Contention Resolution with Predictions

Seth Gilbert
seth.gilbert@comp.nus.edu.sg
National University of Singapore
Singapore

Calvin Newport
cnewport@cs.georgetown.edu
Georgetown University
United States

Nitin Vaidya
nitin.vaidya@georgetown.edu
Georgetown University
United States

Alex Weaver
aweaver@cs.georgetown.edu
Georgetown University
United States

ABSTRACT

In this paper, we consider contention resolution algorithms that are augmented with predictions about the network. We begin by studying the natural setup in which the algorithm is provided a distribution defined over the possible network sizes that predicts the likelihood of each size occurring. The goal is to leverage the predictive power of this distribution to improve on worst-case time complexity bounds. Using a novel connection between contention resolution and information theory, we prove lower bounds on the expected time complexity with respect to the Shannon entropy of the corresponding network size random variable, for both the collision detection and no collision detection assumptions. We then analyze upper bounds for these settings, assuming now that the distribution provided as input might differ from the actual distribution generating network sizes. We express their performance with respect to both entropy and the statistical divergence between the two distributions—allowing us to quantify the cost of poor predictions. Finally, we turn our attention to the related perfect advice setting, parameterized with a length $b \geq 0$, in which all active processes in a given execution are provided the best possible b bits of information about their network. We provide tight bounds on the speed-up possible with respect to b for deterministic and randomized algorithms, with and without collision detection. These bounds provide a fundamental limit on the maximum power that can be provided by any predictive model with a bounded output size.

1 INTRODUCTION

In this paper, we study distributed algorithms that leverage predictions about their environment to improve worst-case performance bounds. Motivated by recent investigations in the context of sequential algorithms (e.g., [18]), we imagine these predictions might be generated in practice by machine learning models able to observe the behavior of a given environment over time. As Mitzenmacher and Vassilvitskii note in their recent study of the ski rental problem with predictions [18], one appealing goal for this general approach is to produce algorithms that perform no worse than our current optimal solutions, but that will subsequently improve "for free" as the machine learning models generating the predictions they leverage improve in the future.

To help establish foundations for studying distributed algorithms with predictions, we focus on the classical contention resolution problem, as it is both simple and well-studied. We begin by producing new lower and upper bounds on the speed-up achievable when the algorithm is provided a predicted distribution over network sizes. To achieve the strongest results, our lower bounds assume the distribution accurately describes the likelihood of each network size occurring, while our upper bounds explicitly capture how this performance degrades with respect to the Kullback–Leibler (KL) divergence between the predicted distribution and the actual distribution. We also study lower and upper bounds for a *perfect advice* setting in which all active processes are provided the same b bits of advice generated by a function with perfect knowledge of the network. The goal here is to understand the maximum possible improvement for a given amount of information.

As we elaborate below, our network size prediction results are built on a novel connection between contention resolution and information theory, leading to speed-up results expressed with respect to the entropy of the size distribution, lower bounds that leverage the classical Source Code Theorem for optimal coding on noiseless channels, and upper bounds that use optimal codes in their design. A nice property of our information theory-based lower bounds is that when they are applied to distributions with high entropy, they match (or almost match) longstanding existing lower bounds for contention resolution with uniform algorithms, highlighting yet another mathematical framework through which to understand this fundamental problem.

1.1 Contention Resolution with Network Size Predictions

The contention resolution problem assumes an unknown and non-empty subset $P \subseteq V$ of players from $|V| = n$ possible participants are activated and connected to a shared channel. Time proceeds in synchronous rounds. In each round, each player can choose to transmit or receive. If two or more players transmit, all messages in that round are lost due to collision. If the channel supports collision detection, all players (including the transmitters) detect a collision. If the channel does not support collision detection, then players detect silence. The problem is solved in the first round in which exactly one player transmits.

In resolving contention on a shared channel, knowing the size k of the participant set P is useful. Indeed, many of the standard optimal worst-case algorithms operate by efficiently trying to find a good estimate of this size. The *decay* strategy for channels without collision detection, for example, introduced by Bar Yehuda et al. [3], can be understood as cycling through $\log n$ geometrically-distributed guesses of the network size. Given collision detection, Willard's algorithm [23] achieves optimal time complexity by conducting a binary search over these same guesses.

In a worst-case setting in which no information is provided about the network size, Jurdziński and Stachowiak proved a lower bound of $\Omega(\log n / \log \log n)$ on the expected rounds to solve contention resolution with no collision detection and uniform algorithms (in which probabilities are fixed in advance) [15]. Colton et al. [12] later eliminated the $\log \log n$ factor, and Newport [19] subsequently generalized the bound to non-uniform algorithms. These results match the $O(\log n)$ expected round complexity of the *decay* strategy [3]. When collision detection is available, Willard establishes $\Theta(\log \log n)$ to be a tight bound [23].

In both the collision detection and no collision detection settings, if the algorithm is given an accurate estimate $\hat{k} = \Theta(k)$ of the actual network size k, the problem can be solved in $O(1)$ rounds in expectation by simply transmitting with probability $1/\hat{k}$ in each round. In real networks, however, it may be difficult to predict with confidence the exact network size to expect next. It is arguably more likely that learning models will generate *distributions* over the likelihood of various sizes. The relevant question we explore in Section 2, therefore, is how much prediction distributions of varying usefulness and quality allow us to speed-up from the worst-case toward the best-case bounds.

In Section 2, we augment the standard contention resolution problem to provide each algorithm as input the definition of a random variable X defined over the possible participant set sizes from 1 to n. The goal is to use the distribution defining X to speed up contention resolution if possible. (For concision, in the following we sometimes say the algorithm is provided a random variable over network sizes as input. In these instances, we mean that the algorithm is provided as input the full distribution defining the variable.)

As we argued earlier, however, we do not need the exact network size to solve contention resolution fast. An estimate within a constant factor of the real size is sufficient. Given a network size random variable X, let $c(X)$ be the *condensed* version of X that aggregates the probability mass over $\lceil \log n \rceil$ values ranges of

	Lower Bounds (*)	Upper Bounds (*)
No Collision Detection	$\Omega\left(\frac{2^{H(c(X))}}{\log \log n}\right)$	$O\left(2^{2H(c(X))}\right)$
Collision Detection	$\frac{H(c(X))}{2} - O(\log \log \log \log n)$	$O(H^2(c(X)))$

Table 1: Here we summarize our results for contention resolution with network size predictions. $H(c(X))$ represents the entropy of the condensed probability distribution over the possible network sizes.
(*) Lower bounds are expressed as the expected number of rounds while upper bounds are expressed with respect the number of rounds with at least constant probability.

geometrically increasing size. Formally, for each $i \in [\lceil \log n \rceil]$ let

$$\Pr(c(X) = i) = \sum_{2^{i-1} < j \leq 2^i} p_j.$$

Intuitively, knowing a value $i \in [\lceil \log n \rceil]$, such that $k = \Theta(2^i)$, is sufficient. We therefore express the bounds that follow for a given X with respect to $c(X)$.

Lower Bounds. We begin with lower bounds. To achieve the strongest possible result, we assume that the variable X is accurate, in the sense that the network will actually determine the number of participants according to X. We also assume *uniform* algorithms in which all players use the same transmission probability in each round from a fixed schedule.[1]

Intuitively, if $c(X)$ places all of its probability mass on a single network size range, then we are in the perfect prediction setting and can solve the problem in $O(1)$ rounds. On the other extreme, if $c(X)$ describes a uniform distribution, we are likely unable to do much better than the worst-case bounds. To describe these cases we need a property of these distributions that succinctly captures their predictive power. The natural candidate here is *entropy*—and this is indeed what we we end up deploying in our results.

