Stable Memoryless Queuing under Contention

Paweł Garncarek
Institute of Computer Science, University of Wroclaw, Poland
pgarn@cs.uni.wroc.pl

Tomasz Jurdziński
Institute of Computer Science, University of Wroclaw, Poland
tju@cs.uni.wroc.pl

Dariusz R. Kowalski
School of Computer and Cyber Sciences, Augusta University, Augusta, USA
SWPS University of Social Sciences and Humanities, Warsaw, Poland
dkowalski@augusta.edu

Abstract

In this work we study stability of local memoryless packet scheduling policies in a distributed system of n nodes/queues under contention. The local policies at nodes may only access their current local queues, and have no other feedback from the underlying distributed system. Moreover, their memory is limited to some basic parameters. The packets arrive at queues according to arrival patterns controlled by an adversary restricted only by injection rate ρ and burstiness b, or driven by a stochastic process; the former model analyzes worst-case stability while the latter – average case. We assume that the underlying distributed system is a classic shared channel, in which no two packets could be successfully scheduled (and removed from queues) at the same time. We show that there is a local memoryless scheduling policy which is both adversarially and stochastically stable for injection rates $\Omega(1/\log n)$. Another algorithm achieves even higher – constant – stable injection rate, but only for a bounded range of burstiness. The first algorithm is utilizing properties of interleaved ultra-selectors, for which we prove stronger properties than known so far, while the second one is based on entirely new concept of selector with thresholds, unlike previously considered binary selectors/codes in the literature.

Note that popular Backoff algorithms, some of which achieve stability for constant (stochastic) injection rates [18], use memory to record current state (e.g., the number of unsuccessful transmissions or the result of random sampling in each window) as well as randomization and feedback from the channel; unlike solutions in this work, which are memoryless and do not rely on randomization or channel feedback (thus, could be used independently from the link layer protocols).

2012 ACM Subject Classification Networks → Packet scheduling; Theory of computation → Online algorithms; Theory of computation → Distributed algorithms

Keywords and phrases packet scheduling, online algorithms, adversarial injections, stochastic injections, stability, memoryless algorithms

Digital Object Identifier 10.4230/LIPIcs.DISC.2019.17

Funding This work was funded by Polish National Science Center grant 2017/25/B/ST6/02010.

1 Introduction

Recently, due to rapidly growing number of devices and popularity of distributed protocols, the impact of congestion on stability of queuing processes has become an important practical and research topic. They are everywhere, often dependent on each other and competing for the same resources (in this paper modeled as a shared channel with contention). Queues are governed by scheduling algorithms, often run locally in a distributed way. The desired property of the whole system of queues is stability – understood as existing of an upper bound on the numbers of packets queued at devices at any time. In this work, we study stability of local memoryless packet scheduling policies in the process of dynamic distributed broadcasting on a shared channel. A shared channel, also called a multiple access channel, is a broadcast network with instantaneous delivery of transmitted messages to every device (also called a node or a station) in the system and a possibility of conflict for access to the transmitting medium. A message sent via a channel by a station is received successfully by all the stations when its transmission has not overlapped with transmissions by other stations. In the queue system in contention, the channel represents the contention and is interpreted as a rule that nothing happens to the queues if at least two of them schedule a stored packet/job to be transmitted/executed at the same time.

The traditional approach to modeling queuing process on a shared channel was through dynamic broadcasting problem and its corresponding queue system, but it could be easily generalized to arbitrary queues with jobs or other types of elements. It has assumed continuous packet injection subject to stochastic constraints (typically, Poisson arrival rates). Recent papers, following the adversarial queuing approach for store-and-forward packet networks, studied stability of the system of queues on a shared channel in adversarial settings. An adversary is determined by two parameters: injection rate ρ, which is the average number of injected packets, and burstiness b, which is the maximum number of packets that may be injected in a round.

We focus on memoryless schedulers, showing that, although very restricted, they are quite powerful in the sense that some of them could guarantee stability of the system for high injection rates. Memoryless schedulers are local policies at nodes, which may only access their current local queues, and have no other feedback from the underlying distributed system. The assumption of limited feedback assures that designed policies are applicable in broad spectrum of scenarios and systems; in particular they could adjust to multi-layer stacks of protocols by assuring independence of a scheduling policy from the actual feedback from the lower layers. Moreover, internal memory of nodes is limited to some basic parameters and $O(\log n)$ control bits per each packet stored (necessary to keep e.g., basic packet informations such as destination), where n is the number of queues in the system. This requirement addressess the need of optimization of resources in contemporary and emerging scenarios, e.g., networks of computationally limited wireless entities run on batteries in IoT applications.

1.1 Our results

This paper studies stability of deterministic local memoryless packet schedulers in the context of distributed broadcasting on a shared channel (as mentioned earlier, it could be generalized to other types of queues with contention). The stability is studied in adversarial setting (corresponding to worst case system behavior), defined in terms of global injection rate ρ and burstiness b, and stochastic setting (corresponding to average case system behavior), in which injections at each node follow the Bernoulli process. We show that there is a local memoryless scheduling policy with relatively small memory (i.e., one bit per packet, indicating whether the packet is old or relatively new), which is stable, both adversarially and stochastically, for injection rates $\Omega(1/\log n)$ on a shared channel. The scheduler is based on interwleaving ultra-selectors defined in [8]. Independently, we show better existential results for ultra-selectors. The second scheduler is based on a different type of selector with thresholds (unlike the previously used binary selectors/codes in the literature), and achieves stability for constant injection rates with bounded values of burstiness. Comparison of our results with the most relevant other recent results is given in Table 1.

1.2 Previous and related work

There is a long history of research on dynamic broadcast on multiple access channels. Early work includes developing and studying properties of protocols like Aloha [1] and binary exponential backoff [24]. Recent study on this topic has focused on scenarios when packets were injected subject to statistical constraints. Stochastic stability has been the basic quality criterion to achieve, understood in the sense that the input and output rates were equal. See the paper by Gallager [13] for an overview of early research; recent work includes the papers of Goldberg et al. [15], Goldberg et al. [16], Håstad et al. [18], Raghavan and Upfal [25], Bender et al. [5] and [22].

Adversarial queuing was introduced by Borodin et al. [7] as a framework to study stability of routing protocols in (point-to-point connected) networks under worst-case traffic scenarios modeled by adversaries. Independently, Andrews et al. [4] defined a greedy protocol to be universally stable when it was stable in all networks for any injection rate $\rho < 1$. This line of research for wired networks has been intensively pursued for many models and protocols.

