Adaptive Batch Resolution Algorithm with Deferred Feedback for practical CSMA Wireless Networks

Andrea Zanella

Dep. of Information Engineering

University of Padova (Italy)

email: zanella@dei.unipd.it [1]

November 26, 2010

[1] TEXT

Abstract

A batch is a group of nodes that have to transmit a single packet each to a common receiver in the shortest time. Most of existing batch resolution algorithms assume immediate feedback and generally neglect the feedback time, being considered much shorter than the packet transmission time. This conjecture, however, fails to apply in many practical high-rate wireless systems, with the consequence that the classical performance analysis of batch resolution algorithms may result overoptimistic. In this report we propose and analyze a batch resolution algorithm for CSMA wireless networks that waives the immediate feedback approach in favor of a deferred feedback method, which shall reduce the overhead costs. The scheme, named *Adaptive Batch Resolution Algorithm with Deferred Feedback* (ABRADE$^+$), is obtained by merging a batch size estimate module with a framed ALOHA access scheme, whose frame length is dynamically adapted to the residual batch size in order to minimize the overall batch resolution time. ABRADE's adaptation strategy is designed under the assumption of known batch size. To remove this constraint, we propose a batch size estimation module that, coupled with ABRADE, gives rise to ABRADE$^+$, a complete batch resolution algorithm that works even with no *a priori* information on the batch size. We then extend ABRADE with a batch size estimation module, thus obtaining a complete batch resolution algorithm, called ABRADE$^+$, that works even with no *a priori* information on the batch size. ABRADE$^+$ is compared against the batch resolution algorithms based on the immediate feedback paradigm. Results confirm that, in practical CSMA systems, ABRADE$^+$ outperforms the algorithms based on the immediate feedback paradigm, both in case of partial and no *a priori* knowledge of the batch multiplicity.

Chapter 1

Introduction

The *batch resolution problem*, also known in literature as collision or conflict resolution problem, consists in managing the channel access of a group of nodes, which form the *batch* or conflict set,[1] in such a way that each node successfully delivers *a single message* to a common recipient in the shortest time. This problem typically arises in wireless communication scenarios with densely populated clouds of terminals, such as Internet of things, opportunistic networks, dust networks, where it is common for a node to inquiry the surrounding nodes for different purposes, such as discovering neighbors, collecting data, updating connectivity information and so on. The probe message sent by the inquirer may, in fact, solicit reply from a potentially large set of nodes, the batch, that compete to transmit their message over the shared wireless channel. A *Batch Resolution Algorithm* (BRA) is a channel access scheme designed to *resolve* the batch, that is to say, let each node in the batch deliver its message to the inquirer.

The batch resolution problem resembles, in some aspects, the medium access control (MAC) problem in that both consist in managing the channel access of a group of nodes. However, MAC protocols generally look at the channel contention as a steady-state phenomenon, where the overall traffic offered to the medium can be considered a stationary process. Conversely, BRAs address scenarios where contention has a bursty nature [1]: the medium is typically idle, until a particular event solicits packet transmission from a bunch of nodes simultaneously. Furthermore, nodes that successfully deliver their message to the inquirer are *resolved* and leave the batch, so that the size of the problem, i.e., the number of nodes that still compete to access the channel, progressively decreases over time. Nonetheless, it is possible to apply the BRAs as MAC protocols by adopting a simple gated policy, according to which nodes with pending packets form a batch, which is resolved by letting each node transmit a *single* packet to its intended destination. New messages arriving during the resolution of a batch are held in queue and form the new batch

[1] The two terms will be used interchangeably throughout the report.

that will be resolved next. This access scheme is generally referred to as the *obvious* MAC scheme [8]. In this report, however, we are more concerned with the batch resolution problem as its own, rather than to its interpretation as obvious MAC.

BRAs can be divided in two broad categories, namely *partial*, which aim at resolving only a fraction of the nodes in the original batch, and *complete*, which are designed for resolving all the nodes in the batch. We limit our attention to complete resolution algorithms, which in general provide the ground for the design of partial resolution algorithms.

1.1 Motivations of this work

The design and analysis of efficient BRAs have been deeply studied in the literature, in particular for pure (non-CSMA) slotted channels. Landmarks in this area are the *splitting-tree* algorithms devised by Hayes [2] and Capetanakis [3], whose interest has been recently reawaken in relation with the RFID tags [4]. In [5], Gallager *et al.* proposed the FCFS algorithm, also known as Clipped Modified Binary Tree (CMBT) algorithm, which was successively improved by Mosely and Humblet in [6]. The basic idea behind all these algorithms consists in splitting the initial conflict set in smaller subsets and, then, resolving one subset at a time in a recursive way. The various schemes differ in the policy used to split the conflict set in subsets. A comprehensive overview of these collision-resolution algorithms can be found in [7]. In many practical cases, the number n of nodes in the initial conflict set, namely the *batch size* (or conflict multiplicity), is unknown. This lack of information generally prevents the optimal setting of the parameters that drive the splitting algorithms, resulting in some performance loss. To counteract this inefficiency, in [12] and [**?**] authors define hybrid algorithms consisting of two parts, the first devoted to the estimation of the batch size n, and the second to resolve the (residual) batch by using classical splitting-tree algorithms. More recently, Popovski *et al.* [13, 17] proposed an entire class of collision resolution algorithms that closely integrates the estimation and resolution phases. In particular, the Interval Estimation Collision Resolution (IECR) algorithm is proven to reach asymptotic efficiency comparable to FCFS for Poisson arrivals. Note that, although initially designed for pure slotted systems, these algorithms can be easily adapted to carrier-sense multiple access (CSMA) networks, as done in [8] for FCFS.

In the classical performance analysis of BRAs, the feedbacks transmission time is generally neglected or underestimated, being usually included in the slot duration or approximated to the time length of an idle slot [8]. In practical radio systems, however, each packet transmission (included feedbacks) takes a fixed

amount of time for basic operations, such as RX/TX switching, signal detection and synchronization, and so on. This time may represent a significant part of the overall packet transmission time, in particular when the transmission rate is large. For instance, with reference to the IEEE 802.11g standard, the transmission time of an acknowledgment frame (ACK) at the basic rate of 6 Mbit/s is approximately 20% of the transmission time of a 1500 byte-long data frame, sent at 54 Mbit/s. Therefore, the time cost of feedbacks may have non-negligible impact on the performance of BRAs. This observation makes questionable whether immediate feedback is actually the best strategy for the design of BRAs and motivates the investigation of the frame-based solutions presented in this report.

1.2 Novel contribution

In this report, we first propose a batch resolution scheme for CSMA-based wireless networks, named *Adaptive Batch Resolution Algorithm with Deferred Feedback* (ABRADE), which aims at minimizing the batch resolution time by reducing the impact of feedback messages. To this end, ABRADE waives the immediate feedback approach in favor of a framed-ALOHA contention scheme [9]. The resolution process works in successive *resolution rounds*, each consisting in a *contention frame* of a certain length. Feedback is returned only at the end of each round, through a single broadcast message called *probe*. The core of ABRADE consists in the dynamic adaptation of the frame length to the residual batch size, i.e., the number of still unresolved nodes after each round. The adaptation scheme is designed to minimize the overall time required to resolve the whole batch, inclusive of the time taken by idle sensing, collisions, successful transmissions and probe messages. The optimization problem is resolved by applying a dynamic programming argument, under the assumption of perfect knowledge of the residual batch size at each step of the algorithm.

Successively, we relax this assumption and propose ABRADE$^+$, a complete resolution scheme for batches of unknown size that is obtained by combining ABRADE with a suitable Batch Size Estimation (BSE) module. We compare ABRADE$^+$ with the best performing BRAs based on the immediate feedback paradigm proposed in the literature, namely FCFS, Sift, IECR and Sift/IECR. Results confirm that, in practical CSMA-based wireless networks, ABRADE$^+$ outperforms the existing solutions, both in the case of partial and no prior information on the batch multiplicity. The performance evaluation of the classical BRAs in practical scenarios can also been considered a side contribution of this work.

Summing up, the novelties in this report are the following.

- ABRADE: a frame-based BRA that applies a novel frame length adaptation strategy to minimize

the resolution cost of batches of known size;

- ABRADE$^+$: an extension of ABRADE, realized by combining ABRADE with a suitable BSE technique, that works even with batches of unknown size;
- The performance evaluation of ABRADE$^+$ and of the best performing BRAs based on the immediate feedback paradigm, in two practical scenarios.

1.3 Paper organization

The rest of the report is organized as follows. Sec. 2 details the system model and defines the main notation and the performance measures considered in this work. Sec. 3 provides an overview of the main BRAs with immediate feedback, here considered for comparison purposes. Sec. 4 describes ABRADE and derives the dynamic frame length optimization strategy that is the core of the algorithm. The mathematical and asymptotic analysis of ABRADE performance, under the assumption of perfect knowledge of the batch size, is developed in Sec. 5, whereas in Sec. 6 we introduce the BSE module and the ABRADE$^+$ scheme for batches of unknown size. In Sec. 7 we propose a simulative comparison of ABRADE$^+$ with the best performing BRAs with immediate feedback in two practical scenarios, which are based on the common IEEE 802.11g and IEEE 802.15.4 radio standard. Finally, in Sec. 8 we sum up the main results and discuss possible extensions of the proposed schemes.

Chapter 2

System model and performance measures

Before going in more depth in the analysis of BRAs, it is convenient to formally define the system model and the performance measures considered in this study. For reader convenience, furthermore, we collected in Tab. 2.1 the main notation used throughout the report.

2.1 System model

We consider a scenario where all nodes in the batch are within the transmission/reception range of the inquirer and capable of carrier-sensing each other transmission [10]. Channel access is ruled by the CSMA scheme: nodes are allowed to transmit only whether the channel has been idle for a certain time interval β, called *idle slot*. A basic CSMA scheme is used for managing channel access: nodes refrain from transmitting any time the energy level sensed on the medium is above a certain threshold. Conversely, nodes are allowed to transmit if the channel remains idle for a certain time interval, referred to as *idle slot*. If two or more nodes transmit simultaneously, their signals will interfere at the receiver generating a *collision* and none of them will be successfully decoded. For the sake of simplicity, we make the classical assumption of noiseless radio channel, so that packet transmissions are always successful, except in case of collision. Successful and collided transmissions take constant time β_s and β_c, respectively. As customary, we normalize all the time intervals to the transmission time of a data packet, so that we assume $\beta_s = 1$. The time axis is hence divided in slots of unequal duration, which are referred to as *idle*, *successful* or *collided* according to whether they host zero, one or more than one simultaneous transmission, respectively.

In the deferred-feedback case, feedback is provided only at the end of the frame through a probe message of duration β_p. In the immediate feedback case, instead, each slot is followed by a feedback. Let φ_i, φ_s, and φ_c denote the feedback transmission time in the case of idle, successful or collided slot,

respectively. In practical CSMA-based systems, idle feedback is implicitly provided by the carrier-sense mechanism, so that we have $\varphi_i = 0$. Conversely, successful feedback is typically provided by means of a short acknowledgement packet (ACK), so that it results $0 < \varphi_s < 1$. Finally, collided slots are recognized because the packet transmission is not followed by an ACK within a certain time interval that can, hence, be considered as the collision feedback time φ_c. A more accurate evaluation of these parameters for some practical scenarios is given in Appendix.

