Ph.D. Thesis

Traffic modeling and performance management in data networks

Nicola Aste

Advisor: Prof. Luigi Atzori

University of Cagliari
Department of Electrical and Electronic Engineering

Index

Chapter 3

Routing Multiple Multicast Services using Genetic Algorithms……87

Introduction

Networks continue to grow more complex as industry deploys a mix of wired and wireless technologies into large scale heterogeneous network architectures and as user applications and traffic continue to evolve. For example, increased complexity already affects Department of Defense combat networks, the Internet, and industrial wireless networks. For these reasons, network designers and researchers need to adopt new strategies in order to assure a sufficient *Quality of Service* (QoS) for clients. This is a delicate task both for networks already operative or still in planning phase. Sometimes, when allowed by the system simplicity, the stochastic behaviors of telecommunication networks can be completely treated with analytical statistical means by building a complete model both for system elements and carried traffic. Modeling of network traffic is also important to solve several traffic engineering and management problems, such as routine traffic monitoring, wavelength assignments, route optimization, admission control policies for new data services, traffic policing and shaping, and peering policies [XTF+02]. Together with network nodes and links models, analytical traffic models used to depict dynamic and behavioral load

characteristics, help to capture relevant operational features such as topology, bandwidth, buffer space, and nodal service policies (link scheduling, packet prioritization, buffer management, etc.) [RL00]. Usually, the model to be used is strictly related to the type of traffic management function. The high complexity of an accurate traffic modeling is useless for some operations, such as route optimization and traffic policing while becomes necessary for other functions, such as queuing rate adjustment and scheduling operations, which can be considered as micro control in traffic management [ENN+00], [XTF+02]. Even in the less demanding case of off-line processing, the ever increasing volume of data that can be collected over a given time interval poses huge storage and processing problems. Such limitations are particularly serious if parameters crucial to meaningful traffic characterization have high computational complexity. In the last few years the discovery of the multifractal nature of many kinds of packet traffic has inspired a small revolution in the way that high speed traffic is viewed [PF95], [BST+95], [GW94], [FGW+98], [FGW98]. Although no single model is accepted as definitive, the estimation of multiscaling parameters, which describes the scaling behavior of the load process at different temporal scales and order moments, represents the best analytical tool for the provision of QoS as well as for networks dimensioning. Unfortunately, methods for the estimation of such parameters from data have suffered from poor statistical performance and/or high computational complexity inappropriate for large data sets or real time use. Recent work based on wavelets, however, has provided a semi-parametric estimator for which gives unbiased estimates together with significant computational advantages [AFT+00], allowing real time load process parameterization and bandwidth allocation for incoming traffic. In fact, when accessible, real time modeling represents the best efficient solution since can be used to guide policing operations and then to dynamically choose the best control action required to optimize the QoS.

Computer networks have made revolutionary changes in our way of living since the day they were born. With the advance of technology, computer networks evolve from the original ARPANET to today’s Internet, and the next generation of data networks of faster speed and more convenience is already looming at horizon. The advance has pushed computer networks on both aspects of bandwidth and ease in use. For bandwidth, optical fibers are replacing copper wires to transmit tera bits in a second. With such a dramatic increase in bandwidth at transmission media, new technologies are needed for switches, routers and network management equipments to cope with such changes. On the other hand, to make computer networks easier to reach people no matter where they are, wireless data networks have become one of the hottest topics in both today’s industry and research institutes. With the proliferation of portable computing platforms and small wireless devices, wireless networks have received more and more attention as a means of data

communication among devices regardless of their physical locations. IEEE 802.11 [IEEE97] is a standard for wireless system that operates in the 2.4 – 2.5 GHz ISM (industry, scientific and medical) band, which is available worldwide and allows unlicensed operation for spread spectrum systems. The main attraction of 802.11 wireless LAN is its flexibility and mobility. The wireless LAN can extend access to local area networks, such as corporate intranets, as well as support broadband access to the Internet. 802.11 wireless LAN can also provide quick, easy wireless connectivity to computers, machinery, or systems in a local environment where a fixed communications infrastructure does not exist or where such access is not permitted. However, the popularity of 802.11 LAN requires the vendors to provide higher quality of service for the user. Nearly all mobile computers are battery-powered. Thus, the life of battery power available time is a crucial character of the battery powered mobile device. The wireless network access is a fundamental enabling feature for many portable computers, but if not optimized for power consumption this feature can quickly drain a device's limited batteries. According to the pervious measurement, the overall energy consumption by wireless network interface can represent over 50% of total system power of current handheld computer and up to 10% for high-end laptops. Power is arguably the scarcest resource for mobile devices, and power saving has always been a major design issue for the developers of mobile devices, wireless communication protocols and mobile computing systems. Since users will not accept recharging nor changing batteries very other hour, neither will they carry a heavy battery pack, power puts many limitations on operations of a mobile device. Computing ability is sacrificed because high performance processor needs more power. Transmission range and bandwidth are restricted due to the fact that long range high bandwidth transceivers consume too much energy. The IEEE 802.11 standard specifies a Power Management (PM) mechanism that allows a mobile station to enter in a state of low power consumption when its interface is idle. However, since power represents the main resource within the wireless networks, improving the PM mechanism implemented in the IEEE 802.11 protocol, could bring to the achievement of higher results in terms of performance.

Thesis outline

This thesis is mainly aimed to report in a unified document my three years research about network traffic modeling and performance management. In particular three main projects will be accurately described indicating mathematical theory, experiments set up and consequent final results.

The rest of the thesis is organized as follows: Chapter 1 describes and analyses the multiscale nature of the network traffic. We then propose a segmentation method for estimating in real time the

parameters characterizing the multifractal representation that has been recently indicated by literature as the best traffic model. Such parameters are also fundamental for predicting the effective bandwidth that has to be allocated by the router for incoming streams. Since load processes could have heterogeneous scaling behavior for long time intervals, causing estimations to be meaningless, the method proposes an adaptive strategy able to dynamically adjust the estimation interval length on the basis of the local traffic features in order to assure good CIs. Finally, results will show that adopting the segmentation technique it is possible to allocate a lower effective bandwidth then normal strategies.

In Chapter 2, we describe in details the IEEE 802.11 [IEEE97] standard for the wireless LANs; we then propose three novel algorithms for Power Management infrastructure WLANs: all these are based on giving the Access Point the power of deciding when notifying stations in doze with pending frames so as to avoid waking stations in case of high traffic. According to the first one, the decision is taken on the basis of the probability to find lower traffic in future beacon intervals with a constraint on the transmission delay. The second and the third ones are based on minimizing a cost function that weighs both energy consumption and introduced latency. Simulations have shown that the proposed method provides an overall energy saving of about 75% at the expense of an increase in the latency of about 0.3 sec.

Chapter 3 focuses on the group multicast routing problem with a twofold contribution. Firstly, the use of the genetic algorithms is proposed for solving the complexity problem inherent to the packing of multiple sessions. Secondly, a novel cost function is proposed, weighting in a single expression both network bandwidth allocation and provided one-way delay. The proposed function is guided by few parameters that can be easily tuned during traffic engineering operations; an appropriate setting of these parameters allows the operator to configure the desired balance between network resource utilization and provided QoS (in terms of transmission delay). Experimental results are compared with those of a heuristic algorithm that provides a lower bound for the optimization problem. This highlights that the proposed method allows a strong reduction of the processing time.

Chapter 1

Multiscale nature of Network Traffic

1.1 Traffic and Scaling

The complexity and richness of telecommunications traffic is such that one may despair to find any regularity or explanatory principles. Nonetheless, the discovery of scaling behavior in teletraffic has provided hope that parsimonious models can be found. The statistics of scaling behavior present many challenges, especially in nonstationary environments.

By the term telecommunications traffic or teletraffic we mean the flow of information, or data, in telecommunications networks of all kinds. From its origins as an analog signal carrying encoded voice over a dedicated wire or "circuit," traffic now covers information of all kinds, including voice, video, text, telemetry, and real-time versions of each, including distributed gaming. Instead of the dedicated circuits of traditional telephone networks, packet switching technology is now used to carry traffic of

all types in a uniform format (to a first approximation): as a stream of packets, each containing a header with networking information and a payload of bytes of "data." Although created by man and machine, the complexity of teletraffic is such that in many ways it requires treatment as a natural phenomenon. It can be likened to a turbulent, pulsating river flowing along a highly convoluted landscape, but where streams may flow in all directions in defiance of gravity. The landscape is the network. It consists of a deep hierarchy of systems with complexity at many levels. Of these, the "geographical" complexity or connectivity of network links and nodes is of central importance. Other key aspects include the size or bandwidth of links (the volume of the river beds), and at the lowest level, a wide variety of physical transport mechanisms (copper, optic fiber, etc.) exist with their own reliability and connectivity characteristics.

Although each atomic component is well understood, the whole is so complex that it must be measured and its emergent properties "discovered." Comprehensive simulation is difficult. A key concept in networking is the existence of network protocols, and their encapsulation. Let us explain with an example. The Internet protocol (IP) is used to allow the transport of packets over heterogeneous networks. The protocol understands and knows how to process information such as addressing details contained in the header of IP packets. However, by itself IP is only a forwarding mechanism without any guarantee of successful delivery. At the next higher level, the transfer control protocol (TCP) provides such a guarantee by establishing a virtual connection between two end points and monitoring the safe arrival of IP packets, and managing the retransmission of any lost packets. On a still higher level, web-page transfers occur via the hypertext transport protocol (HTTP), which uses TCP for reliable transfer. The resulting encapsulation "HTTP over TCP over IP" therefore means that HTTP oversees the transfer of text and images, while the actual data files are handed over to TCP for reliable transfer. TCP chops the data into datagrams (packets) which are handed to IP for proper routing through the network. This organization offers hierarchal structuring of network functionality and traffic but also adds complexity: each level has its own dynamics and mechanisms, as well as time scales.

Over this landscape flows the teletraffic, which has even more levels of complexity than the underlying network.

Three general categories can be distinguished.

Geographic complexity plays a major role. Although one can think of the Internet as consisting of a "core" of very high bandwidth links and very fast switches, with traffic sources at the network "edge," the distances from the edge to the core vary greatly, and the topology is highly convoluted. Access bandwidths vary widely, from slow modems to gigabit Ethernet local area networks, and mobile access

creates traffic which changes its spatial characteristics. Sources are inhomogeneously distributed; for example concentrations are found in locations such as universities and major corporations. Furthermore traffic streams are split and recombined in switches in possibly very heterogeneous ways, and what is at one level a superposition of sources can be seen at another level, closer to the core, as a single, more complex kind of "source."