Chlebus at al. [10] were the first who studied adversarial queuing on a shared channel. They however, similarly to all the follow up work (cf., [3, 2, 9]), assumed that schedulers are embedded into the channel, in the sense that they can receive channel feedback or even attach and read additional information bits. Several results were obtained and new protocols designed and analyzed using this model, however they were derived for stronger schedulers or for restartable schedulers (so called acknowledgment-based) which are incomparable to local memoryless class. Recently, Garncarek et al. [14] investigated adversarial stability and other properties of deterministic local packet schedulers. They considered two classes of local schedulers: adaptive and non adaptive. The former allows stations to monitor and store some digest of the local queue history (especially its size), which is much more powerful than memoryless schedulers considered in this work, while the latter allows the policy only to check whether the current local queue is empty or not, which in turn is a type of memoryless policy. They showed that there is a local adaptive scheduling policy with relatively small memory, which is universally stable on a shared channel, that is, it has bounded queues for any $\rho < 1$ and $b \geq 0$. On the other hand, they proved that memoryless policies with only information about non-emptiness of their queues could reach the maximal stable injection rate of $O(1/\log n)$. They also showed a local non-adaptive policy, which is stable for slightly smaller injection rate $c/\log^2 n$, for some constant $c > 0$. In this context, general memoryless local policies considered in this work could be seen as in-between of adaptive and no-adaptive local schedulers considered in [14].

A simplified version of broadcasting on a shared channel with static input, in which some k stations hold packets at the beginning of an execution, was also widely investigated. The goal is to transmit at least one of them (*selection problem*) or all of them (*k-broadcast problem*), in both cases minimizing time complexity. The selection problem was studied in particular by Kushilevitz and Mansour [23] and Willard [26]. The k-broadcast problem was studied by Greenberg and Winograd [17], Komlós and Greenberg [21], and Kowalski [22]. A related leader election problem was studied by Jurdziński et al. [19] for channels without collision detection.

Deterministic solutions for the mutual exclusion and consensus problems on multiple-access channels when the adversary wakes up stations in arbitrary rounds were studied by Czyżowicz et al. [11]. A randomized counterpart of their research was delivered by Bieńkowski et al. [6].

Table 1 Comparison of our algorithms (WBA and QSA) with the previously known results under adversarial packet injections. Row with ρ describes the highest injection rate for which an algorithm is stable. Control messages can be separate packets (that need an additional successful transmission), piggybacked on packets (sent in the same round as successful transmission of any packet) or neither. Local memory is the number of bits to store information about history of events that each node can remember. Queue access denotes whether a node has access to the size of its queue or only knows if its queue is empty.

algorithm	oblivious [14]	WBA	QSA	adaptive [14]	MBTF [9]
ρ	$O(1/\log^2 n)$	$O(1/\log n)$	$O(1)$	< 1	≤ 1
b	any	any	bounded	any	any
control messages	no	no	no	separate	piggybacked
local memory	0	$O(1)$	0	large	$O(n \log n)$
queue access	emptiness	size	size	size	size

2 Model

We follow the description of local scheduling model under contention from [14], enhanced by additional aspects of memoryless policies. There are n stations attached to a shared channel. The stations have distinct names in $[n] = \{0, 1, \ldots, n\}$. Each station v knows n and its name $v \in [n]$.

2.1 Shared channel

A shared channel, also called a multiple access channel, models environments in which distributed nodes compete for access to the shared communication and distribution channel, and in case of contention, no contender wins the access. We assume that the channel operates synchronously. Every station (also called a node or queue) connected to it has its clock and the clock cycles are all of exactly the same length and synchronized. An execution of a protocol is partitioned into rounds – it takes precisely one round to transmit a message. We assume that stations have access to a global clock, meaning that all the local clocks at stations read the same round numbers.

Every station occasionally receives packets to broadcast. Packets are stored in a local queue. Stations use local scheduling algorithms to decide whether to schedule a packet from the queue for transmission in the current round or not. When exactly one station schedules a packet for transmission in a round, then the message containing this packet is successfully delivered and the packet disappears from the queue. We assume that local schedulers do not receive any feedback from the channel – they could only make their decisions based on examining local queue. When at least two stations transmit simultaneously in a round then *conflict for access* or *collision* occurs in the round and none of the transmitted packet is successful (i.e., all remain in their local queues). We consider syntactically weaker, but as we will show still powerful, *memoryless* local policies, in which stations may only access their current local queues and have no other feedback from the underlying distributed system. Moreover, the size of their internal writeable memory is bounded by some basic parameters and $O(\log n)$ control bits per each packet stored, storing e.g., packet destination or estimate of arrival time – see Table 1 for details. W.l.o.g. we assume that the queue at a station operates in the first-in-first-out (FIFO) fashion, as we are only interested in stability (i.e., bounded queue sizes).

2.2 Packet injections

Packets are injected into stations in a dynamic fashion in the course of an execution of a broadcasting protocol. We consider two packet injections – adversarial, corresponding to worst case executions, and stochastic, corresponding to average case executions.

Consider packet injection modeled by an adversary. Adversaries are specified by constraints on the maximum rate of injection ρ and by the burstiness of traffic b. An adversary generates a number of packets in each round and for each packet assigns a station to inject the packet in this round. The number of packets an adversary can inject into stations in one round is called the *burstiness* of the adversary. The adversary of type (ρ, b) can inject at most $\rho t + b$ packets to stations, in total, during any time interval of t rounds. This type of adversary is typically called *leaky bucket*.

We consider stochastic packet injections that are i.i.d. across rounds and independent across nodes. The probability that a (exactly one) packet is injected in a round t into a node v is ρ_v. Thus, each node receives packets in each round according to Bernoulli process. The value $\rho = \sum_v \rho_v$ is called the injection rate. We will only consider injections such that $\rho \leq 1$, since otherwise the system will obviously accumulate infinitely many packets over time.

2.3 Quality of service

We say that a local scheduler is *stable* for injection rate ρ if in any execution of the scheduler against a (ρ, b)-adversary, for any b, queues are bounded at all times. *Stochastic stability* means that the Markov Chain associated with the execution (random, due to stochastic injections) has a positive recurrent set of states with low queues, i.e., starting from any queue size, the expected time until the system contains few packets is bounded.

3 Generic queuing

In this section we present a queuing algorithm in the window-based model, which works for arbitrary burstiness. Thanks to that, our algorithm is stable for both adversarial (worst-case) and stochastic (average) packet injections. It is based on ultra-selectors – a combinatorial tool introduced in [8].

▶ **Definition 1** ([8])**.** *For a given $1 \leq a \leq n$ and $0 < \varepsilon \leq 1$ a (n, a, ε)-US (ultra-selector) of length m is a family of sets $S_1, \ldots, S_m$ such that for any set $A \subseteq [n]$ of size at most a and more than $a/2$, at least a ε fraction of the sets in the family intersect A on a single element: the size of $\{i \in [m] \mid |S_i \cap A| = 1\}$ is at least εm.*

▶ **Lemma 2** ([8])**.** *For each $\varepsilon < 1/32$ and sufficiently large n, there is an integer $c > 0$ such that for each $1 \leq a \leq n$, there exists an (n, a, ϵ)-US of length at most $m = c \cdot a \log(2n/a)$.*

We improve the existential result from [8] by extending the range $\varepsilon \in (0, 1/32)$ to $\varepsilon < (0, 1/(2e))$.

▶ **Lemma 3.** *For any $\epsilon < 1/(2e)$ there is an integer $c > 0$ such that for $1 \leq a \leq n$, there exists an (n, a, ϵ)-US of length at most $m = c \cdot a \log(2n/a)$.*

Proof. Observe that for $n/2 < a \leq n$ it is enough to take the family of all singletons of set $[n]$, as for any admissible set the number of its singleton intersections with the sets in the family is bigger than $a/2 > n/4$, which gives $\epsilon \geq 1/4 > 1/(2e)$ even better than the one claimed in the lemma.