2.2 Performance measures

The aim of BRAs is to minimize the *batch resolution interval* (BRI), defined as the average time required to resolve all the nodes in the initial batch, here denoted by $\mathcal{T}(n)$ where n is the number of nodes in the batch, that is to say, the *batch size*. Another common performance metric to evaluate the effectiveness of BRAs is the *batch throughput*, defined as the average nodes resolution rate, i.e.,

$$\lambda(n) = \frac{n}{\mathcal{T}(n)}\,. \tag{2.1}$$

Clearly, $\lambda(n)$ and $\mathcal{T}(n)$ are related, since for a certain batch size n, maximizing $\lambda(n)$ is equivalent to minimizing $\mathcal{T}(n)$. By letting n approach infinity, we obtain the asymptotic batch throughput

$$\lambda_{\max} = \lim_{n\to\infty} \lambda(n)\,. \tag{2.2}$$

The asymptotic throughput is of interest since it corresponds to the maximal packet arrival rate that can be sustained by the system when the BRA is used as an *obvious MAC access scheme*, that is to say, when packets arriving to the system are served according to a gated policy, according to which all packets generated during the resolution of a batch are held in queue and processed as a new batch once the packets in the previous batch have been all resolved [8].

When the batch size is unknown at the inquirer, it is modeled as an integer random variable N, with probability mass distribution $P_N(n)$ and statistical mean $\mathrm{m}_N = \mathrm{E}[N]$. In this case, a useful metric is the *mean batch throughput*, which is defined as

$$\bar{\lambda}(\mathrm{m}_N) = \frac{\mathrm{E}[N]}{\mathrm{E}[\mathcal{T}(N)]} = \frac{\mathrm{m}_N}{\sum_{n=0}^{\infty} P_N(n)\mathcal{T}(n)} = \frac{\mathrm{m}_N}{\sum_{n=0}^{\infty} \frac{nP_N(n)}{\lambda(n)}}\,. \tag{2.3}$$

Table 2.1: Main notation.

β_s, β_c, β	time duration of successful, collided and idle slots
$\varphi_s, \varphi_c, \varphi_i$	time duration of the feedback message for successful, collided, idle slots in immediate feedback BRAs
β_p	time duration of probe message in ABRADE$^+$
n	current batch size
m_N	mean batch size
$\mathcal{T}(n)$	mean batch resolution interval with batch size n
$\lambda(n)$	throughput with batch size n
$\bar{\lambda}(\mathrm{m}_N)$	throughput with random batch size of mean m_N
$\lambda_{\max}$	asymptotic throughput
w	frame length (in number of slots)
$w_n^\star$	optimal contention frame length with batch size n
μ_∞	asymptotic mean number of transmissions per slot
S, C, I	number of successful, collided and idle slots in a frame
$P_{w,n}(s,c)$	joint probability mass distribution of S and C
$p_{w,n}(s)$	probability mass distribution of S

Chapter 3

BRAs with immediate feedback

In this section we quickly overview the features of the best-known BRAs for CSMA network, based on the immediate feedback model. The section serves a threefold objective: first, it provides a survey of the state of the art on the subject; second, it describes the algorithms used in the performance comparison with ABRADE$^+$; third, it bears out the motivation of this study by showing that, keeping into account the feedback costs, the asymptotic throughput of FCFS may be significantly lower than what reported in the classical literature.

3.1 The FCFS/CMBT algorithm

The first-come first-serve (FCFS) algorithm was proposed by Gallager in [5] as an obvious access scheme for Poisson packet arrivals. As for the other binary-tree algorithms [2, 3, 11], FCFS splits the initial batch in subsets, which are resolved sequentially. Batch splitting occurs according to the packets arrival time: time axis is divided in intervals of duration δ and nodes are resolved sequentially, starting from those that generated packets in the first interval, then in the second interval and so on. When the interval $I = t_0 + [0, \delta)$ is *activated* all packets arrived in that interval are transmitted in the current slot. In case of collision, the last activated time interval is split in two subintervals, namely the left interval $I_L = t_0 + [0, f\delta)$ and the right interval $I_R = t_0 + (f\delta, \delta]$, and I_L is activated in the next slot. The procedure is repeated recursively, till one node is resolved. Then, the right interval is activated and so on.

A simple improvement of the algorithm consists in avoiding *guaranteed* collisions that occur whenever an interval splitting following a collision leaves all the nodes in the right subinterval, whose activation will hence result again in a collision. Therefore, anytime a collided slot is followed by an empty slot, which indicates that the left subset was empty, the right interval is immediately split in two new subintervals and

the left one is activated. The splitting procedure is repeated for any successive idle slot, thus progressively reducing the length of the activated interval, till the first successful or collided slot. At this point, the original algorithm execution is resumed and the resolution process proceeds as usual.

Another improvement is brought about by the *clipping* mechanism that aims at avoiding the activation of intervals of suboptimal size. The rationale is that, for Poisson arrivals, two (or more) collisions in a run "erase" any prior information on the number of nodes in the right subsets. Therefore, after a left subinterval is resolved, the clipping mechanism consists in returning the right subintervals to time axis and activating a time interval of optimal length δ, rather than a shorter interval. (Note, that this rationale is valid only under the assumption of Poisson arrivals [8].) The version of FCFS that includes these improvements, also known as *clipped modified binary tree* (CMBT), is the fastest conflict resolution algorithm in the literature for Poisson arrivals [6].

3.1.1 The impact of feedback costs

in [8], Gallager characterizes FCFS for CSMA channels and derive an approximate expression of the achievable asymptotic throughput $\lambda_{\max}$, under the classical assumption

$$\varphi_i = 0\,, \quad \varphi_c = \varphi_s = \beta \ll 1\,. \tag{3.1}$$

Unfortunately, (3.1) may not hold in practical systems, where feedback times are not negligible, so that the performance analysis may yield turn out to be overoptimistic. We thus revised the derivation of $\lambda_{\max}$ proposed in [8] under the assumption that φ_s and φ_c may be non-negligible. Doing so, we obtained the following result

$$\lambda_{\max} \simeq \frac{g + g^2}{2\beta + (1+\varphi_s)(g+g^2)}\,, \tag{3.2}$$

where

$$g = \sqrt{\frac{2\beta}{1+\varphi_c+\sqrt{\beta}}} \tag{3.3}$$

is the mean number of transmissions per slot. Furthermore, the optimal value of f turned out to be equal to

$$f = -\beta + \sqrt{a^2+a}\,, \quad \text{with} \quad a = \frac{\beta}{1-\beta+\varphi_c}\,. \tag{3.4}$$

Note that, setting the parameters as in (3.1), the expression (3.2) returns the original expressions provided by Gallager in [8]. However, if $\varphi_s > \beta$, as in many practical systems, the maximal throughput given by (3.2) becomes notably lower than the value found in literature, in particular for systems with high transmission rates, for which $\varphi_s \to \beta_s = 1$.

We remark that FCFS has been designed to resolve conflicts among messages that arrive according to a *Poisson* process with *known* arrival rate λ. Therefore, the plain application of FCFS algorithm in case of batch arrivals yields poor performance [12]. However, the optimality of the algorithm can be easily re-established for batch arrivals, provided that the batch size is Poisson distributed with known mean m_N. In fact, as explained in [13], it suffices that each node in the batch generates a random *virtual arrival instant* in the time interval $[0, \mathrm{m}_N/\lambda_{\max}]$. The FCFS algorithm can be then applied by considering the virtual arrival epochs in place of the actual message arrival times.

3.2 The IECR algorithm

FCFS optimality requires to know beforehand the mean batch size m_N, an information that may not be available in many practical situations. To overcome this limit, Popovsky *et al.* proposed in [13] the Interval Estimation Collision Resolution (IECR) algorithm. The basic idea consists in estimating the batch size while the batch resolution is in progress. In practice, IECR starts with all nodes setting their virtual arrival epochs uniformly in the unit interval $(0, 1)$ and the FCFS algorithm is performed. In case of collision, the last activated interval is split in two subintervals, the left one corresponding to a fraction f of the original interval. The algorithm then proceeds by activating the left subinterval, as in FCFS. When a collision is successfully solved, however, the algorithm activates a new interval of length

$$\delta_{IECR} = \frac{g}{\hat{\lambda}}\,, \tag{3.5}$$

where g is as in (3.3), whereas $\hat{\lambda}$ is the current estimate of the average arrival rate given by

$$\hat{\lambda} = \frac{k}{f_{lim}}\,,$$

where k and p_{lim} are the number of nodes already resolved and the length of the resolved interval, respectively. IECR is proved to reach the same asymptotic throughput of FCFS, without requiring any *a priori* knowledge of the batch size.

3.3 The Sift and Sift/IECR algorithms

Another interesting approach to collision resolution in CSMA channels is proposed in [14], where authors introduce the Sift algorithm. Sift differs from the other BRAs in two aspects: first, it aims at

minimizing the time till the *first successful* transmission, rather than the batch resolution interval; second, channel access is managed according to a modified framed ALOHA scheme, still with immediate feedback, rather than resorting to the splitting approach.

More specifically, time is divided in frames, each consisting of $w = 32$ slots. At the beginning of a frame, each unresolved node chooses at random one slot in the frame where to transmit. Slot $j \in \{1, 2, \ldots, w\}$ is picked with probability

$$p_{Sift}(j) = \frac{(1-\alpha)\alpha^w}{1-\alpha^w}\alpha^{-j}\,, \tag{3.6}$$

where $0 < \alpha < 1$ is a parameter that depends on the maximum expected batch size value $N_{\max}$. The algorithm ends at the first transmission, which always occurs within the frame, for $\sum_{j=1}^{w} p_{Sift}(j) = 1$. The probabilities (3.6) are designed in order to maximize the chance that the first transmission is successful, regardless of the batch size.[1]

Although designed for different purposes, Sift may be easily adapted to solve conflicts in CSMA channels. In fact, each execution of Sift will end with either a resolved node or a collision. In both cases, it suffices to reapply the algorithm to the residual batch, till all nodes are resolved.

Alternatively, Popovsky suggested to merge Sift with IECR, thus generating the hybrid Sift/IECR algorithm. The idea is to use Sift to get a first estimate of the batch size and, then, apply IECR. Suppose that Sift ends at slot $m \in [1, w]$ and let

$$f_{lim} = \sum_{j=1}^{m} p_{Sift}(j)\,, \quad f_{low} = f_{lim} - p_{Sift}(m)\,.$$

If Sift ends with a successful transmission, then IECR is started having the interval $[0, f_{lim})$ resolved, and $k = 1$. Conversely, if Sift ends with a collision, then IECR is started as if a collision occurred in the interval (f_{low}, f_{lim}). The Sift/IECR algorithm is expected to yield better performance than IECR, in particular for small values of the batch size.

[1] Actually, Sift was engineered for $N_{\max} = 512$ nodes, for which $\alpha = 512^{-1/31}$ that is the value considered in this report. However, authors showed that performance degrades gracefully when the batch size exceeds this limit.