We begin in Section 2.3 by considering contention resolution without collision detection. We prove that for a given network size random variable X, contention resolution requires $\Omega\left(\frac{2^{H(c(X))}}{\log \log n}\right)$ rounds in expectation, where H denotes Shannon entropy. For maximum values of $H(c(X))$, this reduces to $\Omega(\log n / \log \log n)$ rounds, exactly matching the original uniform algorithm lower bound from [15]—a result that is interesting in its own right, as it shows yet another approach to proving limits to this fundamental problem.

The core idea driving this bound is a connection between contention resolution and coding. Using a reduction argument involving an intermediate game we call range finding, we formalize the intuition that solving contention resolution requires an algorithm to try a transmission probability relatively close to the optimal probability. We can therefore use a contention resolution algorithm designed for X to help construct a code for a given symbol source $c(X)$. Let $s \in [\lceil \log n \rceil]$ be the symbol to be transmitted. We can then

---
[1]It is important to point out that until recent work [19], most existing lower bounds for contention resolution assumed uniform algorithms.

consider the execution of our algorithm in a network of size 2^s, and identify the first round r such that the algorithm attempts a transmission probability near 2^{-s}, and send r to the receiver. Because the receiver knows the same algorithm, it can use the probability scheduled for round r to determine s. If the algorithm terminates fast in expectation with respect to X, then the expected code length of our code for $c(X)$ is small. The Source Code Theorem (see Section 2.2), however, tells us that the average code length for $c(X)$ is lower bounded by $H(c(X))$. Because, roughly speaking, the code length is computed as the logarithm of the round complexity, we get a bound of roughly $2^{c(H(X))}$ rounds.

In Section 2.4, we leverage a similar connection between contention resolution and information theory to prove that given collision detection, uniform algorithms that solve contention resolution with respect to a network size variable X require $\Omega(H(c(X)))$ in expectation. For the maximum possible entropy of $c(X)$, this matches the original $\Omega(\log \log n)$ lower bound proven by Willard, using non-information theoretic techniques [23].

A uniform algorithm in the collision detection setting can be understand as a function that maps a binary collision history[2] to the uniform probability that all players try in the current round given the current collision history. We show how to construct a code for $c(X)$ for such an algorithm by directly sending the shortest collision history that corresponds to a probability well-matched to the current symbol. (As in the no collision detection case, there is actually an intermediate step involved here where our contention resolution algorithm reduces to a simpler game called range finding, and it is in fact this range finding solution from which we generate our code.) In this case, they round complexity is the same as the code length. It follows that $H(c(X))$ must bound the expected value of the former.

Upper Bounds. We next turn our attention to produce contention resolution upper bounds that take a network size random variable as input and attempt to come as close as possible to matching the relevant lower bounds. We start in Section 2.5 with an algorithm for the no collision detection setting. Let X be the actual distribution from which the network size will be drawn. Let Y be the distribution the algorithm is given as input (e.g., the distribution learned). We analyze a natural strategy: sort the values in $[\lceil \log n \rceil]$ in decreasing order of likelihood given $c(Y)$; visit these values in turn, for each such i transmitting withing probability 2^{-i}.

We prove that with constant probability, this strategy succeeds in $O(2^T)$ rounds, where:

$$T = 2H(c(X)) + 2D_{KL}(c(X)\|c(Y)),$$

and D_{KL} denotes the Kullback–Leibler divergence between the two distributions (see Section 2.2). When $Y = X$, this simplifies to $O(2^{2H(c(X))})$ which matches the same general exponential form of our lower bound, but includes an extra constant factor in the exponent. We conjecture that something like this extra factor may be fundamental in the analysis of this natural strategy. We support this conjecture by noting a straightforward application of a

cryptography result due to Pliam [20] indicates that for every constant $\alpha \geq 1$, there is a random variable X_α such that this strategy requires more than $\alpha 2^{H(c(X_\alpha))}$ rounds to succeed with constant probability.[3]

An important characteristic of the Kullback–Leibler divergence is that is if each probability in Y is off by at most a bounded constant fraction from the real probability in X, $D_{KL}(c(X)\|c(Y)) = O(1)$. This establishes that our algorithm does not require a precise prediction to be useful, and its efficiency will increase smoothly along with prediction quality.

In Section 2.4, we consider algorithms for the collision detector setting. Given network size variable Y, the algorithm first constructs an optimal code f with respect to source $c(Y)$. It then considers all codes from shortest to longest in length. For each length ℓ, it considers all symbols mapped to codes of this length. Ordering these symbols from smallest to largest, it deploys the collision detector-driven binary search strategy introduced by Willard [23] to explore if any of these symbols correspond to the correct network size.

We show this algorithm solves contention resolution with constant probability in $O((H(c(X))+D_{KL}(c(X)\|c(Y)))^2)$ rounds, which reduces to $O((H(c(X)))^2)$ when the probabilities in Y are all within a bounded constant factor of the real probabilities in X. As before, our upper bound shares the same general form as the lower bound (in this case, a result expressed as a polynomial of $H(c(X))$), but is not exactly tight. Closing these final gaps between the upper and lower bounds for small divergence is not obvious, but likely tractable; and therefore left as intriguing future work. We also note that for clarity, we focused only on one-shot attempts to resolve contention that succeed with constant probability. Achieving good bounds on expected time also represents important future work.

1.2 Contention Resolution with Perfect Advice

In the interest of further expanding our foundational understanding of contention resolution with predictions, in Section 3 we consider contention resolution with so-called *perfect advice*. In more detail, we fix parameter b on the maximum number of bits of information provided to the algorithm as advice or predictions about the network. We assume an advice function f_A that returns the same advice $f_A(P)$ of size b bits to every node in the set P of participating players. We seek to understand tight bounds on the speed-up possible given b bits of perfect advice. Tight bounds here characterize the best possible improvements possible for a given quantity of advice—lower bounds that can later inform the study of more specific prediction models.

We prove this question amenable to analysis by providing tight asymptotic bounds on achievable speed-up, with respect to parameter b, for both deterministic and randomized algorithms, with and without collision detection. Our lower bound results for deterministic algorithms leverage existing lower bounds on a combinatorial object called a *strongly selective family* [6], whereas our randomized lower bounds leverage a reduction from the well-understood case where $b = 0$.

---

[2]In the collision detection model, in every round, either every player detects a collision or no player detects a collision. We can therefore encode this history for the r rounds of an execution as a binary string $b_1 b_2 \ldots b_r$, where $b_i = 0$ if there was no collision in round i, and $b_i = 1$ if there was a collision in i.

[3]To be more precise, the result in [20] can be used to show more than $\alpha 2^{H(c(X_\alpha))}$ rounds are needed to arrive at the exact correct value. Formalizing this result would require quantification of the reality that an exact guess is not needed to solve the problem. This crude application, however, serves its purpose of bolstering our conjecture that $2^{H(c(X))}$ is insufficient.

	Deterministic	Randomized
No Collision Detection	$\Theta(n^{1-b/\log n})$	$\Theta(\log n / 2^b)$
Collision Detection	$\Theta(\log n - b)$	$\Theta(\log \log n - b)$

Table 2: Here we summarize our results for deterministic
and randomized contention resolution given b bits of perfect
advice. The randomized results are expressed with respect to
expected round complexity. All bounds presented are tight.