Offered traffic complexity relates to the multilayered nature of traffic demands. Users, generating web browsing sessions for example, come and go in random patterns and remain for widely varying periods of time, during which their activity levels (number of pages downloaded) may vary both qualitatively and quantitatively. The users' applications will themselves employ a variety of protocols that generate different traffic patterns, and finally, the underlying objects themselves, text, audio, images, video, have widely differing properties.

Temporal complexity is omnipresent. All of the above aspects of traffic are time varying and take place over a very wide range of time-scales, from microseconds for protocols acting on packets at the local area network level, through daily and weekly cycles, up to the evolution of the phenomena themselves over months and years. The huge range of time-scales in traffic and the equally impressive range of bandwidths, from a kilobytes up to terabytes per second over large optical backbone links, offers enormous scope for scale dependent behavior in traffic.

But is this scope actually "exploited" in real traffic? Is traffic in fact regular on most time scales, with variability easily reducible to, say, a diurnal cycle plus some added variance arising from the nature of the most popular data-type/protocol combination? Since the early 1990s, when detailed measurements of packet traffic were made and seriously analyzed for the first time [LTW+94], we know that the answer is an emphatic "No." Far from being smooth and dominated by a single identifiable factor, packet traffic exhibits scale invariance features, with no clear dominant component. For instance, long memory is a scale invariance phenomenon that can be seen in the time series $Y(t)$ describing the data transfer rate over a link at time t. Other examples of time series with long memory are the number of active TCP connections in successive time intervals or the successive interarrival times of IP packets.

The philosophy of scale invariance or "scaling" can be expressed as the lack of any special characteristic time or space scale describing fluctuations in $Y(t)$. Instead one needs to describe the steady progression across scales. In the case of traffic such a progression has been found empirically and has lead to long memory models and more generally to models with fractal features, as we will explore. The scale invariant features of traffic can be thought of as giving precise meaning to the important but potentially vague notion of traffic burstiness, which means, roughly, a lack of

smoothness. In very general terms, burstiness is important because from the field of performance analysis of networks, and in particular that of switches via queueing theory, we know that increased burstiness results in lower levels of resource utilization for a fixed quality of service, and therefore to higher costs. At the engineering level, service quality refers to metrics such as available bandwidth, data transfer delay, and packet loss. The impact of scale invariance extends to network management issues such as call admission control, congestion control, as well as policies for fairness and pricing [XTF+02]. It is important to distinguish between two canonical meanings of the term burstiness, which have their counterparts in models and analysis. Again let us take "traffic" to be the data rate *Y(t)*, nominally in bytes per second, over a link at time *t*. One kind of burstiness arises from dependencies over long time periods, which can be made precise in terms of the correlation function of *Y(t)* (assuming stationarity and that second order statistics exist). Such temporal burstiness was explored when scaling was first found in packet traffic. More precisely, the well-known long-range dependent (LRD) property of traffic is a phenomenon defined in terms of temporal correlation, whose network origins are now thought to be quite well understood in terms of the paradigm of heavy tails of file sizes of requested objects, which causes sources to transmit over extended periods. A second kind of burstiness describes variability, the size of fluctuations in value or amplitude, and therefore concerns small scales. It refers to the marginal distribution of *Y(t)*, as characterized, for example, by the ratio of standard deviation to mean if this exists, as the local singular behavior of multifractal models, or alternatively as a heavy tail parameter of the distribution of the instantaneous traffic load in the case of infinite variance models.. Even when an apparently stationary subset is selected, the variation in value or amplitude is very significant and highly non-Gaussian. Marginals of other time series do not always yield such extreme power-law tails; however Weibullian or log-normal behavior is more common than Gaussian, unless the data has already been highly aggregated or if scales above a few seconds are examined. The two types of burstiness just described are quite different. However, often it is convenient to work not with a stationary series like *Y(t)*, but with its integrated or "counting process" equivalent *N(t)*, which counts the amount of traffic arriving in $[0, t]$. It is then important to bear in mind that the statistics of *N(t)* are a function both of the temporal and the amplitude burstiness of the rate process *Y(t)*.

Although at large scales (seconds and beyond) astonishingly clear, simple, and relatively well-understood scaling laws are found, the same cannot be said at small scales. This is true, for example, of a discrete series giving the successive intervals (in milliseconds) between the arrival of new TCP connections [ABF+02]. When examined with the naked eye this series may be accused of having long

memory, with a marginal slightly deviating from Gaussianity. In reality, in addition to long memory, it contains much nontrivial scaling structure at small scales which is suggestive of a rich underlying dynamics of TCP connection creation. Knowledge of its scaling properties lays a foundation for an informed investigation. The fact is that much work remains to be done to achieve a clear understanding of traffic scaling over small scales, which is characterized by far higher variability, more complex and less definitive scaling laws, and the necessity of dealing with non-Gaussian data and hence statistics beyond second order. While large-scale behaviour such as long memory matters for many network design and management issues, understanding small-scale behavior is particularly important for flow control, performance, and efficiency. In terms of network performance, variability is (almost) always an undesirable feature of traffic data. Therefore, a key motivation for investigating such scaling is to help identify generating mechanisms leading to an understanding of their root causes in networking terms. If, for example, it were known that a certain feature of the TCP protocol was responsible for generating the observed complex scaling behavior at small scales, then we would be in a position to perhaps eliminate or moderate it via modifications to the protocol. Alternatively, if a property of certain traffic source types was the culprit, then we could predict if the scaling would persist in the future or fade away as the nature of telecommunications services evolve. To conclude this introduction to scaling in telecommunications, we point out that in many series derived from traffic data, in particular TCP/IP traffic a recurring feature is the existence of a characteristic scale at around 1 sec, which separates the now classic “monoscaling” at large scales indicative of long memory, from the more complex, but none-the-less scaling behavior, at small scales. Multifractal models are one possible approach for the latter domain, that offer the possibility of integrating both regimes in a single description.

2.1 Scaling Models

The notion of scaling is defined loosely, as a negative property of a time series: the absence of characteristic scales. Its main consequence is that the whole and its parts cannot be statistically distinguished from each other. The absence of such scales means that new signal processing tools are needed both for analysis and modeling, while standard techniques built on characteristic times (for example, simple Markov models) must be abandoned. This section provides an introductory review of various models used to give flesh to the spirit of scaling.

2.1.1 Self-Similarity

The purest formal framework for scaling is undoubtedly that of exactly self-similar processes. A process $X = \{X_t, t \in R\}$ is said to be self-similar, with self-similarity parameter (also known as Hurst parameter) $H > 0$, if

$$X = \{X_t, t \in R\} \overset{d}{=} \{c^H X_{t/c}, t \in R\}, \ \forall c > 0 \tag{1.1}$$

where $\overset{d}{=}$ means equality for all finite dimensional distributions. A major consequence of this definition is that the moments of X, provided they exist, behave as power laws of time:

$$E\left|X_t\right|^q = E\left|X_1\right|^q \cdot |t|^{qH} \tag{2.1}$$

For applications, one usually restricts the class of self-similar processes to that of self-similar processes with stationary increments (or H-sssi processes). A process X is said to have stationary increments $Y_\delta(t)$ if $Y_\delta(t) = X_{t+\delta} - X_t \overset{d}{=} X_\delta - X_0, \ \forall \delta$. (3.1)

For a H-sssi process X, the self-similarity parameter necessarily falls in $0<H<1$ and the covariance function, when it exists, takes a specific, unique, and constrained form:

$$E[X_t X_s] = \frac{\sigma^2}{2}\left(|t|^{2H} + |s|^{2H} - |t-s|^{2H}\right), \ \sigma^2 = E\left|X_1\right|^2$$

Moreover, it can be shown that the autocovariance function of the increment process $Y_\delta(t)$ reads:

$$E[Y_\delta(t)Y_\delta(t+s)] = \frac{\sigma^2}{2}\ |s+\delta|^{2H} + |s-\delta|^{2H} - 2|s|^{2H} \tag{4.1}$$

The self-similarity of the process X is transferred to its increments insofar as:

$$Y_\delta(t) \overset{d}{=} c^H Y_{\delta/c}(t/c) \tag{5.1}$$

$$E\ |Y_\delta(t)|^2\ = E\ |X_{\delta+t} - X_t|^2\ = \sigma^2|\delta|^{2H} \tag{6.1}$$

Self-similarity implies that an affine dilated subset of one sample path cannot be distinguished from its whole. It is therefore not possible to identify a reference scale of time, and thus there is no such reference scale. Exact statistical self-similarity thereby fulfils the intuition of scaling in a simple and precise way. Self-similar processes are, by definition, nonstationary, as can be seen from (2.1). The most important subclass, namely self-similar processes with stationary increments (H-sssi processes) are nonstationary in a very homogeneous way. They can be thought as the integral of some stationary process. Fractional Brownian motion (fBm) is the unique Gaussian self-similar process with stationary increments and is the most widely used process to model scaling properties in empirical times series. Self-similar processes with stationary increments, and more specifically fractional Brownian motions, are very attractive models to describe scaling because they are mathematically well defined and well documented. In addition, their great advantage lies in being simple and parsimonious: each of their properties is defined and controlled by the same parameter, H. Their main drawback however, lies in them being simple. It is unlikely that the wide variety of scaling encountered in data can be modeled by a process with a single parameter. The model is overly rigid in several respects. First, definition (1.1) is valid for all positive real c, which means that the scaling exists for all scales or dilation factors ranging from 0 to ∞. Equivalently, one can say, that the scaling relation holds whatever the value of the scaling factor. In actual real world data, scaling can naturally exist only within a finite range of scales and will typically only be approximative. Moreover, one may find evidence for scaling only in the asymptotic regions, i.e., only within the very large (or the very small) scales. Second, self-similarity implies (see (2.1)) that scaling holds for each moment order q (provided it exists), with scaling exponent qH. In empirical data, moments of different orders may have scaling exponents that are not controlled by a single parameter, and some moments may simply not exhibit scaling at all. Even worse, the empirical moments might be misleading when the theoretical moments of the true distribution do not exist at all, as is the case with stable laws. In the case of traffic data, most often scaling models with a single parameter are appropriate at large scales, but at small scales more parameters are required. In rarer

cases, definitive evidence for scaling is lacking altogether. Infinite moments can play a role for quantities such as TCP connection durations, but in term of scaling models, those most commonly used are of the finite (positive) moment type.