It remains to consider $1 \le a \le n/2$. We show the existence by using the probabilistic method. Let each set of the (n, a, ϵ)-US be selected independently at random as follows, and the number of sets be $c \cdot a \log(2n/a)$, for some constant $c > 0$, depended on ϵ, to be specified later. For each set and integer $v \in [n]$, element v belongs to the set with probability $1/a$, independently on other elements and sets.

Consider an arbitrary set $A \subseteq [n]$ such that $a/2 < |A| \le a$. For any set T of the randomly created (n, a, ϵ)-US, the probability that $|T \cap A| = 1$ is

$$|A| \cdot \frac{1}{a} \cdot \left(1 - \frac{1}{a}\right)^{|A|-1} \ge \frac{1}{2} \cdot \left(1 - \frac{1}{a}\right)^{a-1} \ge \frac{1}{2e} \ .$$

Thus the expected number of sets T in the (n, a, ϵ)-US such that $|T \cap A| = 1$ is at least $1/(2e) \cdot c \cdot a \log(2k/a)$. By Chernoff bounds, the probability that the number of such sets is smaller than $\epsilon \cdot c \cdot a \log(2n/a)$ is less than

$$\exp\left(-1/(2e) \cdot c \cdot a \log(2n/a) \cdot (1/(2e) - \epsilon)^2 \cdot 1/2\right) = \exp\left(-c \cdot a \log(2n/a) \cdot \frac{1 - 2e\epsilon}{8e^2}\right) \ . \quad (1)$$

The number of all possible sets $A \subseteq [n]$ satisfying $a/2 < |A| \le a$ is

$$\sum_{i=a/2+1}^{a} \binom{n}{i} \le \frac{a}{2} \cdot \binom{n}{a} \le \frac{a}{2} \cdot \left(\frac{ne}{a}\right)^a \ , \quad (2)$$

as $\binom{n}{x}$ are monotonically increasing with growing $x \le a \le n/2$. Therefore, the probability that there is an admissible set A having less than $\epsilon \cdot c \cdot a \log(2n/a)$ singleton intersections with the random (n, a, ϵ)-US is at most

$$\frac{a}{2} \cdot \left(\frac{ne}{a}\right)^a \cdot \exp\left(-c \cdot a \log(2n/a) \cdot \frac{1 - 2e\epsilon}{8e^2}\right) \le$$

$$\le \exp\left(\ln(a/2) + a \ln(ne/a) - c \cdot a \log(2n/a) \cdot \frac{1 - 2e\epsilon}{8e^2}\right) \ .$$

This in turn is smaller than 1 for sufficiently large constant c, depending on ϵ. By the probabilistic method, there is a (deterministic) (n, a, ϵ)-US of length $c \cdot a \log(2n/a)$. ◀

In Sections 3.1–3.3, we devise a scheduling algorithm in window-based model which uses ultra-selectors as a building block and show its stability against adversarial as well as stochastic injections.

3.1 Algorithm

Our algorithm WBA (Window-Based Algorithm) makes use of ultra-selectors $(n, 2^i, \varepsilon)$-US, with $i = 0, 1, \ldots, \log n - 1$ such that the length of $(n, 2^i, \varepsilon)$-US is at most $c \cdot 2^i \log(2n)$ for the constant c from Lemma 2. Let L_i denote the length of the used $(n, 2^i, \varepsilon)$-US. Let j be the smallest number such that $2^j \ge L_i$ for all i. The algorithm works in windows which consist of $L = 2^j \log n$ rounds (see Algorithm 1 on page 7 for a pseudocode for a window). A node is *active* in a window if the number of packets in its queue at the beginning of the considered window is larger or equal to $\varepsilon 2^j$. Windows are split into $\log n$ phases of length 2^j each. During the ith phase of a window $(n, 2^{\log n - i}, \varepsilon)$-US is "repeated" $r_i = \lfloor 2^j / L_{\log n - i} \rfloor$ times and the remainder of the phase is wasted, i.e., no nodes try to transmit. (Note that the wasted period of a phase is no longer than half of the phase.) The ith phase of a window is determined by a chosen $(n, 2^{\log n - i}, \varepsilon)$-US of length $m \le 2^j$ in the following way. Let $S_1, \ldots, S_m$ be the chosen $(n, 2^{\log n - i}, \varepsilon)$-US. Then, S_k for each $k \in [m]$ is the set of *potential transmitters* for all rounds $k + lm$, such that $l \in [0, r_i]$. A node v transmits in the round t of the considered window if it is active in the current window, v belongs to the set of potential transmitters for round t and its queue is not empty.

Algorithm 1 Window(v). ▷ A window of Algorithm WBA

if $|Q_v| \ge \varepsilon 2^j$ **then** ▷ Q_v: the queue of packets of v
  $s_v \leftarrow$ active
else $s_v \leftarrow$ non-active
for $i = 1, \ldots, \log n$ **do** ▷ Phase i
  $(S_1, \ldots, S_m) \leftarrow$ the chosen $(n, 2^{\log n - i}, \varepsilon)$-US ▷ $m \le 2^j$
  for $t = 1, \ldots, 2^j$ **do** ▷ Round t of Phase i
    $P_t \leftarrow S_{1+(t-1) \bmod m}$
    if s_v=active and $Q_v \neq \emptyset$ and $v \in P_t$ **then**
      v transmits a message in round t of phase i

In the next two sections we will prove the following result.

▶ **Theorem 4.** *Algorithm WBA achieves stability against adaptive adversaries with injection rate $\rho \le \varepsilon/(2 \log n)$ and any burstiness b, where ε is a fraction of successes of $(n, 2^i, \varepsilon)$-US used. Moreover, WBA achieves stochastic stability against stochastic arrivals with injection rate $\rho' < \varepsilon/(2 \log n)$.*

3.2 Proof of part 1 of Theorem 4 – Adversarial analysis

Consider a window. We say that the ith phase of the window is *efficient* if the number of active nodes x in the current window satisfies the inequalities $2^{\log n - i - 1} \le x \le 2^{\log n - i}$, for $i = 0, 1, \ldots, \log n - 1$. We split further analysis into two scenarios.

Scenario 1. There is an efficient phase in the window.

If the ith phase is efficient, then the active nodes during this phase transmit without collisions in ε fraction of non-wasted rounds, i.e., they successfully transmit $\varepsilon 2^j/2$ times during the phase, unless some (at least one) active node u has not enough packets to do so. In the latter case, the node u transmitted all packets present in Q_u at the beginning of the current window. As u is active, the number of packets in Q_u at the beginning of the current window is at least $\varepsilon 2^j$. In either case there are at least $S = \varepsilon 2^j/2 \ge \rho L$ successful transmission since the start of the window by the active nodes.

Scenario 2. There is no efficient phase in the window.