1.3 Other Related Work

There is a long history of studying algorithms for contention reso-
lution in wireless networks, from Chlamtac and Kutten in 1985 [5]
and Willard [23] to the present, including tight upper and lower
bounds for the problem in a variety of settings (e.g., with and with-
out collision detection). Several recent directions have influenced
this paper.

First, there have been a series of papers recently looking at
the minimum number of bits of information per node needed to
solve broadcast in a multihop wireless network. In 2006, Fraigniaud,
Ilcinkas, and Pelc [13] proposed measuring the amount of *advice*
needed to solve various a distributed problems, and later Ilcinkas,
Kowalski, and Pelc [14] applied this to the problem of broadcast in
a radio network. Further results on this were given by Ellen, Gorain,
Miller, and Pelc [11], Ellen and Gilbert [10], and Bu, Potop-Butucaru,
and Rabie [4]. These results differed from our results on perfect
advice in Section 3 in that they focused on multihop networks,
giving a small number of distinct bits of advice to each node. Thus
the total number of bits of advice (e.g., $\Theta(n)$) was significantly
larger than in this paper.

Second, there has been much interest lately on how well al-
gorithms can do with additional advice. The general idea is that
machine learning and other prediction techniques may be able
to *sometimes* provide good advice that allows algorithms to run
faster; the challenge lies in ensuring that they continue to perform
well when the advice is faulty. Much of this direction began with
Mitzenmacher's work on the Learned Bloom Filter in 2018 [17], and
has continued with a variety of related online algorithms such as
work by Kumar, Purohit and Svitkina's [21] and Lattanzi, Lavastida,
Moseley and Vassilvitskii [16] on scheduling. Our paper does not
focus on machine learning per se, but does look at how extra infor-
mation can help a distributed algorithm, and examines the impact
on performance of faulty advice.

Third, one of the goals of this paper is to construct a connec-
tion between information theory and distributed algorithms, and
there have been several recent papers that have used information
theory and coding theory techniques to better understand radio
network algorithms. Dufoulon, Burman, and Beauquier [8] recently
showed how to use techniques from coding theory to communicate
efficiently in *beeping* networks; Ashkenazi, Gelles, and Leshem [2]
and Efremenko, Kol, and Saxena [9] independently showed how to
use techniques from coding theory to overcome noise in beeping
networks. This series of recent works, along with our own, leads
us to believe that there are powerful connections to information
theory here.

2 CONTENTION RESOLUTION WITH NETWORK SIZE PREDICTIONS

Here we investigate an obvious question. Assume over time you
have learned a distribution that characterizes the likelihood of var-
ious network sizes for the instance of contention resolution you
are about to execute. To what degree can you leverage these pre-
dictions to outperform worst-case bounds? We study this question
from both the upper and lower bound perspective, considering both
the collision detection and no collision assumptions. As elaborated
below, in doing so we establish a novel connection between con-
tention resolution and core information theorey results regarding
coding on noiseless channels.

2.1 Uniform Algorithms

We focus on *uniform* contention resolution algorithms, in which
participants rely on predetermined probabilities. With no collision
detection, a uniform algorithm can be interpreted as a sequence
of probabilities, $p_1, p_2, p_3, \ldots$, such that in round i, all participants
broadcast with probability p_i. With collision detection, a uniform
algorithm can be interpreted as a function f from collision histories
to broadcast probabilities. For a given round r, let the binary string
$B = b_1, b_2, \ldots, b_{r-1}$ describe the collision history through the first
$r - 1$ rounds of the given execution (i.e., $b_i = 0$ indicates there was
no collision in round i, while $b_i = 1$ indicates there was). In round
r, all players transmit with the same uniform probability $f(B)$.
Uniformity is a common assumption in the study of contention
resolution. Many previous lower bounds assume this property;
e.g., [7, 15, 23].

2.2 Preliminaries

Fix a network size n. Let X be the discrete random variable that de-
termines the number of participants in each instance of contention
resolution. In more detail, X takes its values from $1, 2, \ldots, n$, that oc-
cur with probabilities $p_1, p_2, \ldots, p_n$, respectively. In our setting, the
size k of the participant set is determined by X, leaving the adver-
sary only to determine *which* k nodes participate. Notice, however,
when considering uniform algorithms the identity of the partici-
pants is not consequential as transmission behavior is determined
exclusively by the algorithm and the collision history.

In our algorithms, we sometimes talk about this random variable
X, or perhaps an estimate Y of this random variable, being provided
as input to the algorithm. This is shorthand for the more accurate
statement that the underlying *distribution* defining the variable is
provided as input. That is, the algorithm is given for each network
size, a prediction of the probability that the network size is drawn.

In our analysis, we will find it useful to reference a *condensed*
version of X, we call $c(X)$, which aggregates the probability mass
spread over n possible network sizes into $\log n$ geometric size
ranges. (Assume all logs are base 2.) To formalize this latter def-
inition, let $L(n) = \{1, 2, 3, \ldots, \lceil \log n \rceil\}$. We associate each range
$i \in L(n)$ with the values in the interval $(2^{i-1}, 2^i]$. That is, $i = 1$ is
associated with just the value 2, $i = 2$ is associated with the range
3 to 4, $i = 3$ is associated with 5 to 8, and so on. Implicit in this
definition is the assumption that the network size is always of size

at least 2 as there is no contention to resolve in a network of size less than 2.[4]

The random variable $c(X)$ takes its values from $L(n)$. For each $i \in L(N)$, let $q_i = \Pr(c(X) = i)$, defined as follows:

$$q_i = \sum_{2^{i-1} < j \leq 2^i} p_j$$

The analysis that follows makes use of the following version of Jensen's inequality applied to concave functions:

THEOREM 2.1 (JENSEN'S INEQUALITY). *If $p_1, \ldots, p_n$ are real numbers where $p_i > 0$ for all $i \in [n]$ and $\sum_{i \in [n]} p_i = 1$, and f is a real continuous function that is concave, then: $f\left(\sum_{i=1}^n p_i x_i\right) \geq \sum_{i=1}^n p_i f(x_i)$.*

It also builds on the lower bound result from Shannon's famed Source Coding Theorem, which concerns the efficiency of codes on noiseless channels:

THEOREM 2.2 (SOURCE CODE THEOREM [22]). *Let X be a random variable taking values in some finite alphabet Σ. Let f be a uniquely decodable code from this alphabet to $\{0, 1\}$. Let S be the random variable that describes the length of codeword $f(X)$. Let H be the entropy function. It follows:*

$$H(X) \leq E(S)$$

Our upper bound analysis considers the case in which the network size distribution Y provided as input to our algorithms does not exactly match the actual distribution X from which the network size will be drawn. Drawing from standard statistics, we can use the Kullback-Leibler divergence between the two distributions, denoted $D_{KL}(X\|Y)$, to quantify their differences. We then leverage the following well-known information theory result which bounds the decrease in coding performance, with respect to $D_{KL}(X\|Y)$, when you build an construct an optimal code for Y that you then combine with symbol source X (for more on Kullback-Leibler divergence and the coding bound see the excellent review in [7]):

THEOREM 2.3. *Let X and Y be random variables taking values in some finite alphabet Σ. Let f be an optimal, uniquely decodable code from Y to $\{0, 1\}$. Let S be the random variable that describes the length of codeword $f(X)$. Let H be the entropy function and D_{KL} be the Kullback-Leibler divergence of two distributions. It follows:*

$$H(X) + D_{KL}(X\|Y) \leq E(S) \leq H(X) + D_{KL}(X\|Y) + 1$$

Lastly, we note that $D_{KL}(X\|X) = 0$ for any random variable X.