2.1.2 PowON-PowOFF self similar traffic model

In this paragraph, a well known model able to generate Self Similar traffic in modern discrete event simulator [LTG95] is presented. Reflecting the real traffic nature, the rate process X in the PowOn-PowOff model is the superposition of M i.i.d. On-Off sources. Each of them, at any time $t \in I_{-\infty} = \{...,-1,0,1,...\}$, can be found in one of two states: On or Off. When the source is On, it generates a traffic with a constant packet rate R; when it is in the Off state the generation process is in the idle state. The time spent by every source in an On state is denoted by $\Theta_{On} \in \{1,2,3,...\}$. Next to this state, the source switches to the idle state with duration $\Theta_{Off} \in \{1,2,3,...\}$. The combination of these two periods constitutes a cycle of length $\Theta_{On} + \Theta_{Off}$. Both random variables Θ_{On} and Θ_{Off} have the following pdf:

$$f_{\Theta_h}(\vartheta_h) = \begin{array}{ll} \gamma A_h^{-1} e^{-\gamma \vartheta_h A_h^{-1}} & 0 \le \vartheta_h \le \mathrm{A}_h \\ \gamma e^{-\gamma} A_h^{\gamma} \vartheta_h^{-(\gamma+1)} & \vartheta_h > \mathrm{A}_h \end{array}$$

where h can be either *On* or *Off*, and $1 < \gamma < 2$. When $t \to \infty$ (then, $t >> A_h$):

$$P(\Theta_h > t) = \int_t^{\infty} \gamma e^{-\gamma} A_h^{\gamma} \vartheta_h^{-(\gamma+1)} d\vartheta_h = A_h^{\gamma} e^{-\gamma} t^{-\gamma}.$$

The above expression means that both the burst duration Θ_{On} and the idle state length Θ_{Off} are characterized by a heavy tailed distribution with a finite mean and infinite variance [WPT98]

Let $\mathbf{Y} = \{Y_k : k = 1,2,...\}$ denote the process representing the number of sources that switch from state Off to state On, and $\mathbf{Z} = \{Z_k : k = 1,2,...\}$ be the process representing the number of sources that switch from state Off to state On. The intensity $\overline{\mathbf{Y}}$ at which the traffic sources enter the On state is given by the following expression:

$$\overline{\mathbf{Y}} = \frac{M}{E[\Theta_{On}] + E[\Theta_{Off}]}.$$

In order to avoid divergence and keep the aggregated traffic intensity unchanged, the average idle duration $E[\Theta_{Off}]$ must increase as M increases. As shown in [CMP+02] and [ZNA03], the superposition of heavy tailed On-Off sources produces an asymptotically self similar rate process X with aggregated autocorrelation function:

$$\lim_{m\to\infty} r^{(g)}(\eta) = \frac{1}{2}\left[(\eta+1)^{3-\gamma} - 2\eta^{3-\gamma} + (\eta-1)^{3-\gamma}\right].$$

From (4), it follows that $H = 3-\gamma/2 > 0.5$. Then, the mean arrival rate, the Hurst parameter, and the FOTS in the PowOn-PowOff model are given by the following expressions [RL00]:

$$\begin{aligned} \overline{\mathbf{X}} &= \overline{\mathbf{Y}} E[\Theta_{On}] R = \frac{E[\Theta_{On}]}{E[\Theta_{On}] + E[\Theta_{Off}]} MR \\ H &= \frac{3-\gamma}{2} \\ T_0^{1-\gamma} &= \gamma^{-1}(2-\gamma)(3-\gamma)\left[(\gamma-1)e^{\gamma}+1\right] R^{-1} A^{1-\gamma} \end{aligned},$$

where $\lambda_{\mathbf{X}} = \overline{\mathbf{X}} = E_{\mathbf{X}}[X_k]$ is the traffic intensity and T_0 represents the FOTS, marking the lower time limits from which point the scaling behavior begins to appear. Note that this expression of the FOTS has been obtained assuming $A_{On} = A_{Off} = A$. This mathematical model allows for an easy computation of the activity ratio δ and the peak to mean ratio φ parameters. The latter is defined as the aggregated sources transmission probability: $\delta = 1-(1-p)^M$, where $p = E[\Theta_{On}]/(E[\Theta_{On}] + E[\Theta_{Off}])$. The former represents the ratio between the peak traffic value *MR* and the mean arrival rate $\overline{\mathbf{X}}$:

$\varphi = MR/\overline{\mathbf{X}} = p^{-1}$.

2.1.3 Long Range Dependence

Long-range dependence (LRD) or long memory [B94] is a model for scaling observed in the limit of the largest scales and is defined in terms of second-order statistics. LRD is usually equated with an asymptotic power-law decrease of the autocovariance function, which should be compared to the exponential one encountered in more classical models (like ARMA processes). An exponential behavior implies, by definition, a characteristic time while a power law, in contrast, is naturally scale invariant.

Let $Y=\{Y_t, t\in R\}$ denote a second-order stationary stochastic process, r_Y and Γ_Y its covariance function and spectral density respectively. We will say that the process $Y=\{Y_t, t\in R\}$ is LRD if either:

$$r_Y(\delta)\approx c_1|\delta|^{\gamma-1},\ \delta\to\infty,\ \gamma\in(0,1) \tag{7.1}$$

$$\Gamma_Y(\delta)\approx c_2|\upsilon|^{-\gamma},\ \upsilon\to 0,\ \gamma\in(0,1) \tag{8.1}$$

with $c_2=2(2\pi)^{-\gamma}\Gamma(\gamma)\sin((1-\gamma)\pi/2)c_1$. In most practical situations, r_Y is regularly varying or even asymptotically monotone, in which case these relations are equivalent. With this definition, the autocovariance function decreases so slowly, the past is so weighty, that its sum diverges, i.e., for any $A>0$:

$$\int_A^\infty r_Y(\delta)d\delta=\infty.$$

All processes with exact self-similarity exhibit LRD. Indeed, let X be a H-sssi process with finite variance. Asymptotically, the covariance function of its increments $Y_\delta(t)$ reads

$$r_{Y_\delta}(s)=E[Y_\delta(t+s)Y_\delta(t)]\approx\sigma^2H(2H-1)s^{2(H-1)},\ \sigma^2=E\left|X_1\right|^2,\ s>>\delta$$

which shows that, for $0.5<H<1$, the increments are LRD processes with $\gamma=2H-1$.

LRD is often theoretically and practically studied through the technique of aggregation. Aggregation consists of studying windowed average versions of the data as a function of the window width T. The covariance functions of the aggregated LRD processes converge to the form given in (4.1) for the fractional Gaussian noise (the increment process of fBm), which is itself invariant under aggregation. This explains its canonical role in analyzing long-range dependence in empirical time series. The variance of the aggregated LRD process also behaves as a power law of the aggregation length with an exponent controlled by γ. This property provides the basis for simple time-domain estimators for the exponent. For traffic data, LRD models have been the most widely used.

2.1.4 Multifractals

The most complete way to describe the scaling behaviour overcoming the drawback of simplicity showed by the self-similar processes, is that of multifractal models. A stochastic process $X=\{X_t, t\in R\}$ with stationary increments is said to be multifractal if:

$$E\left|X_t\right|^q = c(q)t^{\tau(q)+1} \tag{9.1}$$

for every t and some real positive moments q [CF02] [MFC97]. A multifractal process is globally scaling in the sense that its moments satisfy the above scaling relationship. The term $\tau(q)$ is called the *scaling function* of multifractality and is a concave function; $c(q)$ is the *moment factor* and is given by $c(q) = E\left|X_1\right|^q$. In multifractal processes, the scaling functions are non-linear.

Relation (9.1) above is highly reminiscent of the fundamental equation (2.1) implied by self-similarity. A major difference, however, lies in the fact that the scaling function need not a priori follow the linear behavior qH of self-similarity. In other words, to describe the scaling behavior using one single exponent or parameter is impossible and an entire collection of exponents is needed. Multifractal scaling offers an extension to self-similarity insofar as the scaling of moments is no longer governed by one single exponent H but by a collection of exponents. However, it maintains a key feature: moments behave as power laws of the scales.

2.1.5 Selected Applications of Multiscale Traffic Models

We have seen that the complexity and richness of teletraffic is well matched by the multiscale analysis and modeling frameworks of self-similarity, long-range dependence, and multifractals. These frameworks not only allow us to confirm and formalize the presence of multiscale behavior in traffic, but also point to possible causes of multiscale structure in the physical networking infrastructure. The choice of framework, from a simple fBm to a more complicated multifractal or cascade, clearly depends on the application and the data at hand.

Network traffic modeling is important to address several traffic engineering and management problems. Among these, it is worth mentioning routine traffic monitoring, wavelength assignments, route optimization, admission control policies for new data services, traffic policing and shaping, and peering policies [XTF+02]. Analytical traffic models are used to depict dynamic and behavioral traffic characteristics, such as burstiness, statistical distributions, and dependence. Together with network nodes and link models, these help capturing relevant operational features such as topology, bandwidth, buffer space, and nodal service policies (link scheduling, packet prioritization, buffer management, etc.) [RL00]. The model to be used is strictly dependent on the type of traffic management function. The high-complexity of an accurate traffic modeling is useless for some operations, such as route

optimization and wavelength assignment. It is instead necessary for other functions, such as queuing rate adjustment, scheduling operations, and connection admission control, which can be considered as micro control in traffic management [B03], [ENN+00].

A triumph of multiscale analysis techniques in networking has been the discovery of strong scaling phenomena as well as convincing evidence pointing to causes behind it: networking mechanisms, protocols, source characteristics, and so on. But the multiscale concept is applicable to network- related problems beyond the mere analysis of traffic traces. In this section, we briefly outline some applications that directly leverage the multiscale framework. Since the construction of network routers consists largely in combining queues (buffers), queuing analysis plays a crucial role in their design and performance. In the simplest queuing analysis, an aggregate traffic input $X(t)$ is fed into a single-server queue of size B bytes with service rate s bytes/s, and we wish to determine information about $Q(t)$, the queue size in bytes at time t. For example, we might desire the average queue size or the probability that the queue will overflow, the tail queue probability $P(Q>B)$. Queuing analysis in general is extremely difficult, owing to the inherent nonlinearities associated with a queue emptying (few packet arrivals) and overflowing (too many packet arrivals). A distinct advantage of the classical Poisson traffic model for $X(t)$ is the existence of analytic formulae for $P(Q>B)$ [L52]. But the fact that real traffic is not Poisson renders these results of limited utility in real-world situations [PF95]. Another approximate approach is to study only the so-called critical time scale that dominates queue overflow. But as we have seen, real traffic is not typically dominated in a simple way by a single time scale. Real traffic is multiscale, and so we should study the queue size $Q(t)$ at multiple time scales and fuse the results into a single statistic. A multiscale model for $X(t)$ facilitates the investigation of the distribution of $Q(t)$ at multiple scales, incorporating the full multiscale structure. Riedi et al. [RCR+99]-[RRC+00] developed a multiscale queueing analysis in the case of tree-based multiscale input models. Gao et al. simulated queues fed by multiplicative multifractal processes in [GR99]. In [DMM03], a general multifractal process without any restrictions is used to derive an analytical approximation for the queue tail asymptotes.