In this case, there are no active nodes in the window and thus each node has less than $\varepsilon 2^j$ packets in its queue at the beginning of the window. As the adversary can inject at most $\rho 2^j \log n + b = \varepsilon 2^j/2 + b = S + b$ packets during the window, there are at most $\rho 2^j(n+1) + b$ packets in the system in each round of the window (in particular, at the end of the window).

Thus, one of the following two conditions is satisfied for each window W:

- Case 1: Window W satisfies Scenario 1. Consider a sequence of windows $W = W_1, W_2, \ldots, W_p$ such that W_{i+1} directly precedes W_i, Scenario 1 holds for the windows $W_2, \ldots, W_p$ and either W_p is the first window during an execution of the algorithm or the window W_{p+1} preceding W_p satisfies Scenario 2. Then, there are at most $(n+1)\rho L + b$

packets at the beginning of W_p (see Scenario 2). Moreover, at most $\rho pL + b$ packets are injected in the time period corresponding to the windows $W_p, W_{p-1}, \ldots, W_1$. On the other hand, at least ρL packets are successfully transmitted in each of the windows $W_1, \ldots, W_p$. Therefore, the number of packets in all queues at the end of W_1 is at most $(n+1)\rho L + 2b$.

- Case 2: Window W satisfies Scenario 2. As argued above in Scenario 2, there are at most $\rho 2^j(n+1) + b$ packets at the end of the window.

So the number of packets in the system is at most $P = (n+1)\rho L + 2b + \rho 2^j(n+1) + b$ at the end of each window. Therefore, there are at most $P + \rho L + b$ packets at each round of an execution of the algorithm, which proves its stability.

3.3 Proof of part 2 of Theorem 4 – Stochastic analysis

We will prove that our algorithm is stochastically stable. That is to say, we will show that the Markov Chain described by the queue states has a positive recurrent set of states with low queues, i.e., starting from any queue size, the expected time until the system contains few packets is finite.

Note that the expected number of packets injected into the system during a window of length $L = 2^j \log n$ equals $E(\sum_{i=1}^{L} X_i) = L \cdot E[X] = \rho 2^j \log n$.

In a window, we consider two scenarios: Scenario (1) – there is an efficient phase; Scenario (2) – there is no efficient phase.

In Scenario (2) there are no active nodes in the window and thus each node has less than $\varepsilon 2^j$ packets in its queue at the beginning of the window. As packet injections into each node are described by binomial distribution, each node may receive at most 1 packet per round. Therefore, during a window, each node can receive at most $L = 2^j \log n$ packets. So, at the end of the window, there are at most $n \cdot (\varepsilon 2^j + 2^j \log n)$ packets in the system.

In Scenario (1), as shown earlier (see Scenario (1) of the adversarial analysis), at least $s \geq \varepsilon 2^j/2$ packets are successfuly transmitted during the window. This means that the expected change in the total number of packets in the system is $\Delta \leq E(\sum_{i=1}^{L} X_i) - s \leq \rho \cdot 2^j \log n - \varepsilon 2^j/2$. For $\rho \leq (\varepsilon - \delta)/(2\log n)$ with some constant $\delta > 0$, we get $\Delta \leq (\varepsilon - \delta)/(2\log n) \cdot 2^j \log n - \varepsilon 2^j/2 = -\delta 2^j/2 < 0$.

We will use Foster-Lyapunov Theorem to prove that there will only be (in expectation) a finite number of windows of the type (2), before a window of the type (1) is reached.

Let Q_i denote a vector of queue sizes of all nodes at round i of the algorithm execution. Note that Q_{i+1} depends on the value of Q_i, the stochastic injections (which are i.i.d. across rounds) and algorithm's transmission (which are a function of Q_i). Therefore, (Q_i) is a time-homogeneous Markov chain.

▶ **Theorem 5** (Foster-Lyapunov Theorem (see Theorem 1 in [12])). *Consider a time-homogeneous Markov chain (X_i). Suppose that the drift $E[V(X_1) - V(X_0)|X_0 = x]$ of some function V in one step satisfies the following conditions, for some positive N_0, c, and H:*

$$E[V(X_1) - V(X_0)|X_0 = x] \leq -c \quad \textit{if} \quad V(x) > N_0,$$

$$E[V(X_1) - V(X_0)|X_0 = x] \leq H < \infty \quad \textit{if} \quad V(x) \leq N_0$$

Then the set $B = \{x : V(x) \leq N_0\}$ is positive recurrent.

Consider a function $V(Q)$ that assigns to a queue state Q the number of packets in the system. We will show that set $B = \{q : V(q) \leq n\varepsilon 2^j + nL\}$ is positive recurrent.

At the end of a window of type (1), the queue sizes are bounded by $n\varepsilon 2^j + nL$. This means that, starting from any queue sizes, the system will return in expected finite time to bounded queues, i.e., the algorithm is stochastically stable.

Now we will estimate the drift of function L at the ends of consecutive windows.

$$E[V(Q_1) - V(Q_2)|Q_1 = q \text{ yields a window of type (2) }]$$
$$= V(Q_1) - E[V(Q_1) + \Delta|Q_1 = q \text{ yields a window of type (2) }] \leq -\delta/(2\log n) < 0$$

Therefore, according to the Foster-Lyapunov Theorem, after a finite (in expectation) number of windows, the system returns to set B of states, i.e., to a state that yields a window of type (1), and after that window, the queues are bounded by $n\varepsilon \cdot 2^j + nL$. Therefore, our algorithm is stochastically stable.

4 Queueing in the model without memory

In this section we consider the qsa model in which nodes do not have memory to write any information about history of an execution of an algorithm. That is, at each round t, a node i has only access to: the size of the network n, the value t of the global clock, its own ID i, the size $q_{i,t}$ of its queue of packets, and the upper bound on burstiness b. Thus importantly, the nodes can not store any information about history of computation and communication. In the remaining part of this section, we prove the following theorem.

▶ **Theorem 6.** *There exists a constant $\rho > 0$ such that for each $b, n > 0$, there exists a scheduling algorithm in qsa model which is stable against each adversary with injection rate ρ and burstiness b.*

4.1 Overview of the proof of Theorem 6

As before, our algorithm relies on an appropriate combinatorial structure determining behaviour of nodes in particular rounds of a window, where the number of rounds of the window corresponds to the size of the underlying combinatorial structure. Algorithm QSA uses $\log n$ different ultra-selectors in order to adjust to the unknown number of active nodes (i.e., the nodes with sufficiently many packets in their queues) in the current window . The fact that we need $\log n$ different selectors limits acceptable injection rate to $O(1/\log n)$. In order to achieve stability for injection rates $O(1)$, we build a new combinatorial structure which adjusts to the actual number of active nodes. However, we face here additional difficulty: the nodes do not have opportunity to store information about history, therefore "activity" can be determined merely on the current size of the queue of the node (not fixed for the whole window, as in Algorithm QSA). In order to overcome this additional difficulty and simultaneously improve acceptable injection rate to $O(1)$, the new combinatorial structure is not just a sequence of sets determining potential transmitters. Instead, each element (i.e., each round of the algorithm) of the structure is a vector of n thresholds $[M_1, \ldots, M_n]$. In our algorithm, the node $i \in \{1, \ldots, n\}$ is a potential transmitter in a round corresponding to $[M_1, \ldots, M_n]$ iff the number $|Q_i|$ of packets in the queue of i is at least M_i. Moreover, in order to prevent the adversary from malicious adjustment of sizes of queues (by injections) to the thresholds, we use thresholds which are multiplicities of some parameter depending on the burstiness b.