2.3 Lower Bound for No Collision Detection

Our goal is to prove the following lower bound that connects contention resolution with a known network size distribution to the entropy of the condensed version of that distribution:

THEOREM 2.4. *Fix a uniform algorithm A for a network of size n. Let $t_X(n)$ be the expected round complexity for A to solve contention resolution on a channel with no collision detection and the number*

of participants determined by random variable X. It follows: $t_X(n) = \Omega\left(\frac{2^{H(c(X))}}{\log \log n}\right)$.

Our proof strategy deploys two steps. We begin by defining a more abstract combinatorial-style problem called *range finding*, which we can more directly and clearly connect to entropy. We then show how to transform a contention resolution solution into range finding solution with a related time complexity. Note that due to space limitations many of the proofs that follow have been omitted but can be found in the full version of this paper [1].

Range Finding. The range finding problem is parameterized with a network size n and range expressed as a function $f(n)$ of this size. A range finding strategy can take the form of a sequence of values from $L(n)$, or a binary tree with its nodes labelled with values from $L(n)$. Here we define the version defined with respect to a sequence, as this is the version needed for our proof of the above theorem. The binary tree variation will be used when we later consider contention resolution with collision detection.

We say a sequence $S = v_1, v_2, \ldots, v_k$ *solves* the $(n, f(n))$-range finding problem in t steps for a given target $v \in L(n)$, if $S[t]$ is the first position in S such that $|S[t] - v| \leq f(n)$, where $S[t] = v_t$ is the tth element of S. To handle probabilistic selections of targets from a known distribution, fix some random variable Y that takes values from $L(n)$. For each $i \in L(n)$, let $p_i' = \Pr(Y = i)$. We say S solves $(n, f(n))$-range finding in expected time t with respect to Y, if t is the expected step at which A solves the problem when the target value is determined by Y.

Bounding Sequence Range Finding Using Entropy. Assume S is a sequence that solves $(n, f(n))$-range finding in expected time T with respect to some distribution Y over $L(n)$. We can use S to design a code for source Y with an efficiency determined by $f(n)$. We leverage this connection to prove the following about the connection between range finding and entropy for a range $f(n) = O(\log \log n)$ that will prove useful for our subsequent attempts to connect contention resolution to range finding:

LEMMA 2.5. *Let S be a sequence that solves $(n, \alpha \log \log n)$-range finding for some constant $\alpha \geq 1$ and network size $n > 1$. Assume that the range is determined by random variable Y. Let Z be the random variable describing the complexity of S. It follows:*

$$E(Z) = \Omega\left(2^{H(Y)}/(\alpha \log \log n)\right),$$

where H is the entropy function.

PROOF. Fix some n, α, S and Y as specified by the theorem statement. We can use S to design a code for transmitting symbols from $L(n)$ over a noiseless channel as follows:

- Initialize the sender and receiver with sequence S.
- To communicate a value $x \in L(n)$, the sender transmits (r, d) to the receiver, where: r is the first round such that $S[r]$ is within $\alpha \log \log n$ of x (i.e., the first round to solve (α, n)-range finding for x); and $d = x - v_r$ is the distance of $S[r]$ from x.
- The receiver can then locally calculate $x = S[r] + d$.

Let us call the above scheme *target-distance* coding. We now relate the expected code length with this scheme with the expected

---

[4]This assumption can hold without loss of generality as all algorithms can eliminate the $n = 1$ possibility in an additional early round in which all players transmit with probability 1.

complexity of range finding with A. Consider the performance of the target-distance coding scheme based on these values. For a given $x \in L(n)$, that is solved by S by round r, the code length of this scheme is upper bounded by $\log r + \log (\alpha \log \log n)$ The expected code length, therefore, is upper bounded as:

$$E(\log Z) + \lceil \log (\alpha \log \log n) \rceil + 1,$$

where the additional bit indicates the sign of d. Noting the concavity of the log function we can apply Jensen's Inequality (Theorem 2.1) to further refine this bound as follows:

$$E(\log Z) + \lceil \log (\alpha \log \log n) \rceil + 1 \leq \log E(Z) + \lceil \log (\alpha \log \log n) \rceil + 1.$$

The Source Code Theorem (Theorem 2.2) tells us that the average code length of this scheme is lower bounded by the entropy of Y. It follows that $H(Y) \leq \log E(Z) + \lceil \log (\alpha \log \log n) \rceil + 1$, which implies:

$$
\begin{aligned}
2^{H(Y)} &\leq 2^{\log E(Z) + \lceil \log (\alpha \log \log n) \rceil + 1} \Rightarrow \\
2^{H(Y)} &\leq 2^{\log E(Z)} 2^{\lceil \log (\alpha \log \log n) \rceil + 1} \Rightarrow \\
2^{H(Y)} &\leq E(Z) \cdot 4\alpha \log \log n \Rightarrow \\
E(Z) &\geq \frac{2^{H(Y)}}{4\alpha \log \log n} = \Omega\left(\frac{2^{H(Y)}}{\alpha \log \log n}\right),
\end{aligned}
$$

as claimed by the lemma. □

Solving Range Finding with Contention Resolution. Here we transform a solution to contention resolution to a sequence that solves range finding with a similar expected complexity. Contention resolution is a more general problem than range finding, so care is needed to tame its possible unexpected behaviors. We begin by defining an algorithmic process for transforming a uniform algorithm A into a range finding sequence S_A. We then analyze the properties of S_A.

Algorithm 1: RF-Construction

Input: Uniform contention resolution algorithm
$\qquad A = p_1, p_2, \ldots, p_z$
Output: Range finding sequence S_A
$S_A \leftarrow \emptyset$;
$j \leftarrow 0$;
for $i \leftarrow 1$ *to* z **do**
$\quad$ Append $\lceil \log (1/A[i]) \rceil$ to end of S_A;
$\quad$ Append 2^j to end of S_A;
$\quad$ $j \leftarrow j + 1$;
$\quad$ **if** $j > \lceil \log n \rceil$ **then**
$\quad\quad$ $j \leftarrow 0$
$\quad$ **end**
end
return S_A

We now analyze the quality of the range finding solution produced by our RF-construction algorithm. We begin a useful helper lemma that formalizes the intuitive notion that a contention resolution algorithm is unlikely to succeed if its probability is too far form the optimal value for the participant count.

LEMMA 2.6. *Assume in a given round of a uniform contention resolution algorithm that the $1 < k \leq n$ participants (where $k \in \mathbb{Z}$) each decide to transmit with a probability p such that $p < \frac{1}{k\beta \log n}$ or $p > \frac{\beta \log n}{k}$, for some constant $\beta \geq 1$. It follows that the probability exactly one participant transmits is strictly less than: $\frac{1}{2 \log n}$.*

We can now prove our primary lemma which connects the performance of a contention resolution algorithm to the range finding solution it induces.

LEMMA 2.7. *Let A be a uniform contention resolution algorithm defined for a network of size n that solves the problem in $t_X(n)$ rounds in expectation when the network size is determined by random variable X. Let S_A be the range finding sequence returned by RF-construction run on A. There exists a constant $\alpha \geq 1$ such that S_A solves $(n, \alpha \log \log n)$-range finding in expected time no more than $2t_X(n)$ with respect to $c(X)$.*

Pulling Together the Pieces. Fix some uniform contention resolution algorithm A that solves contention resolution in expected time $t_X(n)$ when run in a network of size n with no collision detection and a participant size determined by X. By Lemma 2.7, there exists a constant $\alpha \geq 1$, such that the range finding sequence S_A constructed by applying RF-construction on A, solves $(n, \alpha \log \log n)$-ranging finding in expected time $T \leq 2t_X(n)$ when ranges are drawn from $c(X)$.

