Brief Announcement: Resource Competitive Broadcast against Adaptive Adversary in Multi-channel Radio Networks

Haimin Chen
State Key Laboratory for Novel Software Technology
Nanjing University
haimin.chen@smail.nju.edu.cn

Chaodong Zheng
State Key Laboratory for Novel Software Technology
Nanjing University
chaodong@nju.edu.cn

ABSTRACT

Broadcasting in wireless networks is vulnerable to adversarial jamming. To thwart such behavior, researchers have proposed resource competitive analysis. In this framework, sending, listening, or jamming on one channel for one time slot costs one unit of energy. The adversary can employ arbitrary strategy to disrupt communication, but has a limited energy budget \( T \). The honest nodes, on the other hand, aim to accomplish broadcast while spending only roughly \( O(T/C) \) time, while spending only roughly \( \tilde{O}(\sqrt{T/n}) \) energy. However, these algorithms only work for \( C = O(n) \), and can only tolerate an oblivious adversary. We improve the result by considering an adaptive adversary and arbitrary values of \( n \) and \( C \). In our algorithms, for large \( T \) values, each node’s runtime is \( O(T/C) \), and each node’s energy cost is \( \tilde{O}(\sqrt{T/n}) \). The time complexity is asymptotically optimal, while the energy complexity is near optimal in some cases. We use “epidemic broadcast” with proper working probabilities to achieve time efficiency and resource competitiveness, and leverage coupling arguments in the analysis to handle the adaptivity of the adversary.

CCS CONCEPTS

• Theory of computation → Distributed algorithms.

KEYWORDS

Broadcast, radio network, resource competitive algorithm, coupling.

ACM Reference Format:
Haimin Chen and Chaodong Zheng. 2020. Brief Announcement: Resource Competitive Broadcast against Adaptive Adversary in Multi-channel Radio Networks. In ACM Symposium on Principles of Distributed Computing (PODC ’20), August 3–7, 2020, Virtual Event, Italy.
ACM, New York, NY, USA, 3 pages.
https://doi.org/10.1145/3382734.3405697

1 INTRODUCTION

Consider a synchronous, time-slotted, single-hop wireless network consisting of \( n \) nodes. These nodes communicate over a shared wireless medium containing \( C \) channels. In each time slot, each node can operate on one arbitrary channel, but cannot send and listen simultaneously. We are interested in a fundamental communication problem—broadcasting—in which a designated source node wants to disseminate a message \( m \) to all other nodes in the network.

Lots of modern lightweight wireless devices are battery powered, and are able to switch between active and sleep states. Often, sending and listening occurring during active state dominate the energy expenditure, while sleeping costs much less [7]. On the other hand, the open and shared nature of wireless medium allows malicious users to disrupt communication via jamming. Such denial-of-service attacks could quickly deplete nodes’ energy, putting an end to the normal operation of the network [4]. Nonetheless, injecting interfering signals also incurs operational cost.

Above observations motivate the notion of resource competitive algorithms (see, e.g., [1–3, 5, 6]) which focus on optimizing relative cost. More specifically, assume there is a jamming adversary called Eve. She can jam multiple channels in each slot, and jamming one channel for one slot costs one unit of energy. For each node, the cost of sending or listening on one channel for one slot is also one unit of energy, while idling is free. Eve has an energy budget \( T \) that is unknown to the nodes, and can employ arbitrary strategy to stop nodes from accomplishing the distributed computing task in concern. Now, the central question is, can we design algorithms (for, e.g., broadcasting) that ensure each node’s cost is only \( O(T) \)? Interestingly enough, the answer is positive. In particular, Gilbert et al. [3] present a resource competitive broadcast algorithm in the single-channel radio network setting: with high probability, all nodes receive the message and terminate within \( O(T + n) \) time slots, and each node only incurs a cost of \( \tilde{O}(\sqrt{T/n} + 1) \).

This algorithm works even when Eve is adaptive and \( n \) is unknown. Later, Chen and Zheng [2] consider the multi-channel setting and show that when Eve is oblivious and \( C = O(n) \), having multiple channels allows a linear speedup in time complexity, while the cost remains to be \( \tilde{O}(\sqrt{T/n} + 1) \).

In this paper, we briefly introduce two new multi-channel broadcast algorithms that can tolerate an adaptive adversary and work for arbitrary \( n \), \( C \) values, without sacrificing time efficiency and resource competitiveness. The first algorithm—\textsc{MultiCastAdp}—requires \( n \) as an input parameter; while the other more complicated one—\textsc{MultiCastAdp}—works even when \( n \) is unknown. In both algorithms, if \( T \) is large when compared with \( n \) and \( C \), each node’s runtime is \( O(T/C) \), and each node’s energy cost is \( \tilde{O}(\sqrt{T/n}) \). Note that the \( O(T/C) \) time complexity is asymptotically optimal, as

\[ \frac{m}{n} \geq c \]
Eve can jam all C channels continuously for $T/C$ slots. Meanwhile, the $O(\sqrt{T/n})$ term in cost matches existing lower bound [3] up to poly-logarithmic factors, at least when $C = O(1)$.