The general idea of the construction of the new combinatorial structure, called a *capacitated selector* is as follows. Assume that the sum of the sizes of queues $\sum_{i=1}^{n} |Q_i|$ is at most $S \gg b$. Then, there is at most 1 node with at least S packets, at most 2 nodes with at least $S/2$ packets each and in general there are at most 2^i nodes with at least $S/2^i$ packets each. On the other hand, if the sum of the sizes of queues is at least $S/2$ and at most S, then there is at least one $i \in [\log n]$ such that the number of nodes with queue sizes $\geq S/2^i$ is in the range $[2^{i-2}, 2^i]$. (If the number of nodes with queue sizes $\geq S/2^i$ were greater than 2^i for any $i \in [\log n]$, then the sum of the queue sizes would be greater than S. If the number of nodes with queue sizes $\geq S/2^i$ were smaller than 2^{i-2} for all $i \in [\log n]$, then the sum of the queue sizes would be smaller than $S/2$.) Therefore, if only nodes with queue sizes $\geq S/2^i$ are potential transmitters, we can use $(n, 2^{i-2}, \varepsilon)$-US or $(n, 2^{i-1}, \varepsilon)$-US. As we do not know the particular i satisfying these conditions, a direct application of ultra-selectors would require to check all values of $i \in [\log n]$ which requires $\log n$ rounds and results in the injection rate $O(1/\log n)$, as in Theorem 4. Instead, we "compress" all these $\log n$ selectors using thresholds in the following way. We choose the threshold $[M_1, \ldots, M_n]$ determining possible transmitters such that $M_j = 2^i$ with probability $p_i = \frac{2^i}{cS}$ for some (fixed) constant c. This assignment assures that the expected number of nodes exceeding their thresholds is $\Theta(1)$. Using Probabilistic Method, we show that such adjustment of thresholds gives constant fraction of successful rounds with non-zero probability, for large enough sequence of rounds. This in turn guarantees that the number of all packets in the system decreases in a window, provided the number of packets was in the range $[S/2, S]$ at the beginning of that window.

The above ideas need further modifications to obtain an actual scheduling algorithm guaranteeing stability for constant injection rates. In particular, the fact that the sizes of queues can change both by successful transmissions and packet injections has to be taken into account in the the estimation of probability that a chosen set of thresholds guarantees sufficiently many successful rounds for **each** possible initial queue sizes and adversarial injections.

4.2 Proof of Theorem 6

Our algorithm QBA (Query-size Based Algorithm) divides time into windows of length L (to be fixed later). Behavior of each node in each window is determined by a fixed matrix M over natural numbers with n rows and L columns. The node i transmits a packet in round t of a window iff the number $q_{i,t}$ of packets in its queue at the beginning of that round is larger than or equal to $M_{i,j}$. For a fixed matrix M, QBA(M) denotes the instance of QBA algorithm which uses the matrix M in the above described way.

In order to adjust requirements sufficient for stability of an algorithm determined by such a matrix M, we characterize states of the network (i.e., sizes of queues) and adversarial injections by appropriate matrices. To this aim, we introduce the following combinatorial structure.

▶ **Definition 7.** *Given natural numbers n, b and $0 < \rho < 1$, a matrix M over positive natural numbers with n rows and L columns is a (n, ρ, b)* capacitated adversarial selector *((n, ρ, b)-cas) of size L iff there exists a natural number m, called the* load *of M, such that*

$$\sum_{i=1}^{n} q_{i,L} \leq m \tag{3}$$

for each sequence $q_{1,0}, \ldots, q_{n,0} \in \mathbb{N}$ such that $\sum_{i=1}^{n} q_{i,0} \leq m$ and each matrix $A \in \mathbb{N}^{n \times L}$ such that $\sum_{i=1}^{n} \sum_{t=1}^{L} A_{i,t} \leq \rho L + b$, where

$$q_{i,t} = \begin{cases} \max(0, q_{i,t-1} + A_{i,t} - 1) & \text{if } q_{i,t-1} + A_{i,t} \geq M_{i,t} \text{ and } q_{j,t-1} + A_{j,t} < M_{j,t} \\ & \text{for each } j \neq i \\ q_{i,t-1} + A_{i,t} & \text{otherwise} \end{cases}$$

for each $i \in [n]$ and $t \in [L]$.

Intuitively, the vector $q_{1,0}, \ldots, q_{n,0}$ in the above definition corresponds to the sizes of queues at the beginning of a window of the algorithm QBA executed according to the matrix M. Then, $A_{i,t}$ is equal to the number of packets injected in the queue of the station i at round t of the window. Moreover, $q_{i,t}$ describes the number of packets in the node i after round t of the window. The assumption $\sum_{i=1}^{n} \sum_{t=1}^{L} A_{i,t} \leq \rho L + b$ corresponds to the restriction on a leaky bucket (ρ, b) adversary. The condition $q_{i,t-1} + A_{i,t} \geq M_{i,t}$ corresponds to node i transmitting at round t, while the condition $q_{j,t-1} + A_{j,t} < M_{j,t}$ for each $j \neq i$ corresponds to the situation that all nodes other than i are not transmitting in round t. Finally, the inequality $\sum_{i=1}^{n} q_{i,L} \leq m$ implies that the number of packets in all queues does not exceed m at the end of the window, provided the number of packets in queues at the beginning of the window, $\sum_{i=1}^{n} q_{i,0}$, is at most m as well. Below, we formalize this intuition.

▶ **Lemma 8.** *Assume that there exists $0 < \rho < 1$ such that, for each sufficiently large $n \in \mathbb{N}$ and each natural b, there exists a (n, ρ, b)-cas $M \in \mathbb{N}^{n,L}$ for some $L \in \mathbb{N}$, with load $m > \rho L + b$. Then, the algorithm QBA(M) is stable against a (ρ, b) leaky bucket adversary.*

Proof. Let $M \in \mathbb{N}^{n,L}$ be a (n, ρ, b)-cas with load $m > \rho L + b$. We prove that the overall number of packets in all queues is at most m over an execution of QBA(M) against a (ρ, b) leaky bucket adversary. The proof goes by induction with respect to the number of windows.

As discussed above, if the sizes of queues at the beginning of the window are equal $q_{1,0}, \ldots, q_{n,0}$ and $A_{i,t}$ denotes the number of packets injected by the adversary at round t of the window in the queue of the node i, then $q_{i,t}$ (defined as in Def. 7) for $i \in [n]$ and $t \in [L]$ denotes the number of packets in the queue of i after round t, for each $i \in [n]$ and $t \in [L]$. There are no packets at the beginning of the first window of an execution of the algorithm. Thus the number of packets at the end of that window is at most $\rho L + b \leq m$, since the adversary can inject at most $\rho L + b$ packets in L rounds. For the inductive step assume that the overall number of packets at the beginning of the jth window of the execution is $\sum_{i=1}^{n} q_{i,0} \leq m$. Then, the overall number of packets at the end of the window is at most $\sum_{i=1}^{n} q_{i,L}$, where $q_{i,t}$ are determined as in Definition 7. As M is (n, ρ, b)-cas with load m, the final number of packets at the end of the considered window is at most $\sum_{i=1}^{n} q_{i,L} \leq m$. ◀

Given the above connection between capacitated adversarial selectors and QBA algorithm, it is sufficient to show that (n, ρ, b)-cas exists for a constant $\rho > 0$.