Furthermore, to understand and predict the performance of end-to-end protocols such as TCP and modern streaming protocols, it is crucial to understand the dynamics of the end-to-end paths through a network. In particular, we could have interest in the delays and losses experienced by packets transmitted end-to-end. Again, help is on the way with a multiscale model [ENN+00], [RCR+00].

3.1 Estimation of Scaling

In traffic management, real-time estimation of the scaling parameters is required [RVA00], [ZC04]. For that matter, different algorithms have been proposed in the past, such as the Variance Time Plot, the Least Squares Regression in the Spectral Domain, the R/S Statistic, the Variogram [B94]. These are aimed at estimating the Hurst parameter *H*. All these methods have two main drawbacks that makes them inappropriate for real-time applications: the lack of accuracy and the high computational complexity. Furthermore all these methods do not capture the complex scaling nature of netwotk traffic.

We saw that diverse signatures of scaling can be observed both with respect to time (regularity of sample paths, slow decay of correlation functions,...), or to frequency/scale (power-law spectrum, aggregation, zooming, small scale increments, etc). This suggests that to identify and characterize scaling an approach which combines time and frequency/scale, and which formalizes properly the idea of a simultaneous analysis at a continuum of scales, should be taken. In this respect, wavelet analysis appears as the most natural framework [AFT+00].

By definition, wavelet analysis acts as a mathematical microscope which allows one to zoom in on fine structures of a signal or, alternatively, to reveal large scale structures by zooming out. Therefore, when a signal or a process obeys some form of scale invariance, some self-reproducing property under dilation, wavelets are naturally able to reveal it by a corresponding self-reproducing property across scales. Moreover, the time-dependence of the wavelet transform allows for a time-localization of scaling features. In its discrete version operating on dyadic scales, the wavelet transform is a rigorous and invertible way of performing a multiresolution analysis, a splitting of a signal into a low-pass approximation and a high-pass detail, at any level of resolution. Iterating the procedure, one arrives at a representation which consists of a low-resolution approximation and a collection of details of higher and higher resolution. From the perspective of more classical methods used for scaling data, iterating low-pass approximations, at coarser and coarser resolutions, is an implicit way of aggregating data, whereas evaluating high-pass details, as differences between approximations, is nothing but a refined way of computing increments (of order N for a wavelet with N vanishing moments). Combining these two key elements makes of multiresolution a natural language for scaling processes.

To summarize, the wavelet transform closely reproduces the scaling properties that exist in data, be it self-similarity, long-range dependence, or multifractality, and, at the same time, replaces one single poorly behaved (nonstationary, LRD) time series by a collection of much better behaved sequences

(stationary, SRD), amenable to standard statistical tools. Therefore, second-order statistical scaling properties can be efficiently estimated from marginalized scalograms (squared wavelet coefficients averaged over time), circumventing the difficulties usually attached to scaling processes.

Another advantage enjoyed by wavelets is their insensitivity to deterministic trends which may be superimposed onto a process of interest, with undesirable consequences. These include invalidating the stationarity property of the LRD process under study or mimicking LRD correlations when added to a short-range dependent process [AFT+00]. Wavelets are a versatile solution to this crucial issue, since they offer the possibility of being blind to polynomial trends. Recall that any admissible wavelet has zero mean. This is equivalent to having a zeroth-order vanishing moment, or in other words, to be orthogonal to constants. In fact N vanishing moments implies that the wavelet is blind to polynomials up to order p $N \leq -1$. Trends which are "close" to polynomial can be effectively eliminated in this manner, and the advantage of being able to do so without even testing for their presence is an important one when making sense of real data, and in particular when trying the distinguish nonstationarity from scaling behavior. Building on the advantages of the wavelet approach, a statistical test for the constancy of a scaling exponent can be defined which helps resolve this difficult issue. Finally, the analysis of scaling processes is often faced, and particularly so in the case of teletraffic, with enormous quantities of data, thereby requiring methods which are efficient from a computational point of view. Because of their multiresolution structure and the related ability to be implemented as a filter bank, wavelet-based methods are associated with fast algorithms, out performing FFT-based competitors with a complexity of only $O(n)$ in computation (compared to $O(n\log(n))$) and $O(\log(n))$ in memory, for n data points. These advantages hold not only at second order, but more generally, including for the more advanced types of analysis.

In [VA99], Veitch and Abry propose an estimator for the two most important parameters that characterize the long-range dependence phenomenon. The algorithm is based on computing the wavelet-transform of the traffic trace, performing a linear fit from wavelet coefficients, and measuring the slope and the intercept of the fit. This is a simple algorithm that provides an estimator with low variance. More recently, in [DMM03] a simple graphical method for multiscaling functions estimation has been proposed. This algorithm however suffers of a high computational complexity.

Estimation of multifractal parameters is a delicate task, which needs to be performed taking into account the resulting accuracy. This is strictly dependent on the number of considered traffic samples. To obtain an estimate with a certain confidence interval, the input traffic traces have to be of a minimum length. While this problem has always been addressed by the algorithms for parameter

estimation proposed in the past, the complementary issue has never been considered, namely, the maximum length of the input traffic traces. This issue is unfounded from a statistical point of view. In fact, the accuracy increases as the length of the trace increases. However, there is a match between the estimated multifractality and the observed process only if its scaling behavior is homogeneous over the considered interval of time. The variability of the multifractality over time introduces a mismatch between actual and estimated values. Such a mismatch may affect the performance of the subsequent traffic management procedure. This issue is addressed in this work, where we proposed a new strategy for real-time scaling parameter estimation. It is adaptive as the length of the estimation interval is adjusted based on local traffic features in order to extend the number of estimation intervals as long as traffic seems to have a homogeneous behavior.

3.1.1 Multiscale Diagram Estimator

This section is intended to provide a description of the estimator proposed by Abry and Veitch in [VA99] along with the mathematical details that are necessary to follow the description of the proposed algorithm.

The accurate algorithm for the estimation of the scaling parameters presented by Veitch and Abry, makes use of the wavelet transform and is commonly referred to as *Multiscale Diagram Estimator* (MD). Accordingly, let $d_X(j,k)$ denote the coefficients of the discrete or non-redundant wavelet transform of a process *X*, where *j* and *k* index the scale of analysis and the coefficients within each scale, respectively. The family of wavelet basis functions is generated from the mother wavelet ψ_0. For a multifractal process *X*, the wavelet coefficients $d_X(j,k)$ are such that:

$$E\left|d_X(j,k)\right|^q = 2^{j\alpha(q)}c(q)C_{\alpha,\psi_0}, \qquad (10.1)$$

with $\alpha(q)=\tau(q)+1+\frac{q}{2}$. As shown in [VA99], $c(q)C_{\alpha,\psi_0}$ has statistical properties very similar to those of $c(q)$ when *q*=2. The partition functions defined as $\mu_j(q)=\frac{1}{n_j}\sum_k\left|d_X(j,k)\right|^q$ represent an unbiased estimate of $E\left|d_X(j,k)\right|^q$, where n_j is the number of wavelet coefficients at scale *j*. Accordingly:

$$\log_2 E\left[\mu_j(q)\right] = j\alpha(q) + \log_2\left(c(q)C_{\alpha,\psi_0}\right). \quad (11.1)$$

For simplicity of notation, from now on we use $a(q)$ to refer to $\log_2\left(c(q)C_{\alpha,\psi_0}\right)$. (11.1) suggests to use a linear regression approach to estimate $\alpha(q)$ and $a(q)$. In fact, the slope of the regression would estimate $\alpha(q)$ and the intercept would be related to $a(q)$. However, due to the non linearity introduced by the logarithm, this operation doesn't provide accurate results and the following estimate is proposed in [AFT+00]:

$$E\left[\log_2 \mu_j(q)\right] - g_j(q) = j\alpha(q) + a(q). \quad (12.1)$$

Where the corrective term $g_j(q)$ is defined as follows:

$$g_j(q) = -\frac{1}{2n_j \ln 2}\frac{Var\left|d_X(j,k)\right|^q}{E\left|d_X(j,k)\right|^q{}^2} = \frac{-V(q)}{2n_j \ln 2}. \quad (13.1)$$

Assuming the wavelet coefficients to be Gaussian, $V(q)$ is equal to $\sqrt{\pi}\,\Gamma(q+0.5)/\Gamma^2((q+1)/2) - 1$ (where $\Gamma(x)$ is the gamma function) and depends only on q.

The MD estimator is then based on applying a weighted linear regression to $y_j(q) = \log_2\left(\mu_j(q)\right) - g_j(q)$, since the fundamental hypothesis is now valid: $E\left[y_j(q)\right] = j\alpha(q) + a(q)$. Given a range of scales $j_1 \div j_2$, the unbiased estimators for $\alpha(q)$ and $a(q)$ are:

$$\begin{aligned} \hat{\alpha}(q) &= \sum_{j=j_1}^{j_2} w_j y_j(q) \\ \hat{a}(q) &= \sum_{j=j_1}^{j_2} v_j y_j(q) \end{aligned}, \quad (14.1)$$

where w_j and v_j are the weights of the regression. The variance of the estimators are:

$$\begin{aligned} \sigma^2_{\hat{\alpha}(q)} &= \sum_{j=j_1}^{j_2} \sigma_j^2(q) w_j^2 \\ \sigma^2_{\hat{a}(q)} &= \sum_{j=j_1}^{j_2} \sigma_j^2(q) v_j^2 \end{aligned}, \quad (15.1)$$

where $\sigma_j^2(q) = \mathrm{var}\left(y_j(q)\right) = V(q)/n_j \ln^2 2$.

The process multifractality is fully described by $\alpha(q)$ and $c(q)$ and these parameters are those commonly used for resource allocation purposes. However, in our analysis of multifractal temporal variability we refer to $\alpha(q)$ and $a(q)$, in view of the fact that for these the estimation accuracy can easily be computed.

3.1.2 Variability of Multifractal Behavior

The first objective of our study is to analyze the variability of the multifractal behavior over time in real traffic traces. In particular, we are interested in evaluating the extent of changes in multifractal behavior between adjacent intervals of time, in terms of its crucial parameters. As far we are concerned, this type of analysis has never been applied to network traffic data in the past. Instead, in other fields, such as image and video processing, it is frequently performed and several applications rely on it. Among these applications, object segmentation and image understanding are the most representative [GT01].