Applying Lemma 2.5 further tells us $T \geq 2^{H(c(X))}/(\alpha \log \log n)$. It then follows that $2t_X(n) \geq 2^{H(c(X))}/(\alpha \log \log n) \Rightarrow t_X(n) = \Omega\left(\frac{2^{H(c(X))}}{\log \log n}\right)$, which proves Theorem 2.4.

2.4 Lower Bound for Collision Detection

We now adapt the techniques used in the preceding section to achieve an entropy-based lower bound for the setting with collision detection. Our goal is to prove the following:

THEOREM 2.8. *Fix a uniform algorithm A for a network of size n. Let $t_X(n)$ be the expected round complexity for A to solve contention resolution on a channel with collision detection and the number of participants determined by random variable X. It follows: $t_X(n) \geq (1/2)H(c(X)) - O(\log \log \log \log n)$.*

For the maximum possible entropy value $H(c(X)) = \log \log n$, this bound asymptotically matches the best known upper bound, from the 1986 work of Willard [23], which requires $O(\log \log n)$ rounds. It also provides an arguably simpler and more intuitive approach than the original lower bound from [23], which deployed a more complex probabilistic counting argument to establish $\log \log n - O(1)$ rounds as being necessary for uniform algorithms.

We note that the appearance of a quadruple logarithm is unusual, but straightforward to explain in our context. In the argument that follows we seek a probability within a factor of $1/\log \log n$ from the optimal probability for the current participant size. This is a log factor closer than in our argument for no collision detection, as the shorter executions can handle smaller error probabilities. Recall, within our condensed support $L(n)$, each $i \in L(n)$ is associated

with probability 2^{-i}. So if 2^{-i} is the optimal probability, than a range j that is within a distance of $\log \log \log n$ from i will yield a probability within a factor of $1/\log \log n$; e.g.,

$$2^{-j} = 2^{-(i+\log \log \log n)} = 1/(2^i 2^{\log \log \log n})) = 1/(2^i \log \log n).$$

In the coding scheme used below, as in the no collision detection argument, a code contains a distance value from 0 to $O(\log \log \log n)$, which requires $O(\log \log \log \log n)$ bits.

Our proof follows the same structure as in the no collision detection case. We differ, however, in the details of how we construct our range finding solution and bound such solutions form an information theory perspective.

Bounding Tree Range Finding Using Entropy. Assume T is a binary tree that solves $(n, \alpha \log \log \log n)$-range finding in expected time T with respect to some distribution Y defined over $L(n)$. By interpreting T as a code, we can deploy Shannon's source coding theorem to arrive at the following bounds:

LEMMA 2.9. *For some constant $\alpha \geq 1$ and network size $n > 1$, let T be a labeled binary tree that solves $(n, \alpha \log \log \log n)$-range finding. Assume the target range be determined by random variable Y. Let Z be the random variable describing the complexity of solving range finding using T with respect to Y. It follows:*

$$E(Z) \geq H(Y) - O(\log \log \log n),$$

where H is the entropy function.

Solving Range Finding with Contention Resolution. Here we transform a solution to contention resolution with collision detection to a binary tree that solves range finding with a related expected complexity. We begin by defining an algorithmic process for transforming a uniform algorithm A into a range finding tree T_A.

As a uniform contention resolution algorithm that assumes collision detection, A is formalized as a function that maps the sequence of collisions and silences detected so far to probabilities for all participants to use during the next round. A history of length r can be captured by a bit sequence $b_1 b_2 \ldots b_r$, where $b_i = 0$ means silence was detected in round i and $b_i = 1$ means a collision was detected.

We can interpret A as a binary tree where each node is labeled with a probability. In particular, interpret each input string $s = b_1 b_2 \ldots b_r$ as specifying a particular node at depth $r - 1$, reached in a r-step traversal starting from the root, where at step i you descend the to left sub-tree if $b_i = 0$ and descend to the right sub-tree if $b_i = 1$. You labeled this node with probability $A(s)$; that is, the probability mapped to pattern s by the algorithm.

Let T_1 be this binary tree labeled with probabilities. We next create tree T_2 by replacing each label ℓ in T_1 with its related range: $\lceil \log(1/\ell) \rceil$, as in our no collision detection construction. Let T^* be the canonical binary tree of depth $\lceil \log \log n \rceil$ labeled with all the values in $L(n)$. To arrive at our final range finding solution T_A, we must insert T^* into the tree, with the root beginning at depth $\lceil \log \log n \rceil$. There are many equally useful ways to do so. Assume for now that we just follow the left-most path through T_2, and when arrive at node v depth $\lceil \log \log n \rceil$, we remove v's children and instead make the root of T^* the only child of v.

We now analyze the quality of the range finding tree T_A produced by our above procedure. We begin with a useful probability observation.

LEMMA 2.10. *Assume in a given round of a uniform contention resolution algorithm that the $1 < k \leq n$ (where $k \in \mathbb{Z}$) participants each decide to transmit with a probability p such that $p < \frac{1}{\beta(\log \log n)k}$ or $p > \frac{\beta(\log \log n)}{k}$, for some sufficiently large constant $\beta \geq 1$. It follows that the probability exactly one participant transmits is strictly less than: $\frac{1}{2 \log \log n}$.*

We now make our main argument analyzing T_A's performance as a solution to range-finding.

LEMMA 2.11. *Let A be a uniform contention resolution algorithm defined for a network of size n with collision detection, that solves the problems in $t_X(n)$ rounds in expectation when the network size is determined by random variable X Let T_A be the range finding tree returned by apply our above procedure to A. There exists a constant $\alpha \geq 1$ such that T_A solves $(n, \alpha \log \log \log n)$-range finding in expected time no more than $2t_X(n)$ with respect to $c(X)$.*

Pulling Together the Pieces. Fix some uniform contention resolution algorithm A that solves contention resolution in expected time $t_X(n)$ when run in a network of size n with collision detection and a participant size determined by X. By Lemma 2.11, there exists a constant $\alpha \geq 1$, such that the range finding tree T_A constructed by applying our procedure to A, solves $(n, \alpha \log \log \log n)$-ranging finding in expected time $T \leq 2t_X(n)$ when ranges are drawn from $c(X)$. Applying Lemma 2.9 further tells us that $T \geq H(c(X)) - O(\log \log \log n)$. It follows that $2t_X(n) \geq H(c(X)) - O(\log \log \log n) \Rightarrow t_X(n) = (1/2)H(c(X)) - O(\log \log \log n)$, which proves Theorem 2.8.

2.5 Upper Bound for No Collision Detection

In this section we introduce an algorithm for solving contention resolution without collision detection. We assume the algorithm is provided as input the definition of a random variable Y defined over network sizes. Let X be the actual random variable from which the sizes will be drawn. We will produce our round complexity bounds with respect to the statistical divergence between Y and X, quantifying the cost of inaccuracy in predictions. Again, all omitted proofs can be found in the full version of this paper [1].