▶ **Lemma 9.** *There exists a constant $\rho > 0$ such that for each large enough $n \in \mathbb{N}$ and each $b \geq 0$, there exists a (n, ρ, b)-cas M of length $L = O(n \log n + b)$ with load $m = 8nL$.*

Observe that Th. 6 follows directly from Lemmas 8 and 9. Thus, it remains to prove Lem. 9.

4.3 Proof of Lemma 9

In order to emphasize connections with the algorithm QSA, the indices $i \in [n]$ are called nodes, and $t \in [L]$ are called rounds.

Using Probabilistic Method, we will show that for each n and b, there exist L, m and a matrix M which guarantees the properties stated in Definition 7. More specifically, we prove the lemma for each $\rho < 2^{-\frac{33}{8}}$, some $L = O(n \log n)$ such that $L \geq \rho L + b$ and load $m = \frac{1}{2} S_{\max} L$, where $S_{\max} = 16n$.

Consider any $q_{1,0}, \ldots, q_{n,0}$ such that $\sum_{i=1}^{n} q_{i,0} \leq m$ and $A_{i,t}$ such that $\sum_{i=1}^{n} \sum_{t=1}^{L} A_{i,t} \leq \rho L + b \leq L$. Let us consider two cases:

Case 1: $\boldsymbol{\sum_{i=1}^{n} q_{i,0} \leq m/2}$. Then, $\sum_{i=1}^{n} q_{i,L} \leq \sum_{i=1}^{n} q_{i,0} + \sum_{i=1}^{n} \sum_{t=1}^{L} A_{i,t} \leq \frac{1}{2} m + \rho L + b \leq \frac{1}{2} m + L < m$, and therefore the statement of the lemma holds.

Case 2: $m/2 \le \sum_{i=1}^n q_{i,0} \le m$. We consider Case 2 in the remaining part of the proof. Let (round) $t \in [L]$ be *successful* if $q_{i,t-1} + A_{i,t} \ge M_{i,t}$ and $q_{i,t-1} + A_{i,t} > 0$ for exactly one value of $i \in [n]$. Then

$$\sum_{i=1}^n q_{i,L} \le \sum_{i=1}^n q_{i,0} + \sum_{i=1}^n \sum_{i=1}^L A_{i,t} - |\{t \in [L] \,|\, t \text{ is successful}\}$$

Now, our goal is to show that there exists matrix M which guarantees that $\sum_{i=1}^n \sum_{i=1}^L A_{i,t} - |\{t \in [L] \,|\, t \text{ is successful}\}| \le 0$, i.e., $|\{t \in [L] \,|\, t \text{ is successful}\}| \ge \rho L + b$.

Consider a random matrix M with n rows and L columns such that

$$M_{i,t} = \begin{cases} L \cdot 2^k - t & \text{with probability } \dfrac{2^k}{c \cdot S_{max}} \text{ for } k \in [\log S_{\max}] \\ \infty & \text{with probability } 1 - \sum_{j=1}^{\log S_{\max}} \frac{2^j}{c \cdot S_{\max}}, \end{cases} \tag{4}$$

where $c = 4$. Our final goal is to show that such a matrix is (n, ρ, b)-cas of size L with load m with non-zero probability which implies that such a matrix exists.

Given the above description of the probabilistic choice of M, we introduce some auxiliary terminology and examine its properties. We say that (the node) i has $s_{i,t} = \lceil (q_{i,t} + t)/L \rceil$ *blocks* at round t. Below, we observe that the number of blocks in a node i can change (at most) twice in rounds $1, \ldots, L$ and this prospective changes are just increments by one.

▶ **Proposition 10.** *Let $s_{i,0}, \ldots, s_{i,L}$ be the number of blocks of the node i in rounds $1, \ldots, L$. Then, $s_{i,0} \le s_{i,1} \le \cdots \le s_{i,L} \le s_{i,0} + 2$, provided that $\rho L + b \le L$.*

Proof. Note that $q_{i,t+1} \ge q_{i,t} - 1$ and therefore $s_{i,t+1} = \lceil (q_{i,t+1} + (t+1))/L \rceil \ge \lceil (q_{i,t} - 1 + (t+1))/L \rceil = s_{i,t}$. On the other hand, $q_{i,t} \le q_{i,0} + \sum_{t'=1}^t A_{i,t'} \le q_{i,0} + \rho L + b \le q_{i,0} + L$ and therefore
$s_{i,t} \le \lceil (q_{i,0} + L + t)/L \rceil \le \lceil q_{i,0}/L \rceil + 2 = s_{i,0} + 2$. ◀

The following proposition shows that the number of all blocks is limited for all $t \in [L]$. Intuitively, it follows from the facts that the initial number of packets is at most $m = O(nL)$, the sum of "injections" is $\sum_{i=1}^n \sum_{t=1}^L A_{i,t} = O(L)$, while the size of a block is of the order of L.

▶ **Proposition 11.** *The overall number of blocks $S_t = \sum_{i=1}^n s_{i,t}$ in all nodes at round $t \in [L]$ is at least $S_{\max}/4$ and at most $S_{\max}$, provided that $\sum_{i=1}^n q_{i,0} \in [m/2, m]$, $m = \frac{1}{2} S_{\max} L$ where $S_{\max} = 16n$.*

Proof. At the beginning ($t = 0$), we have

$$\sum_{i=1}^n s_{i,0} = \sum_{i=1}^n \lceil q_{i,0}/L \rceil \le \sum_{i=1}^n (q_{i,0}/L + 1) \le \frac{1}{L} m + n = \frac{1}{2} S_{\max} + n \le \frac{9}{16} S_{\max}$$

where the last inequality follows from the assumption $S_{\max} = 16n$. By Proposition 10, $s_{i,t} \le s_{i,0} + 2$ for each $t \in [L]$. Therefore, for each $t \in [L]$,

$$\sum_{i=1}^n s_{i,t} \le \sum_{i=1}^n (s_{i,0} + 2) \le \frac{9}{16} S_{\max} + 2n = \frac{11}{16} S_{\max} \le S_{\max}.$$

For the lower bound on the number of blocks, we use the property from Proposition 10 that $s_{i,t} \ge s_{i,0}$ for each $t \in [L]$: $\sum_{i=1}^n s_{i,t} \ge \sum_{i=1}^n s_{i,0} \ge \sum_{i=1}^n (q_{i,0}/L) \ge \frac{1}{L} \cdot \frac{m}{2} = \frac{1}{4} S_{\max}$. ◀