To conduct this analysis, we have defined a simple heuristic procedure. It is based on dividing the entire traffic trace into consecutive sub-traces and to extract the multifractal features related to each one. The length of each sub-trace is critical. On one hand, the temporal resolution of our analysis benefits from using short sub-traces, where the temporal resolution is connected with the frequency of estimates over time. On the other hand, the accuracy in parameter estimation increases as the number of samples in each sub-trace increases. An appropriate compromise has to be found. To partially solve this problem, we make use of temporally overlapped sub-traces. At a given temporal resolution, this approach improves the accuracy of the estimations. Next, we describe the proposed procedure.

Given a traffic trace T_k ($k = 1,...,M$), where k indexes the samples, we divide it in L *base windows* of size N (in number of samples), such that $M = L \cdot N$. Herein, the base windows correspond to the sub-traces previously mentioned. We are interested in estimating the multifractal parameters $\hat{\alpha}_i(q)$ and $\hat{a}_i(q)$ for every base window i (i=1,...,L) by computing according to (14.1). Note that we add subscript i to the notation to refer to window i. The parameters are estimated for scales in the range $j_1 \div j_2$ and moment order in the range $q_1 \div q_2$. As anticipated, the choice of N is critical, since it influences the accuracy of every estimation. To evaluate the accuracy, we refer to the 95% confidence

interval $\Delta CI_{ij}(q)$ of each estimated $y_{ij}(q)$. Since $y_{ij}(q)$ is a Gaussian random variable [VA99], we can easily obtain the resulting 95% confidence interval:

$$\Delta CI_{ij}(q) = 3.92\sqrt{\sigma_{ij}^2(q)} = 3.92 \cdot \sqrt{V(q) \big/ \left(n_{ij} \ln^2 2\right)} \quad . \tag{16.1}$$

We have then used the ratio between this expression and the expected estimator value to evaluate the accuracy; indeed, the absolute value of the confidence interval is not of great significance alone. The accuracy is then controlled by enforcing the ratio between $\Delta CI_{ij}(q)$ and $\hat{y}_{ij}(q)$ to be lower than a selectable threshold ρ :

$$\rho_{ij}(q) = \Delta CI_{ij}(q) \big/ \hat{y}_{ij}(q) \le \rho \,, \tag{17.1}$$

for $i = 1,...,L$, $j = j_1,...,j_2$, and $q = q_1,...,q_2$. This constraint is applied to each scale and moment order. However, we have experimentally observed that the higher the moment order and the scale, the higher $\rho_{ij}(q)$ is. This can partially be realized by noting that n_{ij} decreases as scale *j* increases and that $V(q)$ increases as *q* increases. We then verify (17.1) only for $j = j_2$ and $q = q_2$:

$$\rho_{ij_2}(q_2) = \left(3.92 \big/ \hat{y}_{ij_2}(q_2)\right) \cdot \sqrt{V(q_2) \big/ \left(n_{ij_2} \ln^2 2\right)} \le \rho \,. \tag{18.1}$$

The number *N* of samples may not be enough to satisfy such a constraint for every *i*. Accordingly, for each base window, we compute $\rho_{ij_2}(q_2)$. If the constraint is not satisfied, it is extended by introducing an overlapping of adjacent windows. In this case, the size changes from *N* to N_i, so that:

$$N_i \ge \left(3.92 \big/ \left(\hat{y}_{ij_2}(q_2) \cdot \rho\right)\right)^2 \cdot \left(2^{j_2} V(q_2) \big/ \ln^2 2\right) = \phi \,. \tag{19.1}$$

This comes out by using $n_{ij_2} = N_i \big/ 2^{j_2}$. Note that the number of base windows in the trace remains unchanged and equal to *L*. This brings to an overlap of $N_i - N$ samples for window *i*. To measure the extent of the overlapping, we have defined the following overlapping coefficient: $\vartheta_i = (N_i - N)/N$, which obviously depends on the selected ρ . ϑ_i has been defined just to have a measure of the level of overlapping introduced in the analysis. The higher the desired accuracy in our evaluation, the higher the overlapping coefficient is.

Next, we present the results of the analysis of the multifractality variability when applying the described method to two real traffic traces.

3.1.3 Auckland-II archive

It is an archive of several traces taken from both directions of the access link connecting Auckland University to the Internet [MGB00]. Herein, we present the results of our analysis for a trace providing the packet inter-arrival time for a period of 2 hours and 18 minutes, and a total of 8,199,021 samples (see Figure 1.1). We have considered base windows of an initial length *N* of 73728 samples in order to have 16 wavelet coefficients at scale 2^{12}. This yielded to *L*=110 base windows. To satisfy the constraint in (18) with $\rho = 0.1$, we have changed the length of 16 windows and obtained a maximum overlapping factor of 0.58. We have estimated $\hat{\alpha}_i(q)$ and $\hat{a}_i(q)$ for each window by means of the MD estimator at scales $2^8 \div 2^{12}$ and for moments $q = 2,4,6$. This scale has proved, after several attempts, to provide the best results in terms of precision in the linear regression of the MD algorithm. It is important to underline that an appropriate choice of the scales is crucial for a correct use of the estimator. Figure 2.1 shows the results for only 30 base windows for presentation convenience. However, this behavior is representative for the entire trace. Note that the trend is irregular, especially for the higher-order moments. However, some homogeneous zones can be identified in both $\hat{\alpha}_i(q)$ and $\hat{a}_i(q)$ curves, such as those related to base windows 50-53, 66-67, 69-72, 74-75, and 76-77. To better analyze the results, in Table 1.1 we provide average and variance for the two parameters for all the 110 base segments at different *q*; we also provide the mean absolute difference between adjacent values (*D*). These outcomes highlight that $\hat{\alpha}_i(q)$ is characterized by a lower variance with respect to $\hat{a}_i(q)$. In both cases, the variance increases as the moment order increases. This is true also for *D*. Additionally, in Table 2.1 we provide the absolute difference between adjacent estimates for two groups of base windows. In the first one (windows 60-62) the absolute difference is higher than the average while in the second one is lower than the average. This represents an explicatory case of the

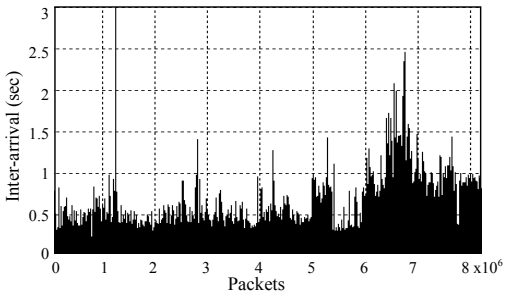

Figure 1.1. Packet inter-arrival times for the Auckland-II trace.

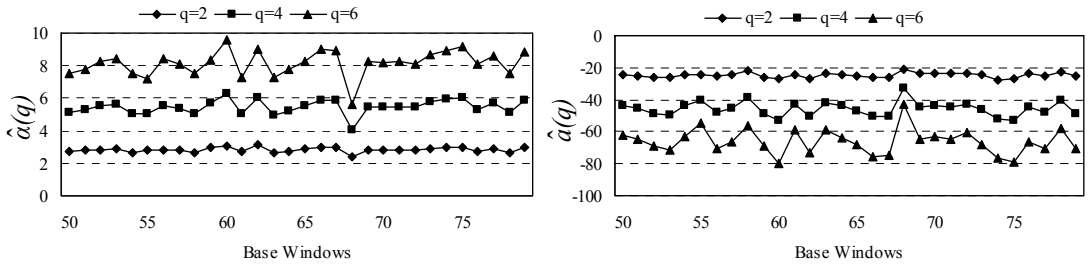

Figure 2.1. Multifractal parameters estimates for the Auckland-II packet inter-arrival measures. These graphs refer to base windows 50÷79.

Table 1.1. Average, variance and mean absolute difference between adjacent windows (*D*) for estimates $\hat{\alpha}_i(q)$ and $\hat{a}_i(q)$.

	$\hat{\alpha}_i(q)$			$\hat{a}_i(q)$		
	$q=2$	$q=4$	$q=6$	$q=2$	$q=4$	$q=6$
μ	2.83	5.48	8.15	-24.54	-45.13	-64.85
σ^2	0.03	0.24	0.69	5.93	38.03	101.25
D	0.18	0.48	0.83	1.9219	5.08	8.66

Table 2.1. $|\hat{\alpha}_{i+1}(q)-\hat{\alpha}_i(q)|$ and $|\hat{a}_{i+1}(q)-\hat{a}_i(q)|$ values for base windows 60÷62 and 69÷71.

	$\lvert\hat{\alpha}_{i+1}(q)-\hat{\alpha}_i(q)\rvert$			$\lvert\hat{a}_{i+1}(q)-\hat{a}_i(q)\rvert$		
	$q=2$	$q=4$	$q=6$	$q=2$	$q=4$	$q=6$
60	0.34	1.20	2.35	2.82	10.54	21.06
61	0.40	1.02	1.76	2.92	7.98	14.18
62	0.47	1.09	1.78	3.70	8.59	14.23
69	0.03	0.01	0.06	0.71	0.80	1.38
70	0.02	0.01	0.07	0.36	0.61	1.52
71	0.02	0.001	0.15	0.06	1.49	4.53

temporal traffic behavior in terms of multifractality. It highlights that notwithstanding the general variability of the estimated values, it is possible to find some adjacent windows belonging to a period with uniform multifractal behavior.

3.1.4 Digital Equipment Corporation trace

It contains an hour worth of all wide-area traffic between Digital Equipment Corporation and the rest of the world [WEB]. This trace provides the cumulative packet inter-arrival time for a period of 1 hour and a total of 2,153,462 samples. Similar to the Auckland-II sequence, we have fragmented such trace into several sub-traces. Due to the difference in trace length, we have obtained L=23 base windows, each of N=93194 samples (45 wavelet coefficients at scale 2^{11}). Using $\rho = 0.1$ for the accuracy condition, we had never needed to change the length of the base windows so that no overlapping has been introduced. Figure 3.1 show the resulting $\hat{\alpha}_i(q)$ and $\hat{a}_i(q)$ for each window applying the MD in the scales range $2^7 \div 2^{11}$ and for moments $q = 2,4,6$. Also in this case, the scale has been selected so as to encompass the wavelet decomposition levels that allowed for a correct application of the MD estimate. Figure 3.1 shows the results for all the 23 base windows. Note that also this trace is characterized by a heterogeneous multifractal behavior, except for some regions: 1-2, 9-12, 16-17, and 18-19.