Chapter 4

ABRADE

In this section we describe ABRADE and the dynamic frame length adaptation strategy that is the core of the algorithm. The theoretical analysis of ABRADE is developed under the assumption of deterministic and known value of the batch size n. The extension to the more realistic case of unknown n will be discussed in Sec. 6.

As mentioned in the introduction, ABRADE works in successive *resolution rounds*. In each round, every single unresolved node, independently of the others, picks a random slot in the *contention frame* of w slots, with uniform probability, and transmits its packet in that slot, according to a CSMA-based framed ALOHA scheme. At the end of the contention frame, the inquirer broadcasts a *probe message* that carries the aggregate feedback field (ACK), together with other control information, included the frame length w to be used in the next round. Acknowledged nodes are resolved and leave the batch, whereas the others transmit in the next round. The batch resolution process is completed when all nodes have been resolved.

4.1 Dynamic optimization of the contention frame length

The core of ABRADE consists in adjusting the frame length w after each resolution round, according to the residual batch multiplicity n. The adaptation aims at maximizing the batch throughput $\lambda(n)$ given by (2.1) that, in turn, corresponds to minimizing $\mathcal{T}(n)$. In other words, we wish to find the frame length $w_n^\star$, such that

$$w_n^\star = \arg\min_w \{\mathcal{T}(n)\} \ . \tag{4.1}$$

The BRI $\mathcal{T}(n)$ is given by

$$\mathcal{T}(n) = \mathrm{E}\left[\tau(n)\right] \ , \tag{4.2}$$

where $\tau(n)$ is the actual time required to resolve a batch of size n. Let S, C and I denote the number of successful, collided and idle slots, respectively, in a frame of w slots, so that

$$S + C + I = w\,. \tag{4.3}$$

The conditional expectations of S, C and I, given w and n, can be easily determined as

$$\begin{aligned} \mathrm{E}\left[S|w,n\right] &= wnq(1-q)^{n-1}\,; \\ \mathrm{E}\left[C|w,n\right] &= w\Big[1 - nq(1-q)^{n-1} - (1-q)^{n}\Big]\,; \\ \mathrm{E}\left[I|w,n\right] &= w(1-q)^{n}\,; \end{aligned} \tag{4.4}$$

where for ease of notation we set

$$q = \frac{1}{w}\,. \tag{4.5}$$

The round duration, denoted by $y_{w,n}$, can then be written as

$$y_{w,n} = S + C\beta_c + I\beta + \beta_p\,, \tag{4.6}$$

whereas $\tau(n)$ can be expressed in the following recursive form

$$\tau(n) = y_{w,n} + \tau(n - S)\,, \tag{4.7}$$

with $\tau(0) = 0$.

Now, let $P_{w,n}(s,c) = \mathrm{P}\left[S = s, C = c|w,n\right]$ be the joint probability mass distribution of S and C with a frame of w slots and n transmitting nodes. The expression of $P_{w,n}(s,c)$ can be obtained by basic combinatorial analysis [15] or, more conveniently, through the following recursive form [16]

$$\begin{aligned} &P_{w,n}(s,c) = 0\,; \quad \text{for} \quad n < 0 \text{ or } s < 0 \text{ or } c < 0 \text{ or } w \leq 0\,; \\ &P_{w,n}(s,c) = 1\,; \quad \text{for} \quad n = 0, s = 0, c = 0 \text{ or } n = 1, s = 1, c = 0\,; \\ &P_{w,n}(s,c) = P_{w,n-1}(s,c)\frac{c}{w} + P_{w,n-1}(s,c-1)\frac{s+1}{w} + P_{w,n-1}(s-1,c)\frac{w-c-s+1}{w}\,; \quad \text{for } n > 1\,. \end{aligned} \tag{4.8}$$

Furthermore, let $p_{w,n}(s) = \mathrm{P}\left[S = s|w,n\right]$ be the probability mass distribution of S, given w and n, which can be obtained from (4.8) through the marginal law:

$$p_{w,n}(s) = \sum_{c=0}^{w-s} P_{w,n}(s,c)\,. \tag{4.9}$$

Replacing (4.7) in (4.2) and applying the total probability theorem with respect to S, we obtain

$$\begin{aligned} \mathcal{T}(n) &= \mathrm{E}\left[y_{w,n}\right] + \sum_{s=0}^{\infty} \mathrm{E}\left[\tau(n-S)|S=s\right] p_{w,n}(s) \\ &= \frac{\mathrm{E}\left[y_{w,n}\right] + \sum_{s=1}^{\infty} \mathcal{T}(n-s) p_{w,n}(s)}{1 - p_{w,n}(0)}\,. \end{aligned} \tag{4.10}$$

Then, using (4.6) into (4.10) and recalling (4.4) we can express (4.1) in the following recursive Bellman-equation form[1]

$$w_n^\star = \arg\min_w \left\{ \frac{\beta_p + w\beta_c + wnq(1-q)^{n-1}(1-\beta_c) + w(1-q)^n(\beta-\beta_c) + \sum_{s=1}^{\infty} p_{w,n}(s)\mathcal{T}^\star(n-s)}{1-p_{w,n}(0)} \right\} \tag{4.11}$$

where $\mathcal{T}^\star(k)$ denotes the optimal BRI for a batch of size k. This minimization problem can be solved recursively, starting from $k=1$, with $\mathcal{T}^\star(1) = 1+\beta_p$ and $w_1^\star = 1$, and using (4.10) to obtain $\mathcal{T}^\star(k)$ for $k > 1$.

[1]For ease of notation we extend the sum to infinity, though $p_{w,n}(s) = 0$ for $s > \min\{w, n\}$.

Chapter 5

Performance Analysis of ABRADE with known batch size

In this section we analyze the performance of ABRADE under the assumption that the batch size is known beforehand, so that the optimal frame length can be used throughout the entire batch resolution process.

To begin with, in Fig. 5.1 we plot the optimal frame length $w_n^\star$ as a function of the batch size n for different values of β, with $\beta_c = 1$, $\beta_p = 1$ (solid lines) and $\beta_p = 2$ (dashed lines). We can observe that $w_n^\star$ grows almost linearly with n, though the optimal contention frame is proportionally longer for small batches than for large ones. The reason is that, with small batches, the cost of probe messages is determinant, so that it is advantageous to choose longer frames to reduce the collision probability and, in turn, the number of resolution rounds required to resolve the batch, though at the cost of a larger number of idle slots.

In Fig. 5.2 we report the throughput curves as a function of β, for different values of n. All curves have been obtained by setting $\beta_p = \beta_c = 1$. First, we note that the throughput increases with n, which confirms that the overhead due to the control traffic becomes progressively less relevant as the size of the batch grows. Second, we note that the throughput curves rapidly converge toward the superior limit $\lambda_{\max}$, represented by the dashed curve in the figure. To determine an analytical expression of $\lambda_{\max}$, we consider that, for sufficiently large n, it holds

$$\mathcal{T}^\star(n) \simeq \frac{n}{\lambda_{\max}} . \tag{5.1}$$

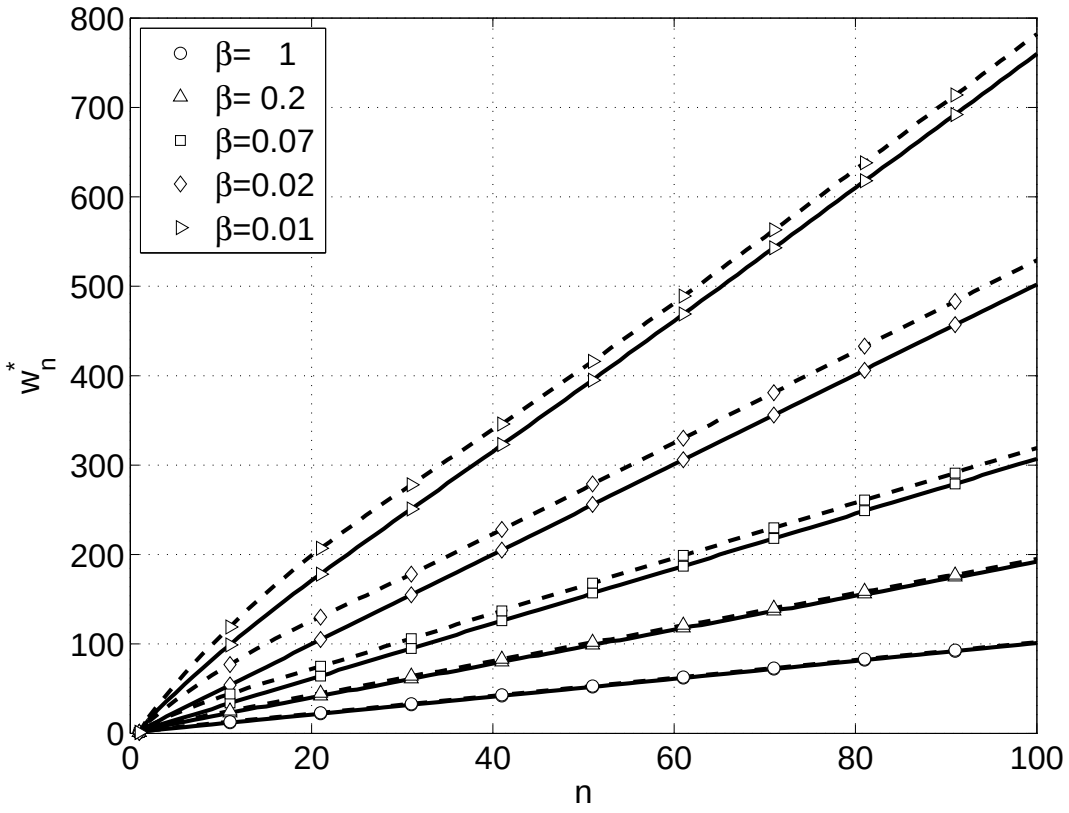

Figure 5.1: Optimal frame length $w_n^\star$ as a function of the batch size n, for different values of β, with $\beta_c = 1$, $\beta_p = 1$ (solid) and $\beta_p = 2$ (dashed).