**Theorem 1.1.** *MultiCastAdp* guarantees the following properties w.h.p.: (a) all nodes receive the message and halt within $O(T/C + \max(n/C, C/n))$ slots; and (b) each node costs $O(\sqrt{T/n} + n/C)$. 

**Theorem 1.2.** *MultiCastAdvAdp* guarantees the following properties w.h.p.: (a) all nodes receive the message and halt within $O(T/C + \max(n/C, C/n))$ slots; and (b) each node costs $O(\sqrt{T/n} + n/C)$. 

Additional model details. All nodes start execution simultaneously and can independently generate random bits. In each slot, each node either sends a message on a channel, or listens on a channel, or remains idle. Only listening nodes get feedback regarding channel status. The adversary Eve is adaptive: at the beginning of each slot, she is given all past execution history and can use this information to determine the channels to jam in the current slot.

In each slot, for each listening node, the channel feedback is determined by the number of sending nodes on that channel and the behavior of Eve. Specifically, consider a slot and a channel $ch$. If no node sends on $ch$ and Eve does not jam $ch$, then nodes listening on $ch$ hear silence. If exactly one node sends a message on $ch$ and Eve does not jam $ch$, then nodes listening on $ch$ receive the unique message. Finally, if at least two nodes send on $ch$ or Eve jams $ch$, then nodes listening on $ch$ hear noise. Note that nodes cannot tell whether noise is due to jamming or message collision (or both).

The definition of resource competitive algorithms is given in [1]:

**Definition 1.3.** Consider an execution $\pi$ in which nodes execute algorithm $A_N$ and Eve employs strategy $A_E$. Let cost$_u(\pi)$ denote the energy cost of node $u$, and $T(\pi)$ denote the cost of Eve. We say $A_N$ is ($\rho, \tau$)-resource competitive if max$_u$ [cost$_u(\pi)] \leq \rho(T(\pi)) + \tau$ for any execution $\pi$ and any strategy $A_E$.

In above, $\rho$ is a function of $T$ and possibly other parameters (such as $n, C$). It captures the additional cost nodes incur due to jamming. The other function $\tau$ captures the cost of the algorithm when Eve is absent, thus $\tau$ should not depend on $T$. Most resource competitive algorithms aim to minimize $\rho$, while keeping $\tau$ reasonably small.

2 **THE MULTICASTADP ALGORITHM**

**Algorithm design.** Most resource competitive broadcast algorithms divide slots into consecutive epochs, and execute a jamming-resistant broadcast scheme within each epoch. In the single-channel setting, often the core idea of the scheme is to broadcast "sparingly" [3, 5, 6]. Consider 1-to-1 communication as an example. If both nodes send and listen in $\Theta(\sqrt{R})$ random slots in an epoch containing $R$ slots, then by a birthday-paradox-like argument, successful transmission is likely to occur even if Eve jams some constant fraction of all $R$ slots. On the other hand, in the multi-channel setting, epidemic broadcast is employed [2]. In the simplest form of this scheme, in each time slot, each node will randomly choose a channel from $[C]$. Then, each informed node (i.e., the node knows the message) will broadcast the message with a constant probability, while each uninformed node will listen with a constant probability. If $C = n/2$, without jamming, all nodes will be informed in $O(\log n)$ slots w.h.p. In fact, this scheme is also competitive against jamming: Eve cannot stop message dissemination even if she jams some constant fraction of $C$ channels for some constant fraction of $R$ slots.

In designing *MultiCastAdp*, one key challenge is to extend the basic epidemic broadcast scheme to enforce optimal runtime $O(T/C)$ for arbitrary $n, C$ values. To that end, recall that in the single-channel setting, [3] has shown that $\Theta(1/\sqrt{R})$ is roughly an optimal working probability (i.e., sending/listening probabilities). When $C$ channels are available, we observe that a good way to adjust the probability would be to multiply it by $\sqrt{C}$ (i.e., $\Theta(\sqrt{C}/\sqrt{R})$).

Intuitively, the reason being: if each node works on $\sqrt{C}$ random channels simultaneously in each slot, then again by a birthday-paradox-like argument, each pair of nodes will meet on at least one channel with at least constant probability, which effectively means the optimal single-channel analysis would work again. Of course nodes cannot work on multiple channels simultaneously, but over a period of time, multiplying the single-channel working probability by $\sqrt{C}$ achieves a similar effect. Besides, although the working probability of nodes is increased by a factor of $\sqrt{C}$, the energy expenditure of Eve will increase by a factor of $\Theta(C)$. As a result, compared with single-channel algorithms, *MultiCastAdp* has a $\Theta(C)$ speedup in time, with each node’s cost unchanged.