In Section 2.3, we proved that with accurate predictions (i.e., $Y = X$), $\Omega(2^{H(c(Y))}/\log \log n)$ rounds are needed in expectation to solve contention resolution. Our goal here is produce a result that comes close to matching this exponential bound. To do so, we analyze a natural strategy: trying range predictions in $c(Y)$ in decreasing order of likelihood. We prove that with constant probability, this strategy solves the problem in 2^T rounds, where $T = 2H(c(X)) + 2D_{KL}(c(X)\|c(Y))$. If every probability used by Y is within some bounded constant factor of the corresponding probability in X, this reduces to $2^{2H(c(Y))}$, which is within the same general form as the lower bound but with an extra factor in the exponent. It is not obvious how to remove any extra factor in the exponent. Indeed, as elaborated in Section 1, we have reason to believe that some exponential factor greater than 1 is necessary for this algorithm.

2.5.1 Algorithm. Let $\pi = \langle \pi_1, \ldots, \pi_{\log n} \rangle$ represent an ordering over $L(n)$ sorted by non-decreasing probability of the corresponding range with respect to $c(Y)$. In other words such that for all $i < j$, $\Pr(k \in (2^{\pi_i - 1}, 2^{\pi_i}] \geq \Pr(k \in (2^{\pi_j - 1}, 2^{\pi_j}]$. Our algorithm consists of $\log n$ rounds where in round i each node broadcasts with probability $1/2^{\pi_i}$.[5]

2.5.2 Analysis. We prove the following time complexity statement regarding the above algorithm.

THEOREM 2.12. *In the above algorithm, a node broadcasts alone after at most $O(2^T)$ rounds where $T = 2H(c(X)) + 2D_{KL}(c(X)\|c(Y))$ rounds with probability at least $1/16$, where X is the actual distribution over the network sizes and Y is the provided distribution.*

First we prove that in round i that if $k \in (2^{\pi_i - 1}, 2^{\pi_i}]$ then a single node broadcasts alone during the round with probability at least $1/2$.

LEMMA 2.13. *For $k \geq 2$ and $i > 0$ if $k \in (2^{\pi_i - 1}, 2^{\pi_i}]$, a single node broadcasts alone in round i with probability at least $1/8$.*

Consider an optimal variable-length code f over the values of $L(n)$ based on their probabilities according to $c(Y)$. We will relate this code to the number of rounds required by our algorithm.

LEMMA 2.14. *With at least probability $1/8$ the above algorithm takes 2^{S+1} rounds where S is the random variable representing the code lengths of f applied to $c(Y)$.*

Note that since $D_{KL}(c(X)\|c(X)) = 0$, if instead our algorithm learns from the actual distribution $c(X)$ we can bound the running time with respect to the entropy of $c(X)$ alone.

COROLLARY 2.15. *(of Theorem 2.12.) In the above algorithm, a single node broadcasts alone after at most $O(2^{2H(c(X))})$ rounds with probability at least $1/16$, where X is both the actual distribution over the network sizes and the distribution provided to the algorithm.*

2.6 Upper Bound for Collision Detection

We now turn our attention to a setting with collision detection. We once again make use of an optimal code f constructed for source $c(Y)$, and use its code words to structure our algorithm's behavior.

Algorithm. We group the values from $L(n)$ into x equivalence classes based on the length of their code according to $f(c(Y))$. Let $\pi_1, \ldots, \pi_x$ be these classes where the ith class contains all values from $c(Y)$ which have codes of length exactly i. More formally, for all $i \in [x]$, let $\pi_i = \{j \in L(n) \mid |f(j)| = i\}$. Note that $x \leq \log \log n$ since we can assign a unique code of length $\log \log n$ bits to all ranges, so the existence of larger codes would contradict the assumed optimality of f.

Our algorithm then divides rounds into x phases, one dedicated to each π_i. In the phase for π_i, we use transmissions and collision detection to perform a binary search over the possible network size ranges represented by the values in class π_i. The binary search algorithm we use is an adaptation of the classical strategy presented

in [23] which searches over a collection of $\lceil \log n \rceil$ geometrically distributed network size guesses, transmitting with a corresponding probability for each guess, and using collision and silence to indicate if a guess is too small or too large, respectively.

More formally, when searching over π_i, the nodes order the ranges in π_i from smallest to largest. They then broadcast with probability 2^{-m}, where m is the median of these values. If a collision is detected, the nodes then recurse over the values greater than m. Otherwise, if silence is detected, they recurse over the values smaller than m. If a single node broadcasts, then contention resolution is solved.

The algorithm proceeds through the phases in order of the classes; i.e., π_1 then π_2, and so on. If the problem is not solved during the search for class π_i, then it moves on to search π_{i+1}. As without our no collision detection algorithm, we present this result here as a one-shot attempt that solves contention resolution with constant probability. For higher probability, it can be repeated, but we do not analyze this form.

Analysis. Our goal is to prove the following about the time complexity of our algorithm.

THEOREM 2.16. *In the above algorithm, with constant probability: contention resolution is solved in $O((H(c(X)) + D_{KL}(c(X)\|c(Y)))^2)$ rounds, where X corresponds to the actual distribution over the network sizes and Y is the distribution provided to the algorithm.*

We first analyze our algorithm with respect to the random variable describing the code lengths generated by f when symbols are drawn from $c(Y)$.

LEMMA 2.17. *With constant probability: the algorithm solves contention resolution in $O(S^2)$ rounds where S is the random variable describing the code lengths of f applied to $c(Y)$.*

We now prove our final result by leveraging the optimality of the code to express the expected code length with respect to the entropy of $c(Y)$.

PROOF. (of Theorem 2.16) We have from Lemma 2.17 that the algorithm solves contention resolution with constant probability in $O(S^2)$ rounds where S is the random variable describing the length of the code word for the target value drawn from $c(Y)$. We have from our optimal code f and Theorem 2.3 that the expected value of S is at most $H(c(X)) + D_{KL}(c(X)\|c(Y)) + 1$. Therefore from Markov's inequality we have that $S \leq 2(H(c(X)) + D_{KL}(c(X)\|c(Y)) + 1)$ with probability at least $1/2$.

To obtain our final result, we multiply the probability that S is not larger than the bound from the theorem with the probability that we solve contention resolution in $O(S^2)$ rounds. Both values are constant, providing the claimed constant success probability. ∎

As with Theorem 2.12, because $D_{KL}(c(X)\|c(X)) = 0$, if $Y = X$, then we get a sharper bound:

COROLLARY 2.18. *(of Theorem 2.16.) In the above algorithm, if the input distribution Y equals the actual network size distribution X: a single node broadcasts alone after at most $O(H^2(c(X)))$ rounds.*

---

[5]Because we seek only a constant probability result, we analyze here only one pass through all $\log n$ possible probabilities. In the pursuit of good expected times, you would instead cycle through these probabilities in a clever manner. We do not prove expectation bounds on this algorithm here, and note that an expectation close to our constant probability result is not necessarily easily obtained.

3 CONTENTION RESOLUTION WITH PERFECT ADVICE

In the previous section, we studied how to speed up contention resolution given probabilistic predictions on the network size. We proved bounds with respect to the *quality* of the predictions. Here we investigate bounds on speed up proved with respect to the *size* of predictions. More concretely, we assume a general setting, in which some abstract learning model provides b bits of predictive advice to the algorithm. Our goal is to understand for a given advice size b, the theoretical limit on the speed up possible given this much information. That is, given the best possible advice of this size, how much improvement is possible?

For $b = 0$, the worst case lower bounds apply. For $b \geq \log n$, it is possible to encode the id of a participant in b bits, enabling a solution in only 1 round. We study the space between these extremes, providing tight bounds for both deterministic and randomized algorithms. The goal with these bounds is to set a ceiling on how much improvement we can ever expect from a given specific learning model with bounded advice size.