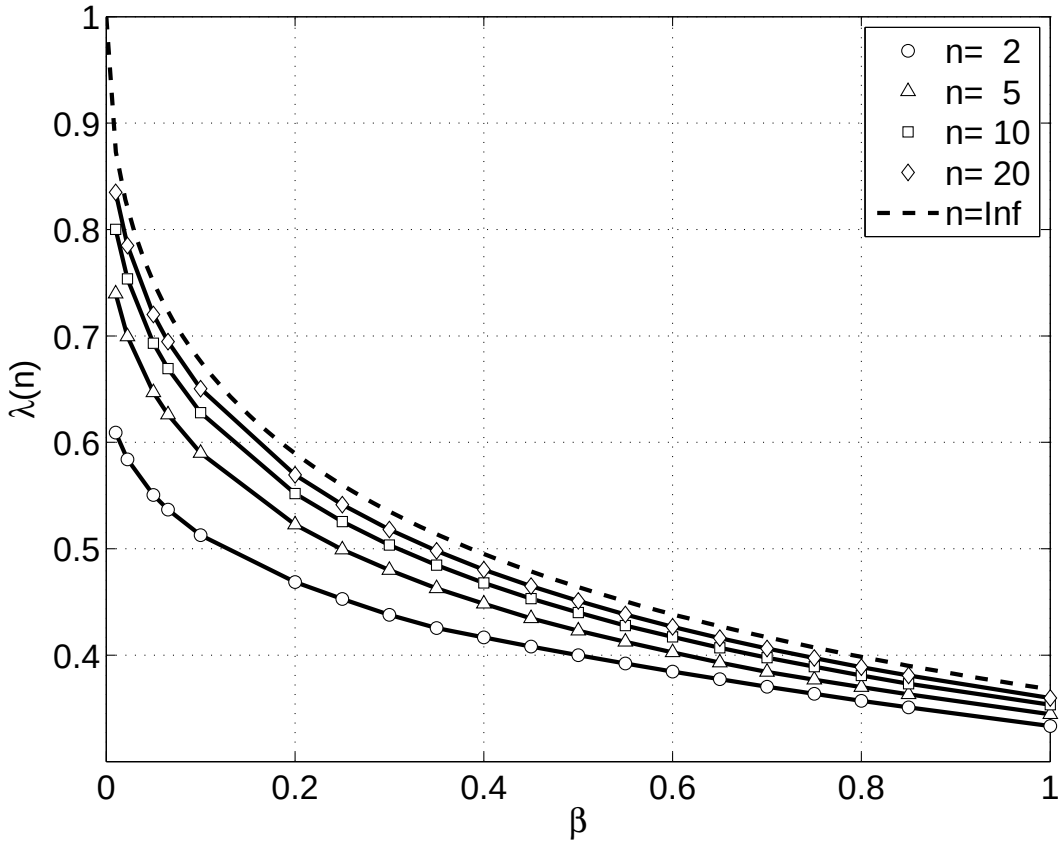

Figure 5.2: Throughput $\lambda(n)$ *vs* β, for different values of the batch size n, with $\beta_p = \beta_c = 1$. Dashed curve represents $\lambda_{\max}$.

From (4.10) we hence obtain

$$\begin{aligned}\frac{n}{\lambda_{\max}} &\simeq \mathrm{E}\left[y_{w_n^\star,n}\right] + \sum_{s=0}^{\infty} \frac{n-s}{\lambda_{\max}} p_{w_n^\star,n}(s) \\ &\simeq \mathrm{E}\left[y_{w_n^\star,n}\right] + \frac{n}{\lambda_{\max}} - \frac{\mathrm{E}\left[S\right]}{\lambda_{\max}}\,,\end{aligned} \tag{5.2}$$

from which

$$\lambda_{\max} \simeq \frac{\mathrm{E}\left[S\right]}{\mathrm{E}\left[y_{w_n^\star,n}\right]}\,. \tag{5.3}$$

Recalling (4.4) and (4.6), (5.3) can be written as

$$\lambda_{\max} \simeq \frac{w_n^\star n q_n^\star (1-q_n^\star)^{n-1}}{\beta_p + w_n^\star (1-q_n^\star)^n (\beta - \beta_c) + w_n^\star \beta_c + w_n^\star n q_n^\star (1-q_n^\star)^{n-1}(1-\beta_c)}\,, \tag{5.4}$$

where $q_n^\star = 1/w_n^\star$. Now, let μ_n denote the mean number of transmissions per slot, equal to

$$\mu_n = n q_n^\star\,, \tag{5.5}$$

and μ_∞ be its asymptotic value

$$\mu_\infty = \lim_{n\to\infty} \mu_n\,. \tag{5.6}$$

Letting n approach infinity, we thus have

$$\lim_{n\to\infty} (1-q_n^\star)^n = \lim_{n\to\infty} \left(1 - \frac{\mu_n}{n}\right)^n = e^{-\mu_\infty}\,,$$

so that, from (5.4) we obtain

$$\lambda_{\max} = \frac{\mu_\infty e^{-\mu_\infty}}{b_p + \beta_c + e^{-\mu_\infty}(\beta - \beta_c) + \mu_\infty e^{-\mu_\infty}(1-\beta_c)}\,, \tag{5.7}$$

where b_p accounts for the fact that, the longer the frame, the longer the aggregate ACK field and, in turn, the size of the probe message.[1] It thus remains to determine the expression of μ_∞, which is the average number of transmissions per slot that maximizes the asymptotic throughput $\lambda_{\max}$. Setting to zero the derivative of the right-hand side of (5.7) with respect to μ_∞, after some algebra we find the following transcendent equation

$$\mu_\infty = 1 - \frac{\beta_c - \beta}{b_p + \beta_c} \exp(-\mu_\infty)\,, \tag{5.8}$$

which can be easily solved using numerical methods, such as bisection. Note that μ_∞ strictly depends on β, spanning from 0 to 1 as β varies from β_c to 0, as shown in Fig. 5.3. This behavior reflects the principle according to which the channel access strategy shall be less aggressive when idle slots are much shorter than busy slots.

[1] As we will see in Appendix, b_p is generally negligible in practical scenarios.

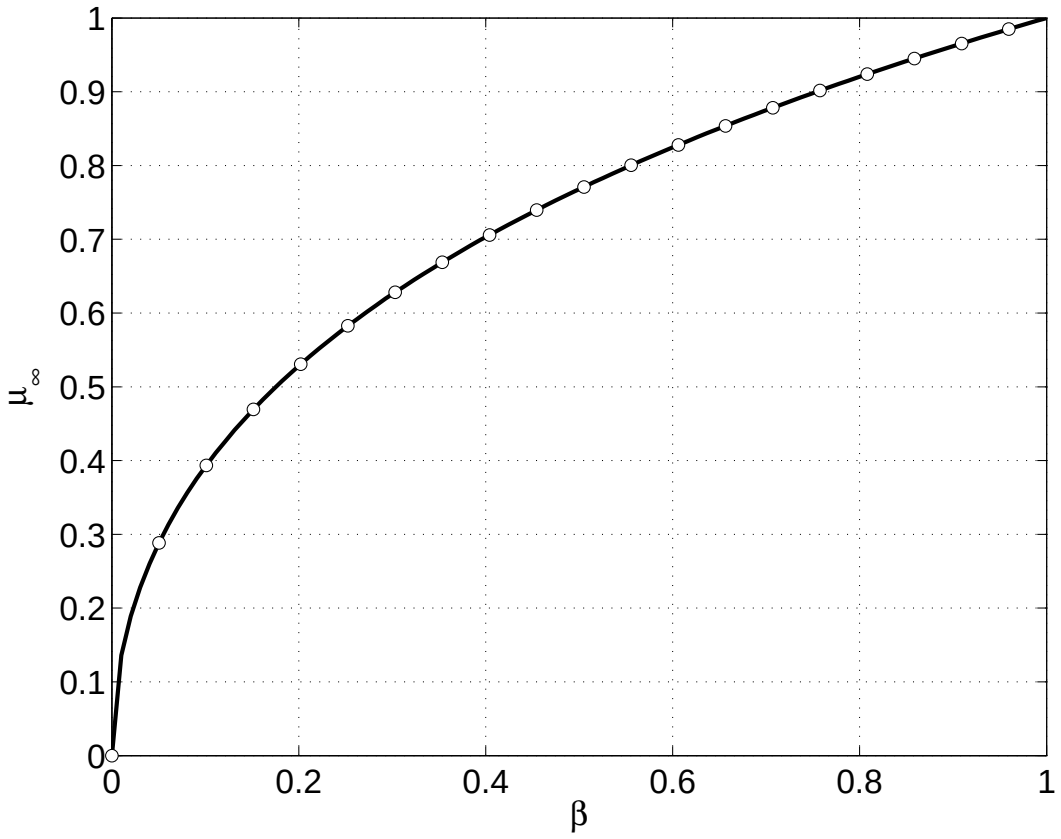

Figure 5.3: Asymptotic number of transmissions per slot, μ_∞, as a function of the normalized idle slot duration β, with $\beta_c = 1$, $b_p = 0$.

Replacing (5.8) into (5.7) we finally obtain

$$\boxed{\lambda_{\max} = \frac{e^{-\mu_\infty}}{b_p + \beta_c + e^{-\mu_\infty}(1-\beta_c)}\,,} \tag{5.9}$$

which is the superior limit of ABRADE throughput, represented by the dashed curve in Fig. 5.2.

Chapter 6

ABRADE$^+$: extending ABRADE to batches of unknown size

In many practical cases, the exact number of nodes in the conflict set may not be known beforehand. In this case, ABRADE needs to be coupled with a BSE module that provides an estimate $\hat{n}$ of the number of nodes in the batch. This estimation problem is interesting by its own and has received considerable attention in the literature [12, 13, 17]. Most of these estimation schemes assume immediate feedback, so that they are not directly applicable to ABRADE. BSE schemes for framed ALOHA systems have been investigated, for instance, in [18] and [19]. In both cases, the goal was to minimize the number of slots (of fixed length) required to attain high estimation accuracy. In our study, instead, we are interested in an estimate algorithm that is capable of driving the ABRADE optimization procedure, rather than providing extremely accurate estimate.

We hence propose a novel estimate procedure that, integrated with ABRADE adaptation scheme, provides a complete solution for the batch resolution problem that will be referred to as ABRADE$^+$.

ABRADE$^+$ inherits the contention scheme and the frame length adaptation strategy of ABRADE, with two important add ons: the *batch size estimate function* (BSEF) and the *contention probability* p.

The BSEF is invoked after each resolution round to get an updated estimate of the residual batch size, which is then used in place of the exact value of n to select the frame length for the next round, as in ABRADE. As a matter of fact, when the batch size estimate converges to the exact value, ABRADE$^+$ works exactly as ABRADE.

The contention probability p, which is broadcasted in the probe message together with the frame length w, determines the probability that a node in the batch transmits in the upcoming round. As we will see later, this parameter is particularly useful at the very beginning of the resolution process, when the estimate

of the batch size is totally unreliable.

In Fig. 6.1 we provide the pseudo-code of ABRADE$^+$ for both inquirer and nodes, whereas in the following, we first detail the BSEF and, then, discuss the estimate start up problem.

6.1 The batch size estimate function

The BSEF, denoted by $\mathcal{H}_{w,p}(s,c)$, is a deterministic function that takes as inputs the frame length w, the contention probability p, and the number s and c of successful and collided slots at the end of the frame, and returns an estimate $\hat{n}$ of the batch size given by

$$\hat{n} = \mathcal{H}_{w,p}(s,c) = \frac{\hat{n}_t}{p} \tag{6.1}$$

where $\hat{n}_t$ is the estimate of the number of nodes that have actually transmitted in the frame.[1] This estimate, in turn, is equal to

$$\hat{n}_t = s + c\, n_c\,, \tag{6.2}$$

where $n_c \geq 2$ denotes the mean number of nodes involved in each collision. To determine n_c, we assume that the number of transmissions per slot can be approximated by a Poisson-distributed random variable with parameter μ, as done in [18]. In this case, n_c can be expressed as

$$n_c = \frac{\mu - \mu\exp(-\mu)}{1 - \exp(-\mu) - \mu\exp(-\mu)}\,, \tag{6.3}$$

that, replaced in (6.2), gives

$$\hat{n}_t = s + c\,\frac{\mu - \mu\exp(-\mu)}{1 - \exp(-\mu) - \mu\exp(-\mu)}\,. \tag{6.4}$$

The value of μ is finally obtained by equalling (6.4) to μw, which is the expected number of transmissions in the frame, so that we obtain

$$s + c\frac{\mu - \mu\exp(-\mu)}{1 - \exp(-\mu) - \mu\exp(-\mu)} = \mu w\,. \tag{6.5}$$

Although transcendent, (6.5) is well behaved and admits a single positive solution $\hat{\mu}$ that can be quickly determined by using standard methods, as bisection search. Hence, from (6.4) and (6.5), it follows

$$\hat{n}_t = \hat{\mu} w\,, \tag{6.6}$$

that applied to (6.1) yieds

$$\boxed{\hat{n} = \mathcal{H}_{w,p}(s,c) = \frac{\hat{\mu} w}{p}\,,} \tag{6.7}$$

[1] For compactness, we omit to indicate the dependency of $\hat{n}$ and $\hat{n}_t$ on s, c, w, and p.