Given the notion of blocks and its properties, we split the set (of nodes) $[n]$ at round t into *groups* $B_{0,t}, B_{1,t}, \ldots, B_{\log S_{\max},t}$ according to the numbers of blocks such that

$$B_{l,t} = \{i \in [1,n] \mid 2^{l-1} \le \lceil (q_{i,t}+t)/L \rceil < 2^l\} \tag{5}$$

for $l > 0$ and $B_{0,t}$ is the set of nodes $i \in [n]$ such that $q_{i,t} + t < L$, i.e., the nodes with just one block. Thus, $i \in B_{l,t}$ iff the number of blocks of node i in round t satisfies the inequality $2^{l-1} \le s_{i,t} < 2^l$. For the sake of brevity, we say that a node i belongs to the group B_l in rounds t iff i is in the group $B_{l,t}$. Proposition 10 implies that each node $i \in B_{j,\_}$ can only move to $B_{j+1,\_}$ and then to $B_{j+2,\_}$. Using (5) and Proposition 11, we can bound the number S_t of blocks at round t from above:

$$S_{\max}/4 \le S_t = \sum_{i=1}^{n} s_{i,t} \le \sum_{j=0}^{\log S_{\max}} |B_{j,t}| 2^j \tag{6}$$

and below

$$S_{\max} \ge S_t = \sum_{i=1}^{n} s_{i,t} \ge \sum_{j=0}^{\log S_{\max}} |B_{j,t}| 2^{j-1} \tag{7}$$

Now, given the column/round t, we would like to estimate the probability that the inequality $q_{i,t} \ge M_{i,t}$ holds for exactly one $i \in [n]$ (i.e., t is successful). For a node $i \in B_{l,t}$, we have $2^{l-1} \le \lfloor (q_{i,t}+t)/L \rfloor < 2^l$. Given the possible values of $M_{i,t}$ (see equality (4)), the highest value of $M_{i,t}$ that is no larger than $q_{i,t}$ is $M_{i,t} = L \cdot 2^l - t$. Therefore, we have

$$\text{Prob}[q_{i,t} \ge M_{i,t}] = \text{Prob}[M_{i,t} \le L \cdot 2^l - t] = \sum_{k=1}^{l-1} \frac{2^k}{cS_{\max}} \in \left[\frac{2^{l-1}}{cS_{\max}}, \frac{2^l}{cS_{\max}}\right]. \tag{8}$$

Let $p_t = \sum_{i=1}^{n} \text{Prob}[q_{i,t} \ge M_{i,t}]$, i.e., p_t is the expected number of nodes i such that $q_{i,t} \ge M_{i,t}$. Using (8) and (6), we obtain the following bounds:

$$\begin{aligned} p_t &= \sum_{i=1}^{n} \text{Prob}[q_{i,t} \ge M_{i,t}] \ge \sum_{j=1}^{\log S_{\max}} |B_{j,t}| \frac{2^{j-1}}{c \cdot S_{\max}} \ge \frac{1}{2c \cdot S_{\max}} \sum_{j=0}^{\log S_{\max}} |B_{j,t}| 2^j - |B_{0,t}| \\ &\ge \frac{1}{2c \cdot S_{\max}} \cdot \left(\frac{S_{\max}}{4} - |B_{0,t}|\right) \ge \frac{1}{16c} \end{aligned}$$

where the last inequality follows from the fact that $S_{\max} = 16n$ and $|B_{0,t}| \le n$. On the other hand

$$\begin{aligned} p_t &= \sum_{i=1}^{n} \text{Prob}[q_{i,t} \ge M_{i,t}] \le \sum_{j=1}^{\log S_{\max}} |B_{j,t}| \frac{2^j}{c \cdot S_{\max}} \le \frac{2}{c \cdot S_{\max}} \cdot \sum_{j=1}^{\log S_{\max}} |B_{j,t}| 2^{j-1} \\ &\le \frac{2}{c \cdot S_{\max}} S_{\max} = \frac{2}{c}. \end{aligned}$$

Recall that $c = 4$. It is well known (see e.g. [20]) that given independent events $E_1, \ldots, E_n$ such that $x \le \sum_{i=1}^{n} \text{Prob}[E_i] \le \frac{1}{2}$, the probability that exactly one of the events $E_1, \ldots, E_n$ is satisfied is at least $x(1/4)^x$. Thus, in our case, the probability that exactly one of the events $q_{i,t} \ge M_{i,t}$ occurs is at least $\frac{1}{16c}(1/4^{1/(16c)}) \ge d$ for $d = \frac{1}{4^4}$. Let $X = |\{t \in [L] \mid t \text{ is successful}\}|$ denote the number of successful "rounds" in $[L]$ for fixed values of $q_{i,0}$ and $A_{i,t}$ for $i \in [n]$ and $t \in [L]$. The expected value of X is $EX = dL$. For the sake of derandomization, we would like to show that $X \ge \frac{1}{2}EX = \frac{1}{2}dL$ with probability $\ge 1 - 1/2^{-\Omega(L)}$.

Observe that $q_{i,t} \geq M_{i,t}$ for $M_{i,t} = L \cdot 2^k - t$ iff $i \in \bigcup_{j=k}^{\log S_{\max}} B_{j,t}$. Let us define a *scenario* as a fixed sequence of partitions of $[n]$ into groups $B_{0,t}, \ldots, B_{n,t}$ for all $t \in [L]$ which might appear in our setting. There are at most $(1 + \log S_{max})^n$ initial partitions into groups $B_{0,0}, \ldots, B_{n,0}$, since each $j \in [n]$ is in exactly one block from $B_{i,0}, \ldots, B_{i,\log S_{\max}}$. Moreover, each $i \in [n]$ can change its group at most twice (increase by one – see Prop. 10). One can encode such changes of (a node) $i \in [n]$ by (at most) two numbers in $[L]$ determining indices in $[L]$ of (possible) increases of the index of the block at some rounds $t_1, t_2 \in [L]$. Thus, there are at most $(1 + \log S_{max})^n \cdot L^{2n}$ possible scenarios. For a fixed scenarios, the success probabilities $\text{Prob}[|\{i \,|\, q_{i,t} \geq M_{i,t}\}| = 1]$ for various rounds $t \in [L]$ are independent. Therefore, we can use Chernoff Bound to estimate the probability that there are less than $dL/2$ successful rounds:

$$\text{Prob}[X \leq dL/2] \leq \text{Prob}[X \leq (1 - 1/2)EX] \leq e^{-dL/8}.$$

Now, we would like to estimate the proabability (for a matrix M chosen randomly as described above) that there are at least $dL/2$ successful rounds $t \in [L]$ for any scenario. By the union bound, the probability that there exists a scenario with less than $dL/2$ successful rounds is at most

$$e^{-dL/8} \cdot (1 + \log S_{max})^n \cdot L^{2n} < e^{-dL/8} \cdot (17n)^n \cdot L^{2n} = e^{-dL/8 + n\ln(17n) + 2n\ln L} < 1$$

for each $L \geq L_0$ such that $L_0 = O(n \log n)$. Thus, there exists a matrix M which guarantees $dL/2$ successful rounds, provided that $\sum_{i=1}^{n} q_{i,0} \in [m/2, m]$ as assumed in Case 2. If ρ and L are such that $\rho L + b \leq dL/2$, then $\sum_{i=1}^{n} \sum_{t=1}^{L} A_{i,t} \leq \rho L + b \leq dL/2$ is smaller than

$$|\{t \in [L] \,|\, q_{i,t} \geq M_{i,t} \text{ for exactlny one } i \in [n]\}| \geq dL/2.$$

Thus, given the constants $d = 1/4^4$, $\rho < d/2$, any $L \geq L_1$ for $L_1 = O(n \log n + \frac{b}{d/2-\rho}) = O(n \log n + b)$ guarantees $\sum_{i=1}^{n} q_{i,L} \leq \sum_{i=1}^{n} q_{i,0}$, which finishes the proof of Lemma 9.