3.1 The Perfect Advice Model

We study the *perfect advice model*, a generalization of the standard contention resolution model, parameterized with an advice size $b \geq 0$, in which we augment a contention resolution algorithm A with an *advice function*, $f_A : \mathbb{P}(A) \to \{0, 1\}^b$. At the beginning of each execution, after the adversary selects the set $P \subseteq V$ of players to participate, each player in P is provided the same b bits of advice $f_A(P)$. The advice function, in other words, has perfect knowledge of the participants for the current execution, allowing it generate the best possible prediction for the given size bound.

3.2 Tight Bounds for Deterministic Algorithms

An interesting place to start this investigation of perfect advice is with deterministic algorithms, as these allow us to study the impact of b bits of advice from a purely combinatorial perspective; e.g., asking how many relevant participant sets can be eliminated given a specific advice string?

With no advice (i.e., $b = 0$), deterministic collision detection requires $\Theta(n)$ rounds without collision detection and $\Theta(\log n)$ rounds with collision detection [19]. We prove that with $b = \alpha \log n$ bits of advice, for positive $\alpha < 1$, the problem still requires at least $n^{1-\alpha}/2$ rounds to solve without collision detection. With collision detection, b bits of advice can improve the bound at best to $\log n - b$ rounds. We show both bounds to be tight within small constant factors.

Our strategy is to first bound a harder problem that we call *non-interactive contention resolution* that forces nodes to decide whether or not to transmit in a single round, based only on their advice. Leveraging existing theory on a combinatorial object known as a strongly selective family [6], we formalize the intuitive result that very close to $\log n$ bits of advice are needed to solve this problem deterministically. That is, there is no strategy much better than simply having the advice function specify exactly which single participant should transmit. We then leverage this result as a foundation for our lower bounds on the round complexity of normal contention resolution with and without collision detection. These bounds use efficient contention resolution algorithms to construct efficient solutions to non-interactive resolution.

Non-Interactive Contention Resolution. We say an algorithm A and advice function f_A solves $b(n)$-*non-interactive contention resolution*, if for every participant set $P \subseteq V$, the advice $f_A(P)$ contains no more than $b(n)$ bits and leads to exactly one participant from P transmitting in the first round.

Our goal is to show that this requires $b(n) = \Omega(\log n)$. To do so, we leverage bounds on the following object:

DEFINITION 3.1. *Fix integers n, k, $1 \leq k \leq n$. A family $\mathbb{F}$ of subsets if $[n]$ is (n, k)-strongly selective if for every subset Z of $[n]$ such that $|Z| \leq k$ and for every element $z \in Z$, there is a set F in $\mathbb{F}$ such that $Z \cap F = \{z\}$.*

As proved in [6], for large k, there are no small strongly selective families:

THEOREM 3.2 (FROM [6]). *Let $\mathbb{F}$ be an (n, k)-strongly selective family. If $k \geq \sqrt{2n}$ then it holds that $|\mathbb{F}| \geq n$.*

We can leverage this fact to establish our lower bound on non-interactive contention resolution.

THEOREM 3.3. *Assume deterministic algorithm A and advice function f_A solve $b(n)$-non-interactive contention resolution in a network of size n. It follows that $b(n) \geq \log n$.*

PROOF. Let S contain all $2^{b(n)}$ possible advice strings. For each $s \in S$, let $V(s)$ be the nodes in P that would broadcast according to A given advice s. By the assumption that A is correct, for each $P \subseteq V$, with corresponding advice $s_P = f_A(P)$: $|V(s_P) \cap P| = 1$. We can therefore consider $\mathbb{F} = \{V(s) \mid s \in S\}$ to be an (n, n)-strongly selective family. It follows from Theorem 3.2 that $|\mathbb{F}| \geq n$, which implies $|S| \geq n$, which implies the number of bits required to encode elements of S is at least $\log n$, as claimed by the theorem. □

Bounds for No Collision Detection. Here we build on Theorem 3.3 to prove a tight bound on the advice required for deterministic solutions to the contention resolution problem. The problem can be trivially solved in one round given $\log n$ bits of advice. We show that as we reduce this advice to $\alpha \log n$ bits, for some positive fraction $\alpha < 1$, the rounds required grow roughly as $n^{1-\alpha}$. We will then show this result is tight within constant factors.

THEOREM 3.4. *Assume deterministic algorithm A and advice function f solve contention resolution in $t(n)$ rounds in a network of size n with no collision detection, such that f returns at most $\alpha \log n$ bits of advice, for some positive constant $\alpha < 1$. It follows that $t(n) \geq n^{1-\alpha}/2$.*

PROOF. Fix some A, f, t, n, and α, as specified. Our strategy is to deploy the advice function f, designed for general contention resolution without collision detection, to create a related advice function f' for the non-interactive setting. To do so, given a participant set $P \subseteq V$, consider the execution that results when the participants in P run A given initial advice $f(P)$. By assumption, this execution solves contention resolution by round $r \leq t(n)$. Crucially, because A assumes no collision detection, no participant in this execution receives or detects anything until round r. Therefore,

each participant can simulate the first $r - 1$ rounds locally by simply simulating A detecting silence for $r - 1$ rounds.

It follows that to solve non-interaction contention resolution it is enough for $f'(P)$ return the pair $(f(P), r)$, allowing each node in P to locally simulate A with advice $f(P)$ through the first $r - 1$ rounds, and then simply execute round r, solving contention resolution in one round without any interaction. To encode this advice requires f' to return $f(P) + \lceil \log t(n) \rceil = \alpha \log n + \lceil \log t(n) \rceil$ bits. By Theorem 3.3, this sum must be at least $\log n$. It follows:

$$
\begin{aligned}
\alpha \log n + \lceil \log(t(n)) \rceil &\geq & \log n \Rightarrow \\
\lceil \log(t(n)) \rceil &\geq & \log n - \alpha \log n \Rightarrow \\
\log(t(n)) + 1 &\geq & (1 - \alpha) \log n \Rightarrow \\
2^{\log(t(n))} 2 &\geq & 2^{(1-\alpha) \log n} \Rightarrow \\
2t(n) &\geq & n^{1-\alpha} \Rightarrow \\
t(n) &\geq & n^{1-\alpha}/2,
\end{aligned}
$$

as claimed by our theorem. □

It is straightforward to show that this lower bound is tight (within constant factors). Assume we are restricted to $q = \alpha \log n$ bits of advice for some positive $\alpha < 1$. Assign the n possible participants to the leaves of a balanced binary tree of height $\lceil \log n \rceil$. Given $\log n$ bits of advice, it is possible to encode a traversal through this tree from the root down to a specific leaf, corresponding the id of a participant. For $q < \log n$ bits, we can record the first q steps of such a traversal toward a participant. Given this traversal prefix, all participants can reduce the number of possible identities of the target participant by a factor of 2 with each step deeper into the tree. The result is that only $n/q = n/2^{\alpha \log n} = n/n^{\alpha} = n^{1-\alpha}$ possible identities remain. The participants can solve the problem in an additional $n^{1-\alpha}$ rounds by giving each of these remaining identities one round to transmit alone. The resulting algorithm is within a constant factor of the bound proved above.

Bounds for Collision Detection. We now consider the collision detection case. As mentioned, with no advice, $\Theta(\log n)$ is a tight bound on contention resolution with collision detection. Here we prove that b bits of advice can improve this by at most an additive factor of b.