//* ***Inquirer side*** *//

Input: m_N

Set w_0 and p as in (6.10) and (6.14), respectively.

ACK $\leftarrow null$ //* *Clear feedback field* *//

$w \leftarrow w_0$

$unresolved \leftarrow$ **true**

while $unresolved$ **do**

`send_probe_pck(`w`,`p`,ACK)`

$\{s, c, i\} \leftarrow$`receive_from_channel(`w`)`

$\hat{n} \leftarrow \mathcal{H}_{w,p}(s, c)$ as in (6.7)

$n_{est} \leftarrow \lceil \hat{n} - s \rceil$ //* *estimate residual batch size* *//

if $n_{est} = 0$ **then**

if $p < 1$ **then**

//* *See Sec. 6.2.3* *//

$p \leftarrow 1, w \leftarrow w^{\star}_{n_0}$ with n_0 as in (6.15)

else

$unresolved \leftarrow$ **false**//* *The batch is resolved* *//

end if

else

//* *Dynamically adjust the frame length according to (4.11)* *//

$p \leftarrow 1, w \leftarrow w^{\star}_{\hat{n}}$

end if

ACK $\leftarrow s$ //* *Update ACK field* *//

end while

//* ***Node side*** *//

Input: $\{w, p,$ACK$\}$ = `receive_probe_pck`

$unresolved \leftarrow$ **true**

while $unresolved$ **do**

$k \leftarrow 0$

$compete \leftarrow$ (`rand` $\leq p$) //* *Compete with probability* p *//

if $compete$ **then**

$k = \lceil$`rand` $\times w \rceil$ //* *Choose a random slot* *//

`receive_from_channel(`$k-1$`)` //* *Wait for* $k-1$ *slots* *//

`send_data_pck()` //* *Send packet on the selected* k*th slot* *//

end if

$\{w, p,$ACK$\}$ = `receive_probe_pck`

if slot k ACKed **then**

$resolved \leftarrow$ **true**

end if

end while

Figure 6.1: ABRADE^{+} pseudocode at the inquirer and node side.

where $\hat{\mu}$ is the solution of (6.5).

For space constraints we cannot present the performance analysis of (6.6) and (6.7). However, we observed that the statistical mean and power of the estimate error are lower for (6.6) than for the other estimate methods proposed in [18], which consider separately the number of idle and collided slots in the frame. Furthermore, the estimate provided by (6.6) is always bounded, even when all the slots in the contention frame are collided, thus avoiding the divergence problem that affects other estimate techniques [18].

Here, we will be satisfied by discussing a single parameter that is of particular interest for our analysis, namely the *operating range* $n_\delta(w)$. Formally, we defined $n_\delta(w)$ as

$$n_\delta(w) = \arg\max_{n^\star} \left\{ \varepsilon_n < \delta \,,\ \forall n \leq n^\star \right\} \,,$$

where ε_n is the absolute mean relative estimate error, given by

$$\varepsilon_n = \left| \frac{\mathrm{m}_{\hat{n}} - n}{n} \right| \tag{6.8}$$

with

$$\mathrm{m}_{\hat{n}} = \mathrm{E}\left[\hat{n}\right] = \sum_{s=0}^{w} \sum_{c=0}^{w-s} \mathcal{H}_w(s,c) P_{w,n}(s,c) \,.$$

In practice, for sufficiently small δ, the operating range gives the maximum value of n for which the estimate is *good*, in the sense that $\varepsilon_n < \delta$ and the estimate can be reasonably assumed unbiased. In Fig. 6.2 we report the operating range $n_\delta(w)$ when varying the frame length w, for different values of the relative error threshold δ. The lowest curve in Fig. 6.2 gives the operating range for which ε_n is practically negligible. We can first note that this curve grows quasi-linearly with w, thus the longer the contention frame, the wider the range of n within which the estimate is good. Second, we observe that $\frac{n_\delta(w)}{w} \simeq 1.5$, which means that the estimate is practically unbiased whenever the average number of transmissions per slot is below 1.5, a condition that is rapidly satisfied when ABRADE's adaptation scheme approaches its regime behavior, as guaranteed by (5.8).

6.2 The estimate startup problem

At the very beginning of the resolution process, when the inquirer has no or only partial information about the batch multiplicity, the frame length has to be set to an arbitrary value w_0. In this condition of uncertainty, it is convenient to choose a small value of w_0, in order to have a first estimate of the batch size in the shortest time possible. However, if the batch is large, the frame may experience a lot of collisions

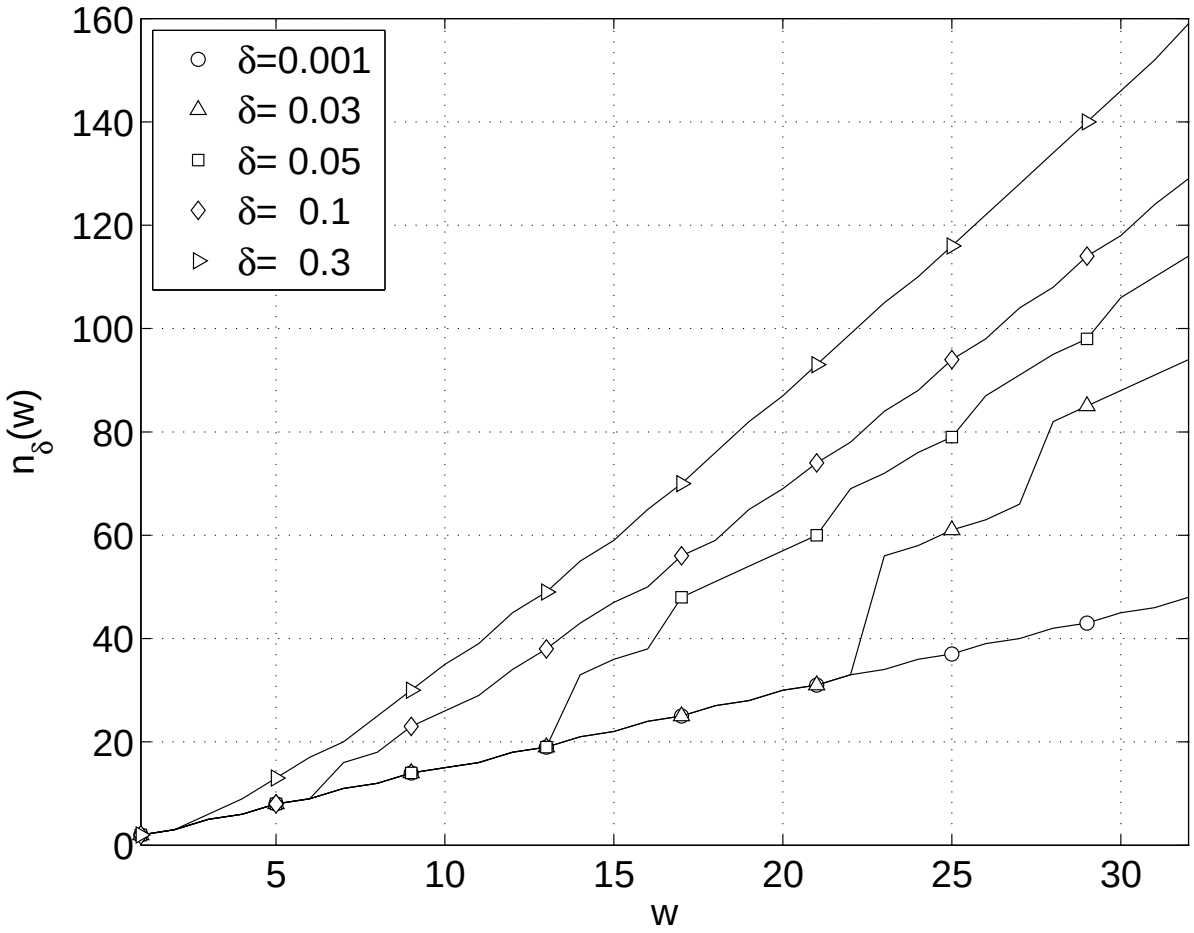

Figure 6.2: Operating range $n_\delta(w)$ *vs* frame length w, for different values of the relative error threshold δ.

that will inflate the BRI. To reduce the risk of collisions, ABRADE$^+$ statistically reduces the number of transmitting nodes through the contention probability p. The choice of p and w_0 shall strike a balance between the performance loss incurred in the first resolution round, due to suboptimal parameters setting, and the accuracy of the estimate $\hat{n}$ given by (6.7) at the end of the round. In the following we describe a possible strategy to set p and w_0 in order to cut this tradeoff.

6.2.1 Minimizing performance loss during the estimate start up period

from the performance analysis developed in Sec. 5, we know that the maximum throughput with batch size n is achieved when the mean number of transmissions per slot is equal to μ_n, given by (5.5). Now, let us assume that the initial batch multiplicity is a random variable N with probability mass distribution $P_N(n)$ and mean $\mathrm{m}_N = \mathrm{E}[N]$. Then, the contention probability p may be reasonably set in such a way that the expectation of number of transmissions per slot in the first frame is equal to the statistical mean of μ_n, i.e.,

$$\mathrm{E}\left[\frac{Np}{w_0}\right] = \mathrm{E}[\mu_n] \,, \tag{6.9}$$

from which it follows

$$p = w_0 \frac{\mathrm{E}[\mu_n]}{\mathrm{m}_N} = w_0 \frac{\sum_{n=0}^{\infty} P_N(n)\mu_n}{\mathrm{m}_N} \simeq \frac{w_0 \mu_\infty}{\mathrm{m}_N} \,, \tag{6.10}$$

where the right-most expression holds for $\mathrm{m}_N \gg 1$. Clearly, if $w_0 \,\mathrm{E}[\mu_n] > \mathrm{m}_N$, the value of p must be limited to 1.