5 Conclusions

We investigated what stability guarantees we could get in a system with contention if protocols have very limited (or no) space to store information inherited from history of computation and communication. A natural research direction would be to prove tight bound on injection rates and optimize other measures, such as packet latency. Schedulers could also be studied in the context of other related models, such as SINR or dependency-graph models. We introduced a novel class of selectors with thresholds, unlike the previously used binary selectors/codes – studying their constructiveness in polynomial time and further applicability is a prospective open direction.

References

1. Norman M. Abramson. Development of the ALOHANET. *IEEE Transactions on Information Theory*, 31(2):119–123, 1985.
2. L. Anantharamu, Bogdan S. Chlebus, and Mariusz A. Rokicki. Adversarial multiple access channel with individual injection rates. In *Proceedings of the 13th International Conference on Principles of Distributed Systems (OPODIS)*, LNCS 5923, pages 174–188. Springer-Verlag, 2009.
3. Lakshmi Anantharamu, Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Deterministic broadcast on multiple access channels. In *Proceedings of the 29th IEEE International Conference on Computer Communications (INFOCOM)*, pages 1–5, 2010.

4 Matthew Andrews, Baruch Awerbuch, Antonio Fernández, Frank Thomson Leighton, Zhiyong Liu, and Jon M. Kleinberg. Universal-stability results and performance bounds for greedy contention-resolution protocols. *Journal of the ACM*, 48(1):39–69, 2001.

5 Michael A. Bender, Martin Farach-Colton, Simai He, Bradley C. Kuszmaul, and Charles E. Leiserson. Adversarial contention resolution for simple channels. In *Proceedings of the 17th Annual ACM Symposium on Parallel Algorithms (SPAA)*, pages 325–332, 2005.

6 Marcin Bieńkowski, Marek Klonowski, Mirosław Korzeniowski, and Dariusz R. Kowalski. Dynamic Sharing of a Multiple Access Channel. In *Proceedings of the* 27*th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 83–94, 2010.

7 Allan Borodin, Jon M. Kleinberg, Prabhakar Raghavan, Madhu Sudan, and David P. Williamson. Adversarial queuing theory. *Journal of the ACM*, 48(1):13–38, 2001.

8 Bogdan S. Chlebus, Dariusz R. Kowalski, Andrzej Pelc, and Mariusz A. Rokicki. Efficient Distributed Communication in Ad-Hoc Radio Networks. In *Proceedings of the* 38*th International Colloquium on Automata, Languages and Programming (ICALP), Part II*, volume 6756 of *Lecture Notes in Computer Science*, pages 613–624. Springer, 2011.

9 Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Maximum throughput of multiple access channels in adversarial environments. *Distributed Computing*, 22(2):93–116, 2009.

10 Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Adversarial Queuing on the Multiple Access Channel. *ACM Transactions on Algorithms*, 8(1):5:1–5:31, 2012.

11 Jurek Czyżowicz, Leszek Gąsieniec, Dariusz R. Kowalski, and Andrzej Pelc. Consensus and Mutual Exclusion in a Multiple Access Channel. In *Proceedings of the 23rd International Symposium on Distributed Computing (DISC)*, LNCS 5805, pages 512–526. Springer-Verlag, 2009.

12 Sergey Foss. Lectures on Stochastic Stability. Abo Akademi University, June 2008. URL: `http://web.abo.fi/fak/mnf/mate/tammerfors08/Foss_Lecture2.pdf`.

13 Robert G. Gallager. A perspective on multiaccess channels. *IEEE Transactions on Information Theory*, 31(2):124–142, 1985.

14 Pawel Garncarek, Tomasz Jurdzinski, and Dariusz R. Kowalski. Local Queuing Under Contention. In *32nd International Symposium on Distributed Computing, DISC 2018, New Orleans, LA, USA, October 15-19, 2018*, pages 28:1–28:18, 2018. `doi:10.4230/LIPIcs.DISC.2018.28`.

15 Leslie Ann Goldberg, Mark Jerrum, Sampath Kannan, and Mike Paterson. A bound on the capacity of backoff and acknowledgment-based protocols. *SIAM Journal on Computing*, 33(2):313–331, 2004.

16 Leslie Ann Goldberg, Philip D. MacKenzie, Mike Paterson, and Aravind Srinivasan. Contention resolution with constant expected delay. *Journal of the ACM*, 47(6):1048–1096, 2000.

17 Albert G. Greenberg and Shmuel Winograd. A Lower Bound on the Time Needed in the Worst Case to Resolve Conflicts Deterministically in Multiple Access Channels. *Journal of the ACM*, 32(3):589–596, 1985.

18 Johan Håstad, Frank Thomson Leighton, and Brian Rogoff. Analysis of Backoff Protocols for Multiple Access Channels. *SIAM J. Comput.*, 25(4):740–774, 1996. `doi:10.1137/S0097539792233828`.

19 Tomasz Jurdziński, Miroslaw Kutyłowski, and Jan Zatopiański. Efficient algorithms for leader election in radio networks. In *Proceedings of the 21st ACM Symposium on Principles of Distributed Computing (PODC)*, pages 51–57, 2002.

20 Tomasz Jurdzinski and Grzegorz Stachowiak. Probabilistic Algorithms for the Wake-Up Problem in Single-Hop Radio Networks. *Theory Comput. Syst.*, 38(3):347–367, 2005. `doi:10.1007/s00224-005-1144-3`.

21 János Komlós and Albert G. Greenberg. An asymptotically fast nonadaptive algorithm for conflict resolution in multiple-access channels. *IEEE Transactions on Information Theory*, 31(2):302–306, 1985.

22 Dariusz R. Kowalski. On selection problem in radio networks. In *Proceedings of the 24th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 158–166, 2005.

23 Eyal Kushilevitz and Yishay Mansour. An $\Omega(D \log(N/D))$ Lower Bound for Broadcast in Radio Networks. *SIAM Journal on Computing*, 27(3):702–712, 1998.

24 Robert M. Metcalfe and David R. Boggs. Ethernet: Distributed Packet Switching for Local Computer Networks. *Communications of the ACM*, 19(7):395–404, 1976.

25 Prabhakar Raghavan and Eli Upfal. Stochastic Contention Resolution With Short Delays. *SIAM Journal on Computing*, 28(2):709–719, 1998.

26 Dan E. Willard. Log-Logarithmic Selection Resolution Protocols in a Multiple Access Channel. *SIAM Journal on Computing*, 15(2):468–477, 1986.