From these results and those presented in the previous sub-section, we can conclude that the scaling behavior mostly varies over time. However, there are same periods of time over which the multifractality seems to be homogeneous. In other words, among the total amount of traffic buckets, we can identify some regions characterized by very similar scaling features. There are different events that may trigger a change in the multifractality of the process under observation: e.g., re-routing of traffic

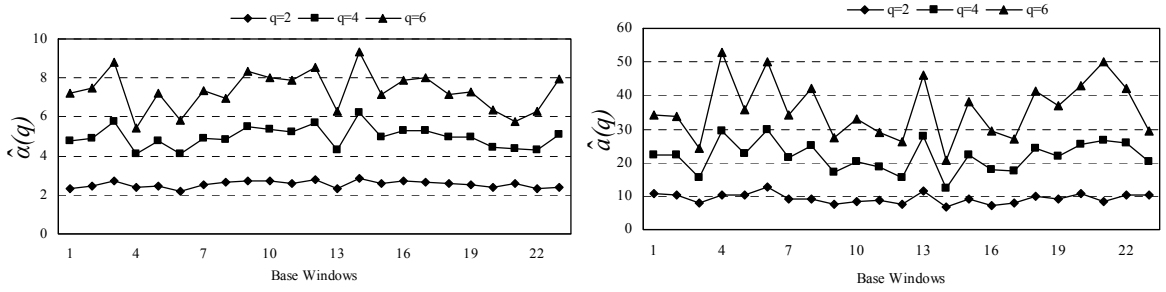

Figure 3.1. Multifractal parameters estimates for the Digital Equipment Corporation trace. These measures refer to the cumulative inter-arrival times.

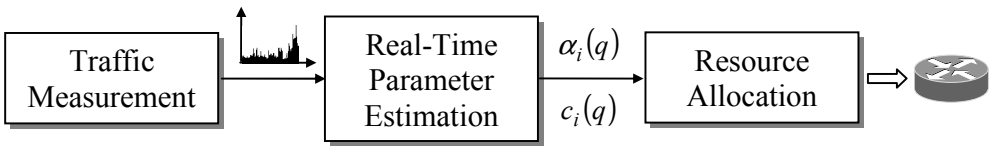

Figure 4.1. Real-time measurement-based resource allocation framework.

flows; changes in end-user behavior; link failures; traffic management actions; or simply arrivals of new flows that have statistical properties different from flows in progress. The analysis of the cause of this phenomenon would be of great interest; however, it is not the objective of this work. We are just interested in observing this phenomenon, whatever the underlying causes are, to better design an adaptive algorithm for the estimation of the scaling parameters.

3.1.5 Real-Time Estimation of Multifractality

Figure 4.1 sketches a real-time measurement-based framework of resource allocation that employs a multifractal model of the network traffic. In this framework, high temporal resolution traffic measurements are performed; the measurement output data is used for parameter estimation of the model and the results are used to drive the resource allocation procedure: e.g., bandwidth reservation, traffic routing, connection admission control (CAC). To provide an accurate traffic modeling, parameter estimation scaling has to be applied over a long interval of time. In fact, the estimator is characterized by a variance inversely proportional to the number of used samples: the higher the number of samples the more accurate the estimations are. The temporal multifractal variability, highlighted by the previous off-line analysis, has to be taken also into account in this framework.
It means that the estimation has to be performed during an interval of time with uniform multifractal behavior in order to have meaningful results.
Based on these considerations we argue that an adaptive approach for the estimation of the model parameters is required. It should be aimed at solving the conflict between estimation accuracy and temporal variability of multifractal behavior. Next, we describe our proposed adaptive estimation algorithm, named AE, whose flow graph is depicted in Figure 5.1. According to our algorithm, values of estimated parameters are provided time-by-time with a frequency that can be controlled by means of an appropriate parameter, as it will be clear later. The estimations are indexed with i (i=1,2,3,…) and refer to overlapped windows W_i of N_i samples.

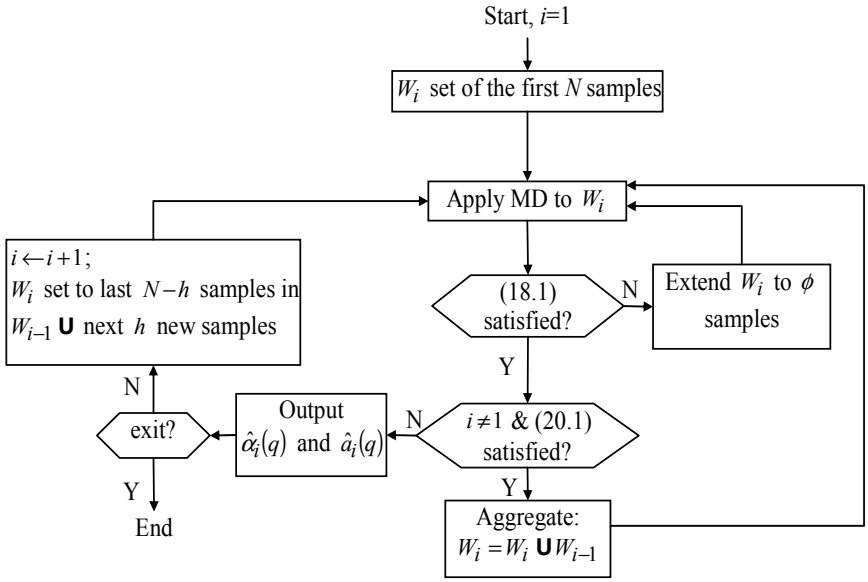

Figure 5.1. Flow diagram of the proposed adaptive real-time estimation algorithm (AE).

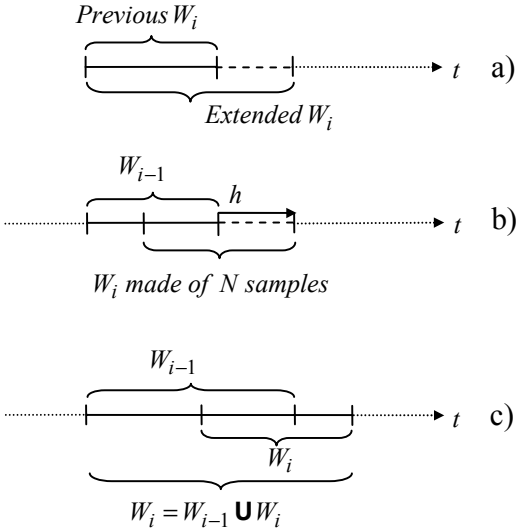

Figure 6.1. Setting of the size for window i in three different situations: a) extension of window size to satisfy accuracy constraint (18.1); b) selection of the initial N samples for a new window W_i ; c) aggregation of two consecutive windows that satisfy conditions (20.1).

Note that, as already stated, in the AE we use the parameters $\alpha(q)$ and $a(q)$ to analyze the multifractality temporal variability and to "control" the length of the segments accordingly, but $\alpha(q)$ and $c(q)$ are those which represent the final output of the estimation as depicted in Figure 4.1.

To describe the procedure, we start by considering the first window W_1. It is initially set to the first N measured values from the beginning of the application of the AE algorithm. Its size is dynamically adjusted based on the constraint expressed by (9.1). If constraint (9.1) is not satisfied, we widen W_1 to

include additional $\phi - N$ samples (see Figure 6.1.a). Clearly, in this case the estimation is delayed by the time necessary to collect these new samples. At the end of this operation, the estimations $\hat{\alpha}_i(q)$ and $\hat{a}_i(q)$ satisfy our constraint on the accuracy. Since we are in the first window (*i*=1), we move to the second window without considering a possible aggregation (see graph in Figure 5.1); at the same time the AE algorithm provides the estimated values for the first window to the resource allocation module. Window W_2 is initially set to the last $N-h$ samples of W_1 plus additional h new samples; h represents the advancing step. Figure 6.1.b illustrates this procedure for two generic consecutive windows W_{i-1} and W_i. As for the previous window, the window size is extended to satisfy (18.1). As opposed to the previous case of *i*=1, we now verify if the two adjacent windows can be aggregated. In fact, the off-line analysis conducted in the previous section showed that adjacent windows may have the same multifractal behavior. In this case, the scaling parameters can be estimated from the samples of the two windows to increase the estimation accuracy. The adjacent windows *i* and *i*-1 (at this point of our description *i*=2 and *i*-1=1) are considered to have the same multifractal properties if the estimated parameters are "similar" for every moment order. The similitude is evaluated by testing the null hypothesis H_0 : the means of the estimated parameters are identical for the two adjacent windows. This is done through a *statistical test* [L86] based on the following statistics:

$$S(\alpha) = \frac{(\hat{\alpha}_{i-1}(q) - \hat{\alpha}_i(q))^2}{\left(\sigma^2_{\hat{\alpha}_{i-1}(q)} + \sigma^2_{\hat{\alpha}_i(q)}\right)}, \tag{20.1.a}$$

$$S(a) = \frac{(\hat{a}_{i-1}(q) - \hat{a}_i(q))^2}{\left(\sigma^2_{\hat{a}_{i-1}(q)} + \sigma^2_{\hat{a}_i(q)}\right)}. \tag{20.1.b}$$

Under H_0, $S(\alpha)$ and $S(a)$ are distributed as a Chi-squared variable with 1 degree of freedom with density function f_1. A critical region boundary C for $S(\alpha)$ and $S(a)$ is identified in terms of a significance level β :

$$\int_C^{\infty} f_1(x)dx \equiv \beta . \tag{21.1}$$

The test reads as follows:

if $S(\alpha) \le C$ and $S(a) \le C$, accept H_0,

otherwise, reject H_0.

In case of aggregation, the MD procedure is applied again to the new W_i window that is made of the samples of the previous W_i plus those of W_{i-1}. Figure 6.1.c illustrates this procedure. The advantage of the aggregation lies in the fact that resulting estimates are characterized by a lower variance due to the increased number of wavelet coefficients compared to the case of not aggregation. Whether the aggregation has been introduced or not, the algorithm moves to the next sub-trace, i.e, *i*=3. Then, the steps performed for window 2 applies to this window and continues until the end of the real-time estimation algorithm.