6.2.2 First-round estimate accuracy

it remains to choose w_0 in order for (6.7) to provide a sufficiently accurate estimate $\hat{n}$ of the actual batch size after the first resolution round. To this end, we introduce the mean square estimation error that, for a batch size n, is defined as

$$\varepsilon^2(n) = \sum_{m=0}^{n} P_M(m|n) \sum_{s=0}^{w} \sum_{c=0}^{w-s} P_{w,m}(s,c)(\hat{n}-n)^2 \tag{6.11}$$

where $\hat{n}$ is as in (6.7), whereas

$$P_M(m|n) = \sum_{m=0}^{n} \binom{n}{m} p^m (1-p)^{n-m}, \quad m = 0, 1, \ldots, n \,. \tag{6.12}$$

is the probability mass distribution of the number of nodes that do transmit in the first slot. We then set w_0 such that $\mathrm{E}[\varepsilon^2(n)] / {\mathrm{m}_N}^2 < \Delta$, which is to say

$$w_0 = \min\left\{ w : \frac{\sum_{n=0}^{\infty} P_N(n)\varepsilon^2(n)}{{\mathrm{m}_N}^2} \leq \Delta \right\}, \tag{6.13}$$

where the parameter Δ, dubbed *normalized mean square estimate error threshold*, is a designed parameter. Clearly, (6.13) has to be evaluated by setting p is as in (6.10). We observed that, setting p as in (6.10), the estimate provided by (6.7) turns out to be unbiased with high probability. In this case, (6.13) simplifies in

$$w_0 = \min\left\{w : \sum_{n=0}^{\infty} P_N(n)\mathrm{M}_{\hat{n}} \leq (1+\Delta)\mathrm{m}_N{}^2\right\}. \tag{6.14}$$

This minimization problem can be solved numerically, starting from $w = 0$ and adding one till the inequality is satisfied. As an example, in Fig. 6.3 we report the result obtained when varying Δ and m_N for a uniform distribution of N from 0 to $N_{\max} - 1$, with $\beta = 0.0225$ (black markers) and $\beta = 0.0654$ (white markers) taken from the case study scenarios that will be described in Sec. 7. It is interesting to note that w_0 grows very quickly for $\Delta < 0.5$, which means that any further improvement of the estimate accuracy beyond this threshold would require a much larger increment of the frame length.

Furthermore, as long as p can be optimized in function of w_0, which is to say, (6.10) is less than 1, the value of w_0 turns out to be basically independent of m_N. The reason is that (6.13) considers a *normalized* error measure, so that the tolerable estimate error scales with m_N.

6.2.3 Empirical fallbacks

the estimate startup phase requires some empirical adjustments to avoid pitfalls. As a matter of fact, without any *a priori* information about the batch size, the choice of $P_N(\cdot)$ and m_N in (6.10) and (6.13) is arbitrary. Commonly, in this situation, N is considered a uniform random variable that take values in the integer interval $[0, N_{\max} - 1]$, with mean $\mathrm{m}_N = (N_{\max} - 1)/2$. If m_N is much larger than the actual batch multiplicity n, then (6.10) will return a small value of p that, in turn, may yield to an "empty" round, that it to say, a round where no nodes transmit. In this case, (6.7) will give $\hat{n} = 0$ even if $n > 0$. Hence, when a round with $p < 1$ ends with no transmissions, ABRADE$^+$ shall perform another resolution round with $p = 1$ to verify whether the batch is actually empty or not. The contention frame in this round is set to $w = w^{\star}_{n_0}$, where n_0 is the maximum value of the batch that may yield an empty round with probability larger than a prefixed value P_{thr}, called *empty-batch threshold*. Mathematically, we thus have

$$n_0 = \min\left\{n : \mathrm{P}\left[N \leq n | S = 0, C = 0\right] \geq P_{thr}\right\}. \tag{6.15}$$

Using the Bayes' rule and the total probability theorem, (6.15) can be expressed as

$$n_0 = \min\left\{n : \frac{\sum_{k=0}^{n} P_N(k)(1-p)^k}{\sum_{k=0}^{\infty} P_N(k)(1-p)^k} \geq P_{thr}\right\}, \tag{6.16}$$

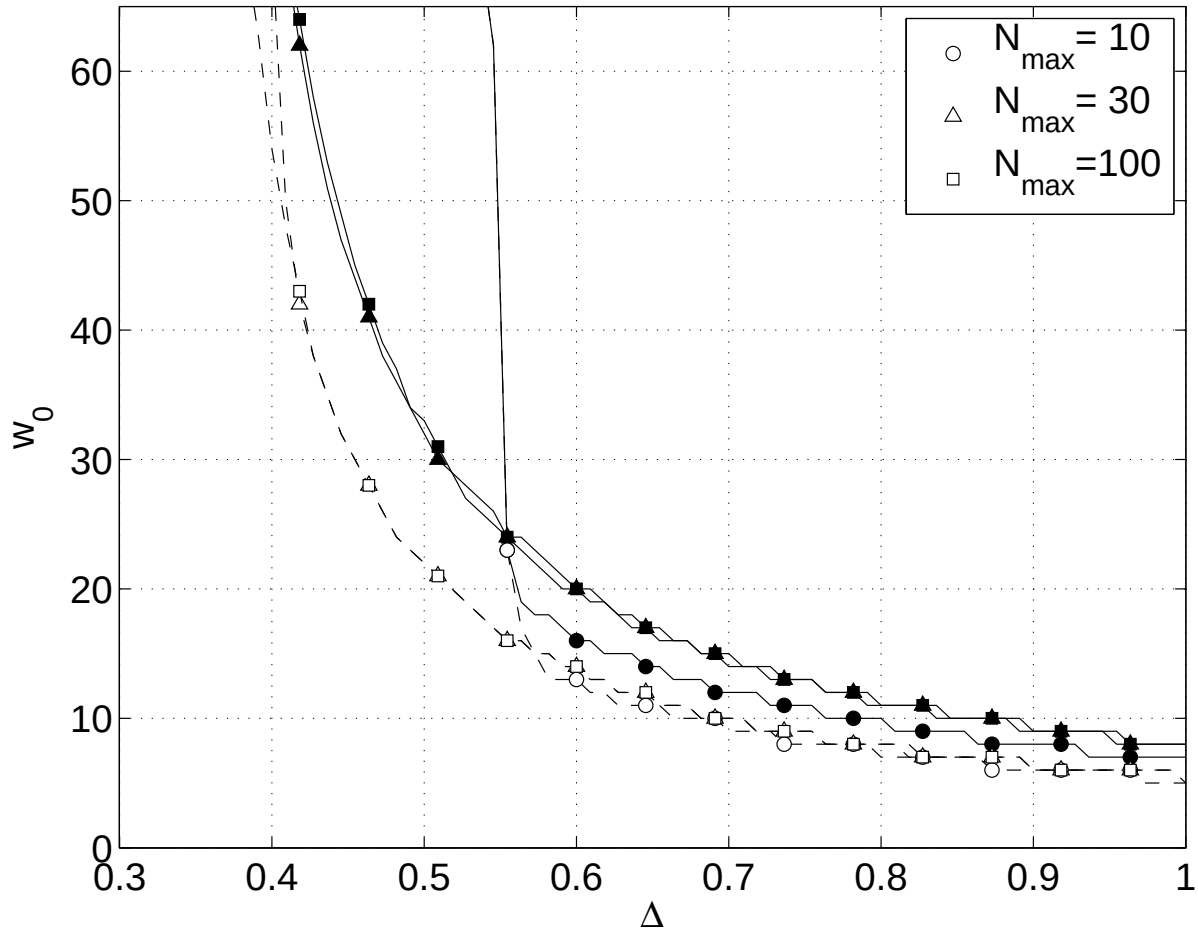

Figure 6.3: Estimate startup contention frame size w_0 *vs* error threshold Δ with $\beta = 0.0225$ (black markers) and $\beta = 0.0654$ (white markers).

where p is the contention probability used in the empty round. Note that, the smaller P_{thr}, the closer n_0 to zero, hence P_{thr} can be somehow interpreted as a measure of our belief that the batch is actually populated, even though the first round turned out empty. Finally, note that when using the uniform distribution for N, (6.16) gets the much simpler expression

$$n_0 = \left\lceil \frac{\log(1 - P_{thr}(1 - (1-p)^{N_{\max}}))}{\log(1-p)} \right\rceil . \tag{6.17}$$

The second critical case occurs when the estimate of the batch size given by (6.7) is out of the estimate operating range, an event that is likely to occur when $\mathrm{m}_N \ll n$. In this case, the start up phase is repeated by setting $\mathrm{m}_N = \hat{n}$, where $\hat{n}$ is the estimate obtained after the previous round. The procedure is repeated until the estimate lies within the operating range. At this point, the resolution process proceeds as usual, that is to say, with $p = 1$ and $w = w^{\star}_{\hat{n}}$.

Chapter 7

Case study

In this section, we compare ABRADE$^+$ against the best-performing BRAs algorithm in the literature. In order for the comparison to have practical significance, we set the system parameters according to the specifications of two popular off-the-shelf radio technologies, namely IEEE 802.11 and IEEE 802.15.4, that for ease of notation will be henceforth referred to as WF (WiFi) and ZB (ZigBee), respectively. The values of the system parameters for these two reference scenarios, whose derivation is detailed in Appendix, are collected Tab. 7.1. For ABRADE$^+$, we fixed $\Delta = 0.6$ and $P_{thr} = 0.25$.

7.1 Poisson-distributed batch size with known mean

We first compare ABRADE$^+$ against the FCFS algorithm that, as explained in Sec. 3, is the best known collision resolution algorithm when batches have Poisson-distributed multiplicity of known mean. For fair comparison, we hence assume that the batch size is Poisson-distributed with mean m_N known at the inquirer. This information is used to initialized the algorithms, which are then simulated by considering the system parameters in Tab. 7.1. The mean batch size m_N has been varied from 1 to 1500. For each value of m_N, we run 20000 independent instances of both the algorithms, generating at each simulation a new batch, with multiplicity drawn at random from the Poisson distribution of mean m_N. The 99% confidence intervals are tight-fitting the curves and, for clarity, have not been reported in the graph.

Fig. 7.1 shows the *mean* throughput (2.3) in WF (white marks) and ZB (black marks) scenarios. Throughput curves grows rapidly with m_N, to bend toward the asymptotic values when $\mathrm{m}_N \geq 150$. For very small batches, FCFS is slightly better than ABRADE$^+$, which suffers the longer duration of the probe message with respect to the immediate feedback messages. However, when m_N is larger than a few units, ABRADE$^+$ outperforms FCFS in both scenarios, though the performance gap is less marked in the ZB

Table 7.1: System parameters setting in WF and ZB scenarios ($\beta_p = h_0 + b_p w$).

