathcal{E}_{\lg n}$. Note that Ω is a disjoint union of $\mathcal{Q}$ and $\mathcal{E}$ and, hence, $E[X] = S_{\mathcal{Q}} + S_{\mathcal{E}}$, where $S_{\mathcal{Q}} = \sum_{\omega \in \mathcal{Q}} X(\omega) \cdot Pr(\omega)$ and $S_{\mathcal{E}} = \sum_{\omega \in \mathcal{E}} X(\omega) \cdot Pr(\omega)$.

To estimate $S_{\mathcal{Q}}$ note that in the levelRanking on level k there are only 2^k time slots in which s is allowed to listen. Thus $X_k(\omega) \leq c \cdot 2^k$ and $X(\omega) \leq c \sum_{k=0}^{\lg n - 1} 2^k = c(n-1)$ and $S_{\mathcal{Q}} \leq c(n-1) \cdot \sum_{\omega \in \mathcal{Q}} Pr(\omega) \leq c(n-1)q'$ (by Theorem 2).

To estimate $S_{\mathcal{E}}$ note that

$$S_{\mathcal{E}} = \sum_{\omega \in \mathcal{E}} \sum_{k=0}^{\lg n - 1} X_k(\omega) \cdot Pr(\omega) = \sum_{k=0}^{\lg n - 1} \sum_{\omega \in \mathcal{E}} X_k(\omega) \cdot Pr(\omega)$$
$$\leq \sum_{k=0}^{\lg n - 1} \sum_{\omega \in \mathcal{E}_k} X_k(\omega) \cdot Pr(\omega) \leq \sum_{k=0}^{\lg n - 1} \sum_{\omega \in \mathcal{E}_k} \frac{X_k(\omega) \cdot Pr(\omega)}{Pr(\mathcal{E}_k)} = \sum_{k=0}^{\lg n} E[X_k | \mathcal{E}_k],$$

where $E[X_k|\mathcal{E}_k]$ is expected value of X_k under the condition $\mathcal{E}_k$ that all indexes up to level k have been computed. (The first inequality above follows from $\mathcal{E} \subseteq$

$\mathcal{E}_k$, and the second one follows from $Pr(\mathcal{E}_k) \leq 1$.) Under the condition $\mathcal{E}_k$ the expected energy used for listening by s during the first levelRanking on level k is at most $1 + k(2/p - 1)$ (by Lemma 3). By Lemma 2, the expected value of $\Delta = \mathsf{maxR}_k[s] - \mathsf{minR}_k[s]$ after the first levelRanking on level k is $2/p - 2$. During each of the remaining $c - 1$ levelRankings on level k station s can listen to at most Δ stations, thus the expected listening energy for these levelRankings can be bounded by $(c-1)\cdot(2/p-2)$. We have $E[X_k|\mathcal{E}_k] \leq 1+k(2/p-1)+(c-1)\cdot(2/p-2)$. Thus $S_{\mathcal{E}} \leq \sum_{k=0}^{\lg n-1}(1 + k(2/p - 1) + (c - 1) \cdot (2/p - 2)) = \lg n(1 + (c - 1)(2/p - 2)) + (2/p - 1)\frac{\lg n(\lg n-1)}{2}$.

The limits on time and sending energy follow from the fact that each station broadcasts only once in each levelRanking. □

Corollary 1. *Algorithm* $\mathsf{sorting}_{1/n}$ *sorts any input with probability at least* $1 - \frac{1}{n}$ *in time* $O(n \lg^2 n)$ *and, for each station* s*, the expected energy used by* s *is* $O(\lg^2 n)$.

Acknowledgments

Thanks to Maciej Gębala for helpful comments.

References

1. M. Ajtai, J. Komlós and E. Szemerédi. Sorting in $c \log n$ parallel steps. Combinatorica, Vol. 3, pages 1–19, 1983.
2. Amitava Datta and Albert Y. Zomaya. An Energy-Efficient Permutation Routing Protocol for Single-Hop Radio Networks. IEEE Trans. Parallel Distrib. Syst., 15:331-338, 2004.
3. M. Gębala, M.Kik. Counting-Sort and Routing in a Single Hop Radio Network. ALGOSENSORS 2007, LNCS 4837, pp. 138-149, 2008.
4. M. Kik. Merging and Merge-sort in a Single Hop Radio Network. SOFSEM 2006, LNCS 3831, pp. 341-349, 2006.
5. M.Kik. Sorting Long Sequences in a Single Hop Radio Network. MFCS 2006, LNCS 4162, pp. 573-583, 2006.
6. K. Nakano. An Optimal Randomized Ranking Algorithm on the k-channel Broadcast Communication Model. ICPP 2002, pp. 493-500, 2002.
7. K. Nakano, S. Olariu, A. Y. Zomaya. Energy-Efficient Permutation Routing in Radio Networks. IEEE Transactions on Parallel and Distributed Systems, 12:544-557, 2001.
8. K. Nakano, S. Olariu, A. Y. Zomaya. Energy-Efficient Routing in the Broadcast Communication Model. IEEE Trans. Parallel Distrib. Syst., 13:1201-1210, 2002.
9. M. Singh and V. K. Prasanna. Optimal Energy Balanced Algorithm for Selection in Single Hop Sensor Network. SNPA ICC, May 2003.
10. M. Singh and V. K. Prasanna. Energy-Optimal and Energy-Balanced Sorting in a Single-Hop Sensor Network. PERCOM, March 2003.