Handling adaptivity in analysis. Termination detection is another key integrant of resource competitive algorithms. For each node $u$, a common and useful termination criterion is comparing $N_u$—the number of silent slots it observed during the current epoch—to some pre-defined threshold. Thus, we often need to show $N_u$ is close to its expected value, which is non-trivial if Eve is adaptive.

**To see this, let $G_i$ denote the behavior (i.e., channels choices and actions) of all nodes in the $i$th slot, and define $Q_i$—the set of channels that are not jammed by Eve—as the jamming result of the $i$th slot. Notice that $N_u$ can be written as the sum of $R$ indicator random variables: $N_u = \sum_{j=1}^{R} N_{u,i}$, where $N_{u,i} = 1$ iff $i$ hears silence in the $i$th slot. In general, $N_{u,i}$ is determined by $G_i$ and $Q_i$, but $Q_i$ can be arbitrary function of $\{G_1, G_2, \ldots, G_{i-1}, Q_1, Q_2, \ldots, Q_{i-1}\}$. Nonetheless, in case Eve is oblivious (i.e., an offline adversary), her optimal strategy would be a fixed vector of jamming results $q = (q_1, q_2, \ldots, q_R)$, thus $\{N_{u,1}, N_{u,2}, \ldots, N_{u,R}\}$ are mutually independent when $\{G_1, G_2, \ldots, G_R\}$ are mutually independent (this can be enforced by the algorithm). As a result, we can directly apply concentration inequalities like Chernoff bounds to show $N_u$ will be close to its expectation. However, when Eve is adaptive, above observations no longer hold, since $Q_i$ may depend on $G_j$ for $j < i$.

To resolve this issue, we apply the coupling technique. Specifically, for each vector of jamming results $q$, we create a coupled execution and relate $N_u$ (with $Q = q$) to a corresponding random variable $Z_u^q$ in the coupled execution. By carefully crafting the coupling, $Z_u^q$ can be interpreted as the sum of a set of independent random variables, allowing us to easily bound the failure probability that $Z_u^q$ deviates a lot from its expectation. However, there is a catch in this approach. To actually bound the probability that $Z_u^q$ deviates a lot from its expectation, we need to sum the failure probability over all vectors of jamming results (i.e., use a union bound). But there are $O(2^C)$ such vectors, and this is too many! Our solution to this problem is to group all vectors into fewer categories, such that vectors within one category have the same or similar effects on...
$N_\eta$ for our algorithms. In the end, we only need to consider $O(R)$ categories and create a coupled execution for each of them.

**Algorithm description.** The $i^{th}$ epoch contains $R_i = \Theta(i^4 \cdot \log^2 n)$ time slots. In each slot within the $i^{th}$ epoch, for each node $u$ that is still active, it will hop to a channel chosen uniformly at random from the $C$ channels, and will choose to broadcast or listen each with probability $p_i = (\sqrt{C/n})^2$. If $u$ decides to broadcast, the content to be broadcast is $m$ if $u$ is informed, and a special beacon message if $u$ is uninformed. Finally, at the end of the $i^{th}$ epoch, $u$ will halt if it heard at least $0.5p_iR_i$ silent slots during this epoch.

3 THE MULTICASTADVADP ALGORITHM

Networks organized in ad-hoc mode without infrastructure often cannot provide $n$ a priori. Our second algorithm deals with such scenario, and its design and analysis are much more involved.

When $n$ is unknown to the nodes, the principal obstacle lies in properly setting nodes’ working probabilities. (Recall the ideal probability is a function of $n$.) In view of this, we adopt the “epoch-phase” structure: MULTICASTADVADP contains multiple epochs, each of which contains multiple phases; for each node, it may use different working probabilities in different phases, but the probability is fixed within one phase. Notice, for each epoch, we need to ensure it contains sufficiently many “good” phases, in the sense that within each such good phase broadcast will succeed if Eve does not heavily jam it. Another challenge posed by the unknown $n$ value is that the simple termination criterion of large fraction of silent slots no longer works, as this can happen when the working probability is too low (thus broadcast will fail and nodes should not stop). Therefore, we also need a new termination mechanism.