N-*h* samples of one window are used in the next with the aim to develop a procedure where the temporal resolution and the estimation statistical accuracy can be controlled independently. The number of samples used for each window is set on the basis of the given target of the statistical accuracy. Instead, *h* controls the resolution in the detection of the traffic multifractality temporal variability. To describe the advantages of this choice, we introduce a simple example depicted in Figure 7.1, which refers to a traffic trace with a discontinuity in the multifractality characteristics (grey and black areas in the time axis in the figure). The ultimate aim of an adaptive real-time procedure is to identify this change, separately using the two groups of samples in each area to estimate the parameters with high accuracy. On the basis of this target, we have analysed the effects of using different combinations of *N* and *h* in terms of the resulting windows which have been drawn in the figure. For presentation purposes, we have assumed that condition (18.1) is satisfied for every window by using the initial set of *N* samples without requiring any widening. In cases I and II, two values of *N* has been used (N_1 and N_2, respectively) and *h* has been set to *N* (no overlapping). In case I, the first three segments are likely to be aggregated into a wider one, whereas the last three would make another segment, getting a satisfying segmentation. Increasing *N* (case 2), the third segment overlaps the traffic

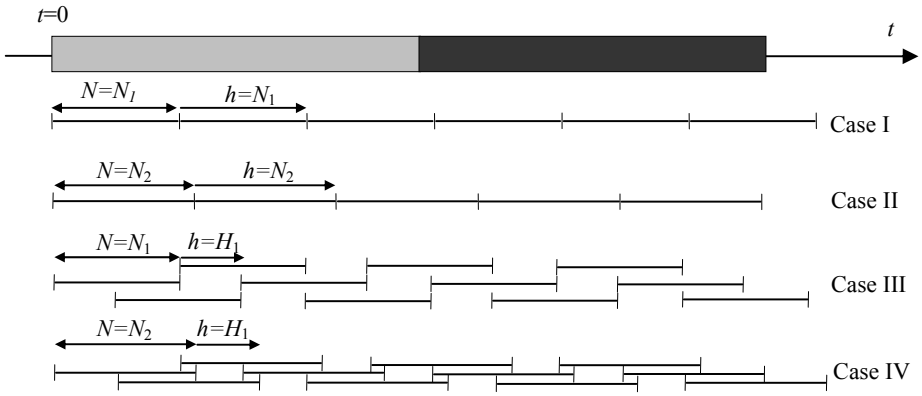

Figure 7.1. Influence of *h* on finding the change in multifractal behavior.

change bringing to inaccurate estimates. In fact, this segment would be considered a separate segment, included in the first group, or included in the second group not reflecting the real condition in all situations. In the other two cases, we have introduced the overlapping ($h = H_1 < N_1 < N_2$) with N equal to either N_1 or N_2. In these cases, the first six windows would be grouped together and the second group would start from the seventh window, with accurate detection of the traffic change. Note that generally, the occurrence of a change in the multifractal behavior can be detected every *h* samples, with $h \in 0 \div N$.

3.1.6 Computational complexity

A major concern on the applicability of the proposed technique is its computational complexity, which is mainly affected by the MD algorithm. This algorithm can be considered made of three main operations:

- Application of wavelet filters to extract coefficients $d_X(j,k)$;
- Computation of the estimator $y_j(q)$ for every scale *j* and moment order *q*;
- Estimation of the multifractal parameters through weighted linear regression.

Note that the MD estimation has to be carried out in three different circumstances in our algorithm, which correspond to the three situations depicted in Figure 6.1. The most frequent is when $\hat{\alpha}_i(q)$ and $\hat{a}_i(q)$ are extracted from *N* collected samples (Figure 6.1.b). In this case, as proposed in [RVA00], operations i) and ii) can be performed in real-time by using a pyramidal filter bank and computing some partial sums for every new collected sample. These partial sums are $\sum_i |d_X(i,j)|^q$ and $\sum_i |d_X(i,j)|^{2q}$ (see (13.1)), which are required to compute $y_j(q)$. Operation iii) is then performed when all the N_i samples have been collected; its computational complexity is of $O(\log_2 n)$ with a memory requirement of $O(\log_2 n)$ and therefore there are no computational difficulties for a real-time application of the AE algorithm.

Another circumstance is when window W_i has to be extended from N_i to ϕ samples (Figure 6.1.a).

It means that it is not possible to compute the wavelet coefficients and the partial sums while collecting traffic samples; in fact, the first N_i have already been collected in this case. This may introduce some

problems in a real-time framework. However, for these N_i samples the estimator has already been computed at this point. It means that this problem can be easily overcome by using the partial sums computed during the previous estimation. The other circumstance is when two windows W_{i-1} and W_i have to be aggregated and the multifractal parameters for these windows have already been computed (Figure 6.1.c). Also in this case, it is important to re-use the partial sum previously computed for the estimations for the windows to be aggregated. In this way, it is easy to perform operation iii) to get $\hat{\alpha}_i(q)$ and $\hat{a}_i(q)$ for the new aggregated window.

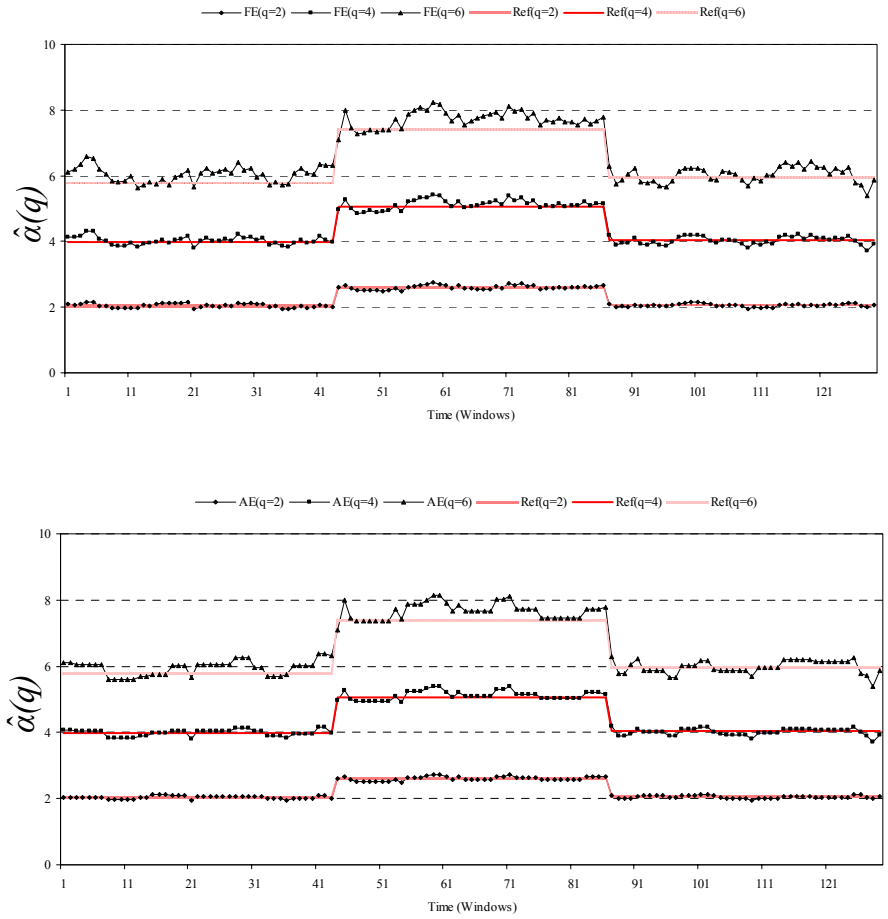

Figure 8.1. Comparison of the FE (top) and AE (bottom) estimates of $\hat{\alpha}(q)$ with respect to the reference values (Ref).

3.1.7 Numerical Simulations

To evaluate the benefits of the proposed adaptive strategy, we have conducted several experiments aimed at comparing it with a fixed estimation (FE) approach. The fixed algorithm works as the proposed AE algorithm does, except for the aggregation feature that is not used. The experiments are based on synthetic traffic generated with OPNET Modeler. We simulated a superimposition of ON-OFF sources at different rates and scaling features, with a heavy-tailed distribution of both the on and off periods. It has been proved that this aggregation results in a multifractal process [XB02]. The aim was to have traffic traces with known multifractal features in order to evaluate the performance of the estimation algorithms.

In this section, we present the results relevant to a synthetic sequence made of three 1 hour long homogeneous multiscaling periods, each made of a total of about 825,000 samples. Each hour has different characteristics. Applying the MD algorithm to each period in isolation, using all the 825,000 samples, we have computed the actual multifractal parameters. These constitute the reference values in our experiments, being the target values for a parameter estimation procedure. With N=46,080, an advancing step h of 18,432, and $\rho = 0.01$, the FE obtains 130 windows whose $\hat{\alpha}_i(q)$ estimates are shown in the graph on top of Figure 8.1. We can see that, for almost all the windows there is a significant mismatch between the FE estimated values and the reference ones. This denotes that the overlapping strategy is not adequate to avoid the oscillation of results around the correct values. We have repeated the experiment with the AE algorithm, using the same previous settings and with a significance level of 90%. The resulting curves are much smoother and closer to the reference ones compared to the FE ones, denoting a higher accuracy (see graph on bottom of Figure 8.1). The presence of some spikes in windows 21, 45, 53, and 128, is due to inaccurate estimations of AE; in fact, these windows have not been aggregated with the surrounding ones since the statistical test wes not satisfied. Indeed, we could have used lower values for β, triggering the aggregation of all the windows in the homogeneous regions. In a simulation framework this could be done since we have a priori knowledge of the uniform region extensions. On the other hand, in a real context, this could lead to incorrect aggregations because estimates related to different multifractal regions could be erroneously considered similar. The choice of the significance level is crucial since this heavily affects the resulting accuracy. A wrong setting may cause the algorithm to introduce erroneous aggregations or to miss proper aggregations.

Table 3.1. Some estimates of α and its variance in some windows where the hypothesis H_0 has not been satisfied. The last two columns show the range of values that would have resulted in fulfilling the similarity condition $S(\alpha)$, while keeping the other parameters unchanged.

	$\hat{\alpha}(2)$	$\sigma^2_{\hat{\alpha}}(2)$	$\not\sigma^2_{\hat{\alpha}}(2)$	$\not\alpha(2)$
21	1.9364	0.0038	≥ 0.970228	2.0573 ± 0.010753
22	2.0573	0.0039	≥ 0.970355	1.9364 ± 0.010753
36	1.9386	0.0038	≥ 0.231343	1.9980 ± 0.010757
37	1.9980	0.0039	≥ 0.231465	1.9386 ± 0.010757

Table 4.1. Some estimates of a and its variance in some windows where the hypothesis H_0 has not been satisfied. The last two columns show the range of values that would have resulted in fulfilling the similarity condition $S(a)$, while keeping the other parameters unchanged.