	T_{data} [μs]	β	φ_i	φ_s	φ_c	h_0	b_p
				[10^{-3}]			
WF	399	22.5	0	131.9	131.9	143.2	0.05
ZB	4896	65.4	0	111.1	45.8	248.4	0.82

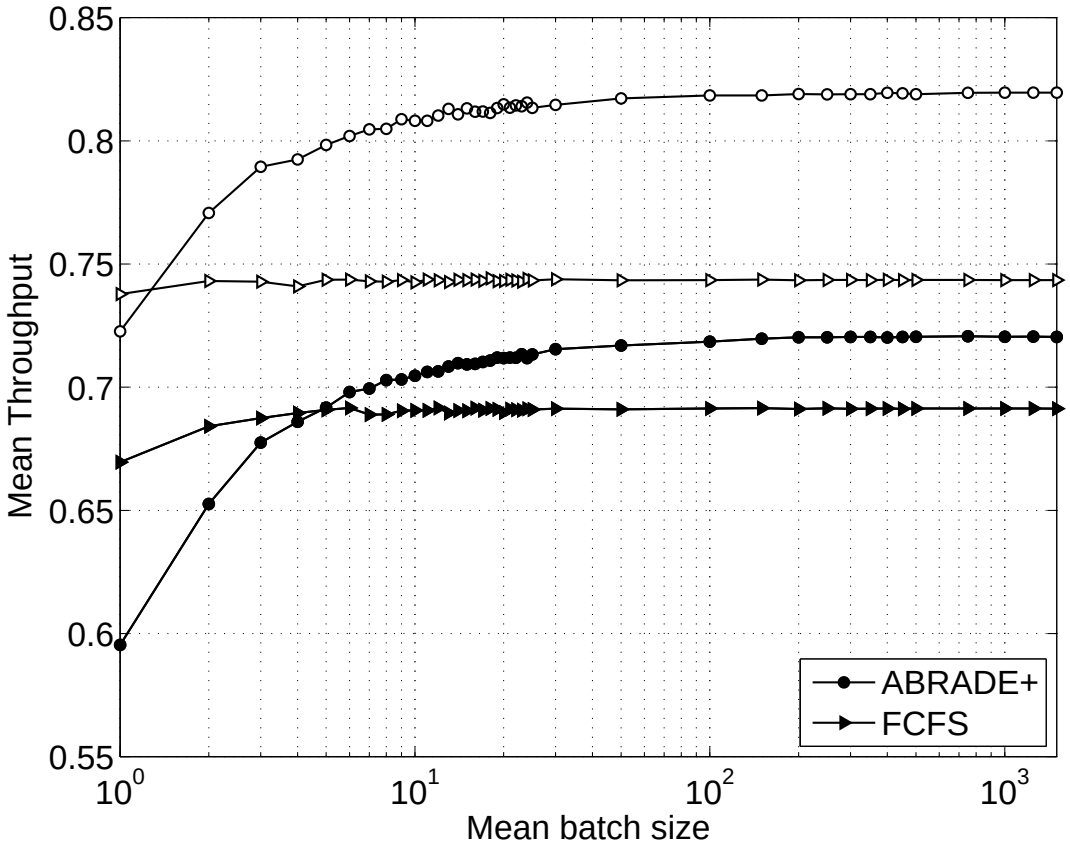

Figure 7.1: Mean throughput $\bar{\lambda}(\mathrm{m}_N)$ of ABRADE$^+$ and FCFS as a function of the mean batch size m_N, in WF (white marks) and ZB (black marks) scenarios.

scenario, whose feedback costs are smaller than for WF.

Tab. 7.2 collects the asymptotic throughput of ABRADE$^+$ and FCFS obtained by simulation and theoretical analysis.[1] The right-most column reports $\lambda_{\max}$ for FCFS obtained under the classical assumption (3.1), as derived in [8]. We can see that the matching between simulation and theoretical values for ABRADE$^+$ is very good, thus confirming the validity of the mathematical model and the marginal impact of the simplifying assumptions considered in the analysis of Sec. 5. Similarly, the simulation results for FCFS, obtained with realistic parameters values, are pretty close to the values obtained with the model (3.2) that keeps into account feedback costs. Conversely, the theoretical model provided in [8], which neglects feedback, overestimates the asymptotic throughput of 10% and 5% in the WF and ZB scenario, respectively.

7.2 Unknown batch size

Here we consider the case where the inquirer has no knowledge about the batch multiplicity distribution. Hence, the parameters w_0 and p in ABRADE$^+$ are set as described in Sec. 6.2, under the (arbitrary) assumption that N is uniformly distributed in the (integer) interval $[0, N_{\max}]$. All the results presented in this section have been obtained by setting $N_{\max} = 100$. We remark that this choice is arbitrary and independent of the actual size n of the batch, which is now deterministic, but unknown to the inquirer.

We compare ABRADE$^+$ with the other BRAs that perform a dynamic batch size estimate, namely IECR, Sift and Sift/IECR. In Fig. 7.2 and Fig. 7.3 we report the algorithms throughput $\lambda(n)$ for different values of n, in the WF and ZB scenarios, respectively.[2] Once again, the 99% confidence intervals are not reported being tight-fitting the curves. The results obtained in the two scenarios are qualitatively the same, though the throughput gaps in the ZB case are less marked than for the WF scenario, as already observed in Sec. 7.1.

At first sight, we see that ABRADE$^+$ outperforms all the other algorithms for batches greater than a few units. ABRADE$^+$ throughput grows almost monotonically with the batch size, except in a interval of values after $n = 100$, where the throughput undergoes a contraction. This small performance flexion originates from the setting $N_{\max} = 100$: when the actual batch size n is larger than $N_{\max}$, the first resolution

[1] The theoretical asymptotic throughput of ABRADE$^+$ is still given by (5.9), since the effect of the initial batch size uncertainty becomes negligible as $n \to \infty$.

[2] Note that, from these results it is possible to obtain the average throughput for an arbitrary distribution $P_N(\cdot)$ of N by using (2.3).

rounds will likely experience more collisions than expected, determining a performance loss. However, when $n \gg N_{\max}$, ABRADE$^+$ has enough time to reach the optimal regime behavior, thus progressively compensating the initial performance loss. Hence, the value of $N_{\max}$ determines the batch size interval that experiences the performance loss.

Concerning the other algorithms, we observe that Sift throughput progressively increases as n approaches the design value $N_{\max} = 512$, to smoothly decrease for larger batches. In any case, Sift performs generally worse than the other algorithms, being designed for a different purpose. When the batch systematically contains a single node, IECR algorithm outperforms all the others, being able to solve the batch in just one slot when the other algorithms generally take in some idle slots. However, as soon as the number of nodes in the batch is greater than one, IECR will experience collisions in the very first slots, which determine the sharp throughput loss that can be observed in the range $1 < n \leq 5$. For larger values of the batch size, the throughput of IECR progressively improves, because the algorithm is capable of dynamically adjusting its parameters to the residual batch size before the batch resolution process is completed, thus partially recovering from the initial inefficiency due to suboptimal parameters configuration. Finally, as expected, Sift/IECR performs better than IECR for small values of the batch size, whereas for large batches the throughput achieved by the two algorithms is practically the same.

Table 7.2: Asymptotic throughput of ABRADE$^+$ and FCFS in WF and ZB scenarios: simulation (at $N = 1500$) vs theory.

	ABRADE$^+$		FCFS		
	Sim.	Theory (5.9)	Sim.	Theory (3.2)	[8]
WF	0.81775	0.8202	0.7409	0.7494	0.8249
ZB	0.71924	0.7229	0.6901	0.7021	0.7345

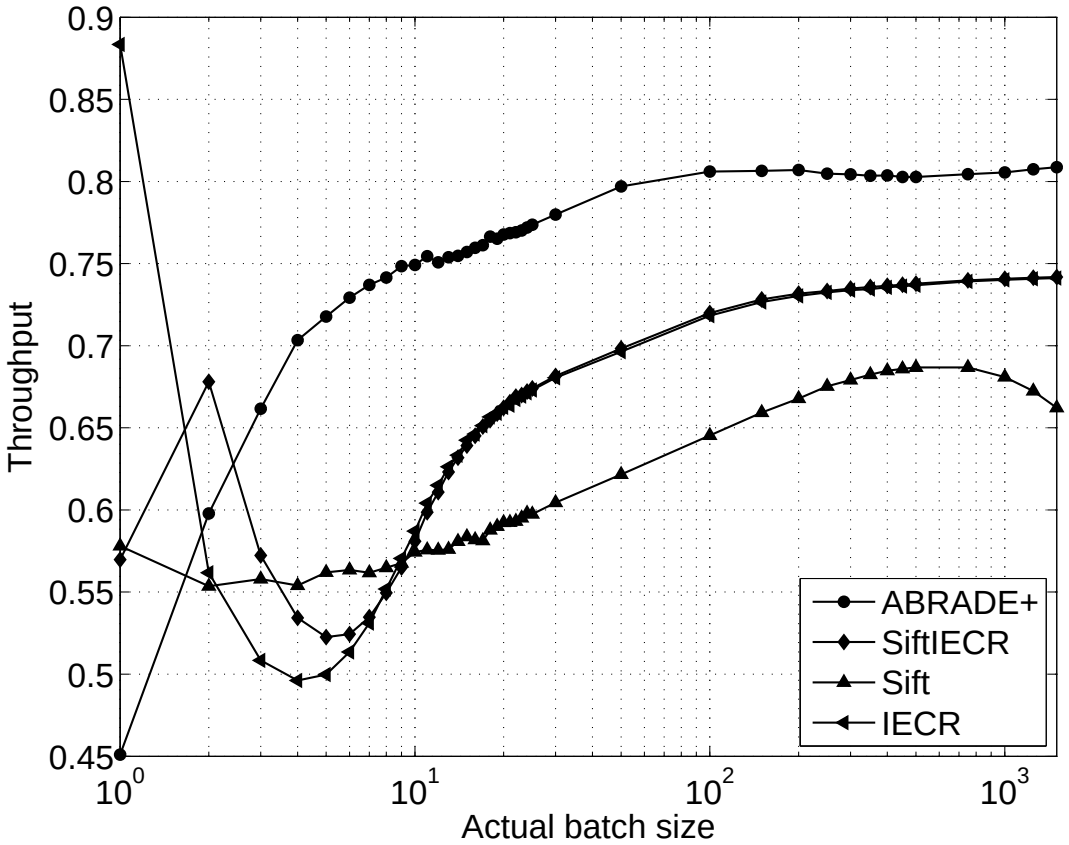

Figure 7.2: Throughput $\lambda(n)$ of ABRADE$^+$, Sift/IECR, Sift and IECR as a function of the batch size n, in WF scenario.