Gilbert et al. [3] provide an approach to the above two challenges in the single-channel setting. In their algorithm, nodes adjust their working probabilities based on the amount of jamming “observed” in previous phases. Specifically, at the beginning of an epoch $i$, nodes set their initial working probability to a pre-defined small value. After each phase, each node $u$ will increase its working probability $p_{u, i}$ by a factor of $2^{\max\{0,\eta_u - 0.5\} / i}$, where $\eta_u$ denotes the fraction of silent slots $u$ observed within the phase. This ingenious expression provides two important advantages: (a) Eve has to keep jamming heavily to prevent $p_u$ from reaching the ideal value; and (b) $p_u$ and $p_v$ might be different for any nodes $u$ and $v$, but the difference is bounded. As for termination, the number of messages nodes heard could be a good metric: when the working probabilities are asymptotically smaller than the ideal value, each node hardly hears the message. However, a simple threshold $h$ would not work. In particular, Eve can carefully control its jamming rate so that, say, roughly half of the active nodes hear the message more than $h$ times and halt. In such case, the remaining nodes might need higher working probabilities to create enough messages in later phases, resulting in a non-optimal resource competitive ratio. To resolve this issue, Gilbert et al. develop a two-stage termination mechanism: when a node $u$ hears the message more than $h$ times, it becomes a $helper$ and obtains an estimate of $n$; Later, when $n$ is sure that all nodes have become $helper$, it will halt.

In MULTICASTADVADP, we extend the above approach to the multi-channel setting. Specifically, we observe that the message dissemination scheme used in [3] is relatively slow in that it needs $\Theta(\lg n)$ phases to accomplish broadcast. By contrast, the application of epidemic broadcast reduces this time period to a single weakly-jammed phase. Notice, this replacement is not a simple cut-and-paste. Instead, we have to adjust the phase structure. Each phase now contains two steps: the first is dedicated to message dissemination, while the second checks whether the working probabilities are large enough. This adjustment further demands us to change the way nodes update the probabilities after each phase: $p_u \leftarrow p_u + 2^{\max\{0,\eta_u - \eta_u - 1.5\}}$. In the end, MULTICASTADVADP provides a slightly better resource competitive ratio than [3].

Handing adaptivity in MULTICASTADVADP also becomes more challenging. As mentioned before, we know $\mathbb{E}[\eta_u] = \mathbb{E}[\eta_v]$, and we need $\eta_u$ and $\eta_v$ to be close for any two nodes in each phase (so the accumulated difference between $p_u$ and $p_v$ can be bounded). Therefore we once again need to show $N_\eta$ is very close to its expectation. To acquire the desired bound, the jamming results vectors have to be grouped at a much finer level, which in turn requires the failure probability for each category of vectors to be much lower (otherwise a union bound over the increased number of categories would not work). Larger deviation from expectation further demands the initial working probability nodes used at the beginning of each epoch to be sufficiently high (otherwise $\eta_u$ will deviate too much from its expectation). Unfortunately, this modification could result in nodes becoming $helper$ with incorrect estimates of $n$, violating the correctness of the termination mechanism. We fix this problem by introducing step three into each phase. In particular, nodes’ working probability in step three is different from that of step one and two, and remains unchanged throughout an entire epoch. By observing the fraction of silent slots in step three, nodes can determine the reliability of theirs estimates.

ACKNOWLEDGMENTS

We would like to thank Weiming Feng for the helpful discussions.

This work is supported by the Ministry of Science and Technology of China under Grant No.: 2018YFB1003200, and by the National Natural Science Foundation of China (NSFC) under Grant No.: 61702255 and 61672275.

REFERENCES

[1] Michael Bender, Jeremy Fineman, Mahnush Movahedi, Jared Saia, Varsha Dani, Seth Gilbert, Seth Pettie, and Maxwell Young. 2015. Resource-Competitive Algorithms. SIGACT News 46, 3 (2015), 57–71.
[2] Haimin Chen and Chaodong Zheng. 2019. Fast and Resource Competitive Broadcast in Multi-channel Radio Networks. In Proceedings of the 31st ACM Symposium on Parallelism in Algorithms and Architectures (SPAA ’19). ACM, 179–189.
[3] Seth Gilbert, Valerie King, Seth Pettie, Ely Porat, Jared Saia, and Maxwell Young. 2014. (Near) Optimal Resource-competitive Broadcast with Jamming. In Proceedings of the 26th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA ’14). ACM, 257–266.
[4] Ramakrishna Gummadhi, David Wetherall, Ben Greenstein, and Srinivasan Sehan. 2007. Understanding and Mitigating the Impact of RF Interference on 802.11 Networks. In Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM ’07). ACM, 385–396.
[5] Valerie King, Seth Pettie, Jared Saia, and Maxwell Young. 2018. A resource-competitive jamming defense. Distributed Computing 31, 6 (2018), 419–439.
[6] Valerie King, Jared Saia, and Maxwell Young. 2011. Conflict on a Communication Channel. In Proceedings of the 30th ACM Symposium on Principles of Distributed Computing (PODC ’11). ACM, 277–286.
[7] Joseph Polastre, Robert Szewczyk, and David Culler. 2005. Telos: Enabling Ultra-low Power Wireless Research. In Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN ’05). IEEE, 364–369.