THEOREM 3.5. *Assume deterministic algorithm A and advice function f solve contention resolution in $t(n)$ rounds in a network of size n with collision detection, such that f returns at most $b(n)$ bits of advice. It follows that $t(n) \geq \log n - b(n)$.*

PROOF. Fix some A, f, t, b and n as specified. We use these tools to build an advice function f' to solve non-interactive contention resolution. In particular, given a participant set $P \subseteq V$, $f'(P)$ can include the advice $f(P)$ as well as the round $r \leq t(n)$ during which these participants running A with advice $f(P)$ solve contention resolution. Unlike in the no collision detection case, however, this advice is not sufficient on its own for participants to simulate A's behavior during round r, as the participants also need to know the *pattern* of collisions and no collisions that occur in the first $r - 1$ rounds of this simulation.

Accordingly, we have $f'(P)$ instead return $f(P)$ plus a bit string of length $r - 1$ that encodes the collision history (e.g., with 0=silence and 1=collision) of the first $r - 1$ rounds of participants in P running A with this advice. This allows each participant to locally simulate these $r - 1$ rounds and proceed with the behavior of round r which solves the non-interactive problem.

We know from Theorem 3.3 that solving non-interactive contention resolution requires at least $\log n$ bits of advice. Therefore, it must be the case that

$$
b(n) + t(n) \geq \log n \Rightarrow t(n) \geq \log n - b(n),
$$

as claimed. □

To show this bound tight, we first note that a standard solution for contention resolution with collision detection is to construct a balanced binary tree of height $\lceil \log n \rceil$ with one potential participant assigned to each leaf. Using the collision detector, participants can traverse from the root to the leaf corresponding to one of the active participants (using silence/collision to indicate a vote for descending to the left/right subtree). An obvious way to deploy $b(n)$ bits of advice to augment this algorithm is to provide the participants the first $b(n)$ steps of the traversal toward an active participant, leaving them with only $\lceil \log n \rceil - b(n)$ steps remaining to complete the identification of a single node to transmit. Resolving the impact of the ceilings provides an almost exactly matching upper bound of $\log n - b(n) + 1$ rounds.

3.3 Tight Bounds for Randomized Algorithms

We now consider tight bounds on the maximum possible speed up to the expected time complexity of randomized algorithms given a bounded amount of advice. These results deploy two styles of reduction arguments. An interesting property of the lower bound results that follow is that because we ultimately reduce to a lower bound that holds for non-uniform algorithms (from [19]), these new bounds also hold for non-uniform algorithms. We prove both of the below theorems in the full version of this paper [1].

THEOREM 3.6. *Fix a network size n and advice size $b(n) < \log n$. Let $t(n)$ be the smallest bound such that an algorithm augmented with an advice function that provides $b(n)$ bits of advice can solve contention resolution with no collision detection in $t(n)$ rounds in expectation. It follows that $t(n) = \Theta(\log(n)/2^{b(n)})$.*

THEOREM 3.7. *Fix a network size n and advice size $b(n) < \log \log n$. Let $t(n)$ be the smallest bound such that an algorithm augmented with an advice function that provides $b(n)$ bits of advice can solve contention resolution with collision detection in $t(n)$ rounds in expectation. It follows that $t(n) = \Theta(\log \log n - b(n))$.*

ACKNOWLEDGMENTS

REFERENCES

[1] 2021. Full version available at https://arxiv.org/abs/2105.12706.
[2] Yagel Ashkenazi, Ran Gelles, and Amir Leshem. 2020. Brief Announcement: Noisy Beeping Networks. In *PODC '20: ACM Symposium on Principles of Distributed Computing, Virtual Event, Italy, August 3-7, 2020*. ACM.
[3] Reuven Bar-Yehuda, Oded Goldreich, and Alon Itai. 1992. On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization. *J. Comput. System Sci.* 45, 1 (1992), 104–126.
[4] Gewu Bu, Maria Potop-Butucaru, and Mikael Rabie. 2020. Wireless Broadcast with Short Labels. In *Proceedings of the Conference on Networked Systems (NetSys 2020)*.
[5] Imrich Chlamtac and Shai Kutten. 1985. On Broadcasting in Radio Networks - Problem Analysis and Protocol Design. *IEEE Transactions on Communications* 33, 12 (1985), 1240–1246.
[6] Andrea EF Clementi, Angelo Monti, and Riccardo Silvestri. 2003. Distributed broadcast in radio networks of unknown topology. *Theoretical Computer Science* 302, 1-3 (2003), 337–364.
[7] Thomas M. Cover and Joy A. Thomas. 2006. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA.
[8] Fabien Dufoulon, Janna Burman, and Joffroy Beauquier. 2020. Can Uncoordinated Beeps tell Stories?. In *PODC '20: ACM Symposium on Principles of Distributed Computing, Virtual Event, Italy, August 3-7, 2020*. ACM.
[9] Klim Efremenko, Gillat Kol, and Raghuvansh R. Saxena. 2020. Noisy Beeps. In *PODC '20: ACM Symposium on Principles of Distributed Computing, Virtual Event, Italy, August 3-7, 2020*. ACM.
[10] Faith Ellen and Seth Gilbert. 2020. Constant-Length Labeling Schemes for Faster Deterministic Radio Broadcast. In *Proceedings of the 32nd ACM on Symposium on Parallelism in Algorithms and Architectures (SPAA)*. ACM.
[11] Faith Ellen, Barun Gorain, Avery Miller, and Andrzej Pelc. 2019. Constant-Length Labeling Schemes for Deterministic Radio Broadcast. In *Proceedings of the 31st ACM on Symposium on Parallelism in Algorithms and Architectures (SPAA)*. ACM, 171–178.
[12] Martín Farach-Colton, Rohan J Fernandes, and Miguel A Mosteiro. 2006. Lower bounds for clear transmissions in radio networks. In *Latin American Symposium on Theoretical Informatics*. Springer, 447–454.
[13] Pierre Fraigniaud, David Ilcinkas, and Andrzej Pelc. 2010. Communication algorithms with advice. *J. Comput. Syst. Sci.* 76, 3-4 (2010), 222–232.
[14] David Ilcinkas, Dariusz R. Kowalski, and Andrzej Pelc. 2010. Fast radio broadcasting with advice. *Theor. Comput. Sci.* 411, 14-15 (2010), 1544–1557.
[15] Tomasz Jurdziński and Grzegorz Stachowiak. 2002. Probabilistic algorithms for the wakeup problem in single-hop radio networks. In *International Symposium on Algorithms and Computation*. Springer, 535–549.
[16] Silvio Lattanzi, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. 2020. Online Scheduling via Learned Weights. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*. SIAM.
[17] Michael Mitzenmacher. 2018. A Model for Learned Bloom Filters and Optimizing by Sandwiching. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*.
[18] Michael Mitzenmacher and Sergei Vassilvitskii. 2020. Algorithms with predictions. *arXiv preprint arXiv:2006.09123* (2020).
[19] Calvin Newport. 2014. Radio network lower bounds made easy. In *International Symposium on Distributed Computing*. Springer, 258–272.
[20] John O Pliam. 2000. On the incomparability of entropy and marginal guesswork in brute-force attacks. In *International Conference on Cryptology in India*. Springer, 67–79.
[21] Manish Purohit, Zoya Svitkina, and Ravi Kumar. 2018. Improving Online Algorithms via ML Predictions. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*.
[22] Claude E Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal* 27, 3 (1948), 379–423.
[23] Dan E Willard. 1986. Log-logarithmic selection resolution protocols in a multiple access channel. *SIAM J. Comput.* 15, 2 (1986), 468–477.