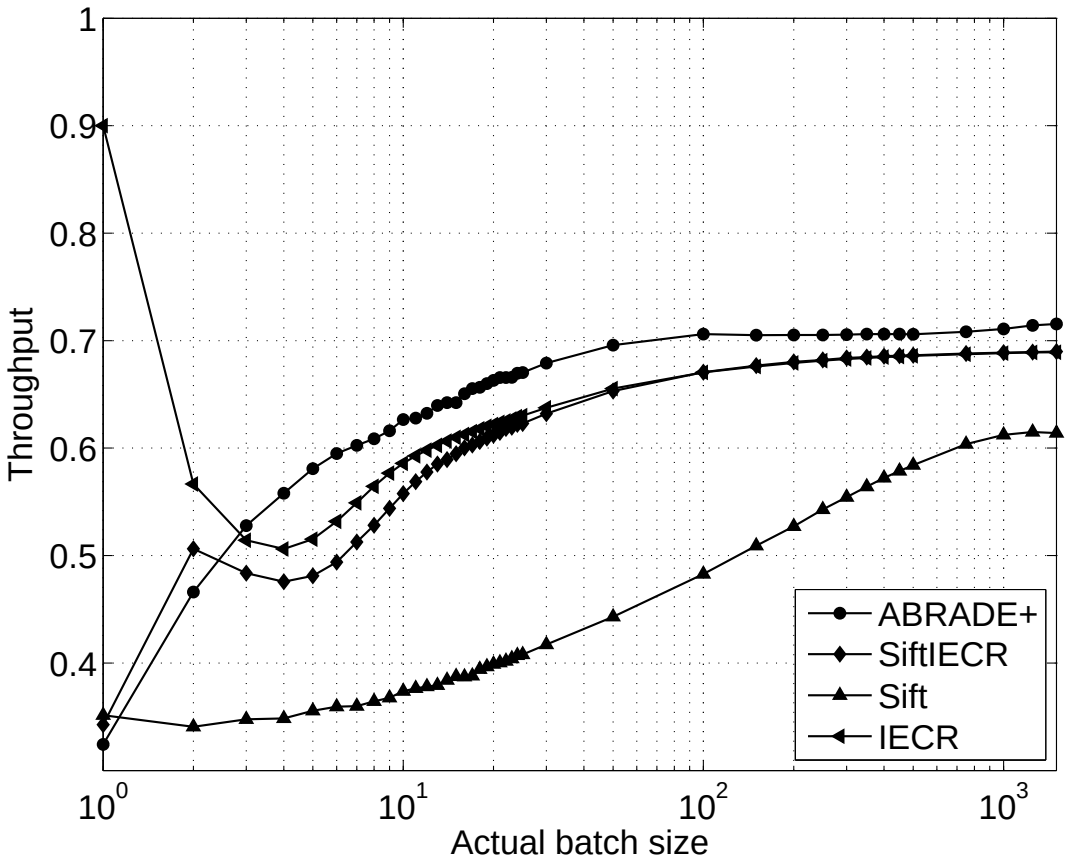

Figure 7.3: Throughput $\lambda(n)$ of ABRADE$^+$, Sift/IECR, Sift and IECR as a function of the batch size n, in ZB scenario.

Chapter 8

Discussion

In this report we presented ABRADE$^+$, a batch resolution algorithm resulting from the combination of a batch size estimation module and a dynamic framed ALOHA scheme. Conversely to most conflict resolution algorithms presented in the literature, ABRADE$^+$ waives the immediate feedback paradigm in favor of an aggregate feedback, in order to reduce the overhead. Furthermore, ABRADE$^+$ dynamically adjusts the length of the contention frame to the estimated multiplicity of the residual batch, in order to minimize the overall batch resolution interval.

ABRADE$^+$ performance has been analyzed, both mathematically and by simulations, and compared against the best performing algorithms in the literature based on the immediate feedback paradigm. The analysis revealed that, by setting the system parameters in accordance with the specifications of common radio standards, such as IEEE 802.11 (WF) and IEEE 802.15.4 (ZB), the throughput achieved by classical BRAs' is actually lower than predicted by the theoretical analysis, which generally neglects or underestimates the cost of immediate feedback. The deferred feedback approach adopted by ABRADE$^+$, thus, appears more suitable in these scenarios, yielding higher throughput, especially with large batches. The performance gain is more relevant when the feedback costs are comparable to the packet transmission time, as occurs in WF and, in general, in high speed wireless LAN, whereas the advantage of the deferred feedback approach is more contained in slow speed systems, such as ZB.

ABRADE$^+$ may seem to be more complex than other solutions, such as FCFS or IECR. As a matter of fact, the frame-length optimization scheme, which is by far the more complex operation in the algorithm, may potentially prevent the on-line applicability of ABRADE$^+$. However, we observe that the complexity is all concentrated into the inquirer, whereas the other nodes have only to perform the same basic operations as for the classical algorithms. Furthermore, knowing the specification of the wireless technology, the optimal parameters can be computed offline and stored in a table in the inquirer. Alternatively, it is possible

to drastically simplify the dynamic frame length adaptation by approximating $w_n^\star$ to the asymptotic value $\lceil \mu_\infty n \rceil$. In this way, the frame length adaptation can be performed by a simple rounded multiplication, at the cost of a marginal performance penalty for small batches.

We finally observe that ABRADE$^+$ may be further ameliorated and extended in different ways. For instance, the contention frame may be slid forward in case of runs of idle slots, in order to avoid almost-certain collisions in the last part of the frame. In quasi static scenarios, it is possible to design the estimate module in order to consider the past events, for instance by using a Bayesian approach, thus speeding up the convergence of the estimate to the actual batch size. Increasing the complexity of the algorithm at the nodes side, furthermore, it is also possible to conceive extensions of the approach to more complex scenarios, involving hidden nodes, multi-hop communications, or priority-based service differentiation. Such extensions are left for future investigation.

APPENDIX

Characterization of system parameters in the reference scenarios

The data structures and carrier-sense mechanism of WF and ZB are similar and will be described in a unified manner. Let t_{BCK} denote the reference idle (backoff) slot time. Furthermore, let T_{PCK} be the transmission time of a data packet, including PHY and MAC headers and payload. In case of unacknowledged transmissions, successive data frames have to be separated by (at least) a certain Inter Frame Space (IFS), here denoted by t_{IFS}. In case of acknowledged transmission, instead, an ACK frame of duration T_{ACK} shall be returned within a time interval $t_{ACK} < t_{IFS}$ after the packet reception. Moreover, an IFS shall follow any ACK frame transmission. If a valid ACK is not received within a certain timeout t_{TO} the transmission is assumed to be collided and a new data frame can be transmitted immediately after.

With reference to the system model defined in Sec. 2, the system parameters can thus be set as follows

$$\begin{aligned} &T_{data} = T_{PCK} + t_{IFS}\,; \quad \beta = \frac{t_{BCK}}{T_{data}}\,; \\ &\varphi_i = 0\,; \quad \varphi_s = \frac{T_{ACK} + t_{ACK}}{T_{data}}\,; \quad \varphi_c = \frac{t_{TO} - t_{IFS}}{T_{data}}\,. \end{aligned} \tag{A-1}$$

The probe message can be realized by using a standard data frame, whose payload is structured in order to carry the required system parameters. We suppose that a fixed number of bits, that we set to 24 bits, are used to encode ABRADE$^+$ parameters, including the contention frame size w and the contention probability p to be used in the upcoming resolution round. Note, that p is always equal to one, except at the estimate startup phase during which, though, w is generally small. Therefore, at regime, most of these bits can be used to encode the value of w. The encoding of the aggregate feedback field may consist of a bit-mask in one-to-one correspondence with the slots in the contention frame, in such a way that bits 1 denote successful slots, whereas bits 0 indicate collided and idle slots. Therefore, the number of bits in this field is equal to the length of the previous contention frame. However, more efficient codings of the aggregate ACK field are possible. For instance, exploiting the sensing capability of nodes, the aggregate ACK field may be compacted by using a bit-map for busy slots only, where bits 1 denote success and 0

collision.

In our simulations, we conservatively set the length of the probe message to

$$\beta_p = h_0 + b_p w \,,$$

where h_0 accounts for the fixed duration of the probe message (PHY and MAC packet headers, fixed payload subfields, IFS), whereas $b_p = \frac{1}{RT_{data}}$ is the normalized bit transmission time, being R the transmission rate of the payload field. For large frame lengths, the length of the probe message may exceed the maximum frame length supported by the transmission standard. In this case it will be needed to send two or more consecutive probe messages.

Bibliography

[1] D. R. Hush and C. Wood, in *In IEEE International Symposium on Information Theory*, p. 107.

[2] J. Hayes, "An adaptive technique for local distribution," *Communications, IEEE Transactions on*, vol. 26, no. 8, pp. 1178–1186, Aug 1978.

[3] J. Capetanakis, "Tree algorithms for packet broadcast channels," *Information Theory, IEEE Transactions on*, vol. 25, no. 5, pp. 505–515, Sep 1979.

[4] P. Popovski, "Tree protocols for RFID tags with generalized arbitration spaces," in *IEEE 10th International Symposium on Spread Spectrum Techniques and Applications, 2008.*, Aug. 2008, pp. 18–22.

[5] R. Gallager, "Conflict resolution in random access broadcast networks," in *AFOSR Workshop Commun. Theory Appl., Provincetown, MA*, September 1978, pp. 74–76.

[6] J. Mosely and P. Humblet, "A class of efficient contention resolution algorithms for multiple access channels," *Communications, IEEE Transactions on*, vol. 33, no. 2, pp. 145–151, Feb 1985.

[7] M. Molle and G. Polyzos, "Conflict resolution algorithms and their performance analysis," Department of Computer Science and Engineering, UCSD, East Lansing, Michigan, Tech. Rep. CS93-300, July 1993.

[8] D. Bertsekas and R. Gallager, *Data networks (2nd ed.)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1992.

[9] S. Tasaka, "Stability and performance of the R-Aloha packet broadcast system," *IEEE Trans. Comput.*, vol. C, no. 32, pp. 717–726, Aug. 1983.

[10] K. Jamieson, B. Hull, A. K. Miu, and H. Balakrishnan, "Understanding the Real-World Performance of Carrier Sense," in *ACM SIGCOMM Workshop on Experimental Approaches to Wireless Network Design and Analysis (E-WIND)*, Philadelphia, PA, August 2005.

[11] P. Mathys and P. Flajolet, "Q -ary collision resolution algorithms in random-access systems with free or blocked channel access," *Information Theory, IEEE Transactions on*, vol. 31, no. 2, pp. 217–243, Mar 1985.

[12] I. Cidon and M. Sidi, "Conflict multiplicity estimation and batch resolution algorithms," *Information Theory, IEEE Transactions on*, vol. 34, no. 1, pp. 101–110, Jan 1988.

[13] P. Popovski, F. H. P. Fitzek, and R. Prasad, "A class of algorithms for collision resolution with multiplicity estimation," *Algorithmica*, vol. 49, no. 4, pp. 286–317, 2007.

[14] Y. Tay, K. Jamieson, and H. Balakrishnan, "Collision-minimizing CSMA and its applications to wireless sensor networks," *Selected Areas in Communications, IEEE Journal on*, vol. 22, no. 6, pp. 1048–1057, Aug. 2004.

[15] G. Pierobon, A. Zanella, and A. Salloum, "Contention-TDMA protocol: Performance evaluation," *IEEE Transactions on Vehicular Technology*, vol. 51, pp. 781–788, 2002.

[16] F. C. Schoute, "Dynamic frame length ALOHA," *IEEE Trans. on Communications*, vol. COM-31, no. 4, pp. 565–568, April 1983.

[17] P. Popovski, F. H. P. Fitzek, and R. Prasad, "Batch conflict resolution algorithm with progressively accurate multiplicity estimation," in *DIALM-POMC '04: Proceedings of the 2004 joint workshop on Foundations of mobile computing*. New York, NY, USA: ACM, 2004, pp. 31–40.

[18] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in RFID systems," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2006, pp. 322–333.

[19] Q. Tong, X. Zou, and H. Tong, "Dynamic framed slotted aloha algorithm based on Bayesian estimation in RFID system," in *Computer Science and Information Engineering, 2009 WRI World Congress on*, vol. 1, march 2009, pp. 384 –388.