	$\hat{a}(2)$	$\sigma^2_{\hat{a}}(2)$	$\not\sigma^2_{\hat{a}}(2)$	$\not a(2)$
21	18.1862	0.3040	≥ 82.76888	17.0699 ± 0.096237
22	17.0699	0.3135	≥ 82.77835	18.1862 ± 0.096237
36	18.3017	0.3046	≥ 31.75602	17.6081 ± 0.096283
37	17.6081	0.3135	≥ 31.76491	18.3017 ± 0.096283

To better understand the role of the statistical test in the AE algorithm, in Tables 3.1 and 4.1, we show two cases of adjacent windows (21-22, 36-37) where the similarity condition has not been satisfied. For simplicity, we show the estimates $\hat{\alpha}$ and $\hat{a}$ for the moment order $q=2$ only. In column $\not\alpha(2)$, we have put the range of values for the estimated alpha that would have resulted in fulfilling the similarity condition for $S(\alpha)$, while keeping the other parameters unchanged. The other columns $\not\sigma^2_{\hat{\alpha}}(2)$, $\not a(2)$, and $\not\sigma^2_{\hat{a}}(2)$ have the same meaning. For instance, in windows 21 and 22 this condition (would have been fulfilled in case of $\sigma^2_{\hat{\alpha}_{21}}(2) \geq 0.970228$ and keeping the other parameters unchanged ($\hat{\alpha}_{21}(2), \hat{\alpha}_{22}(2), \sigma^2_{\hat{\alpha}_{22}}(2)$). As expected, reducing the gap between the adjacent estimates and increasing their variance make the aggregation more likely.

Table 5.1. MSE for the FE and AE algorithms.

		q=2	q=4	q=6
$\hat{\alpha}(q)$	FE	0.0035	0.0176	0.1173
	AE	0.0026	0.0134	0.0835
$\hat{a}(q)$	FE	0.2742	1.4243	10.3131
	AE	0.2005	1.0703	7.4568

Table 6.1. Multifractal features for the synthetic traffic used to evaluate the performance of the FE and AE algorithms in terms of estimated equivalent bandwidth. The last two rows provide the egress buffer size and the target blocking probability.

	Hour 1	Hour 2	Hour 3
$\hat{c}(2)$	1.47E+06	1.53E+06	1.48E+06
$\hat{c}(4)$	3.00E+13	3.36E+13	2.98E+13
$\hat{c}(6)$	1.25E+21	1.59E+21	1.22E+21
$\hat{\alpha}(2)$	2.0451	2.5968	2.0654
$\hat{\alpha}(4)$	3.9854	5.0732	4.0451
$\hat{\alpha}(6)$	5.793	7.3914	5.9472
$b\,(Mbit)$	0.5	0.5	0.5
$\log[P(Q > b)]$	4.6E-14	4.6E-14	4.6E-14

We have also compared the distance of the FE and AE curves respect to reference curves in terms of MSE. In Table 5.1, we show the MSE for both $\hat{\alpha}(q)$ and $\hat{a}(q)$. Note that the AE's MSE is quite lower than the FE's MSE for all the considered moment orders, numerically underlying the higher accuracy of the AE respect to the FE. It is worth noting that the obtained results also depend on the setting of ρ. In these experiments, we have set ρ so as to find an appropriate compromise between the initial parameter estimation accuracy and the responsiveness of the estimation algorithm. Indeed, the lower this parameter is set, the quicker the system reacts to traffic changes.

The MSE measures provide a good indication of the estimation accuracy. However, it is more interesting to evaluate the influence of the proposed strategy on the overall resource allocation procedure (framework sketched in Figure 4.1). This evaluation should refer to a specific resource allocation procedure. To accomplish this, we have considered a CAC procedure accepting incoming service calls on the basis of the equivalent bandwidth, which represents the minimum required

bandwidth such that the QoS requirements can be achieved [KZZ96]. In [DMM03], the expression of the asymptotic blocking probability ($\log(P[Q>b])$) for a queuing system with buffer size b fed with a multifractal input traffic is provided. It is a function of the service rate s, which can be approximated with the equivalent bandwidth. From this expression we can write the equivalent bandwidth in terms of the desired blocking probability and the multifractal characteristics of the input traffic:

$$\log s = \frac{\min_{q>0} \log \frac{c(q)\left(\frac{b\tau_0(q)}{q-\tau_0(q)}\right)^{\tau_0(q)}}{\left(\frac{bq}{q-\tau_0(q)}\right)^{q}} - \log(P[Q>b])}{\min_{q>0}(\tau_0(q))} \qquad (21.1)$$

where $\tau_0(q)=\tau(q)+1$.

The real-time estimated scaling parameters can then be used in this formula to decide whether to accept or not the incoming calls. Erroneous parameter estimation may cause the CAC module to accept more

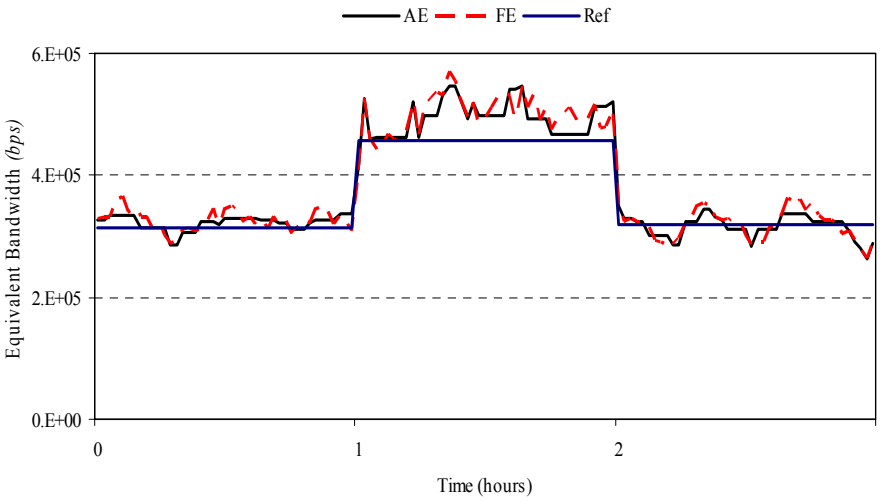

Figure 9.1. Equivalent bandwidth estimation using the FE and the AE algorithms. The proposed adaptive algorithm provides results closer to the reference curve respect to the fixed approach.

Table 7.1. Mean relative error in equivalent bandwidth estimation for the FE and AE approaches.

	Mean relative error
FE	7.30E-02
AE	5.78E-02

or less calls than the network can really handle. Next we show that the AE approach allows the bandwidth allocation module to predict the equivalent bandwidth more accurately than FE. To perform this test, we have again generated three one hour long traffic traces with uniform multiscaling features, as shown in Table 6.1. Enforcing a target blocking probability of 4.6E-14 and using (21.1), we have computed the equivalent bandwidth for the generated traces on the basis of the actual $\hat{\alpha}_i(q)$ and $\hat{c}_i(q)$ values for each hour. As before, the resulting equivalent bandwidth curve represents the reference curve. We have then computed the equivalent bandwidth when using the FE and AE algorithms to estimate the multifractality. Figure 9.1 shows the results. It can be noted that the AE algorithm allows for a higher accuracy in estimating the equivalent bandwidth. Indeed, the FE curve is characterized by a series of spikes that may produce undesired results in the subsequent CAC decision. Table 7.1 also shows the average relative error of the FE and AE estimates respect to the reference values. Note that the proposed algorithm presents an error reduction of about 20% with respect to the FE approach.

3.1.8 Conclusions

We have presented a new algorithm for real-time estimation of multifractal parameters for network traffic. The main feature consists in the use of an adaptive strategy, which allows for an adjustment of the estimation interval over time based on network traffic changes. Experimental results proved that compared to a fixed approach the proposed algorithm provides higher estimation accuracy. This algorithm may introduce significant benefits within a measurement-based resource allocation procedure that uses a multifractal model for the network traffic. In this work we have shown the advantage in case of a CAC procedure. It will be interesting to evaluate the advantages when using this algorithm in other procedures, such as wavelength assignments, route optimization, traffic policing and shaping [XTF+02]. This study will be the object of future work together with the study of the causes of multifractality changes over time.

successful because it enables people to connect with each other regardless of location. New technologies targeted at computer networks promise to do the same for Internet connectivity. The most successful wireless networking technology this far has been IEEE 802.11 [IEEE97].

Wireless networks offer several advantages over fixed (or "wired") networks:

Mobility

Users move, but data is usually stored centrally. Enabling users to access data while they are in motion can lead to large productivity gains.

Ease and speed of deployment

Many areas are difficult to wire for traditional wired LANs. Older buildings are often a problem; running cable through the walls of an older stone building to which the blueprints have been lost can be a challenge. In many places, historic preservation laws make it difficult to carry out new LAN installations in older buildings. Even in modern facilities, contracting for cable installation can be expensive and time-consuming.

Flexibility

No cables means no recabling. Wireless networks allow users to quickly form amorphous, small group networks for a meeting, and wireless networking makes moving between cubicles and offices a snap. Expansion with wireless networks is easy because the network medium is already everywhere. There are no cables to pull, connect, or trip over. Flexibility is the big selling point for the "hot spot" market, composed mainly of hotels, airports, train stations, libraries, and cafes.

Table 1.2. Comparison of 802.11 standards.

IEEE Standard	Speed	Frequency Band	Notes
802.11	1 Mbps – 2 Mbps	2.4 Ghz	First standard (1997). Featured both frequencyhopping and direct-sequence modulation techniques.
802.11a	Up to 54 Mbps	5 Ghz	Second standard (1999), but products not released until late 2000.
802.11b	5.5 Mbps – 11 Mbps	2.4 Ghz	Third standard, but second wave of products.
802.11g	Up to 54 Mbps	2.4 Ghz	Not yet standardized

Cost

In some cases, costs can be reduced by using wireless technology. As an example, 802.11-equipment can be used to create a wireless bridge between two buildings. Setting up a wireless bridge requires some initial capital cost in terms of outdoor equipment, access points, and wireless interfaces. After the initial capital expenditure, however, an 802.11-based, line-of-sight network will have only a negligible recurring monthly operating cost. Over time, point-to-point wireless links are far cheaper than leasing capacity from the telephone company. Until the completion of the 802.11 standard in 1997, however, users wanting to take advantage of these attributes were forced to adopt single-vendor solutions with all of the risk that entailed. Once 802.11 started the ball rolling, speeds quickly increased from 2 Mbps to 11 Mbps to 54 Mbps. Standardized wireless interfaces and antennas have made it possible to build wireless networks. Several service providers have jumped at the idea, and enthusiastic bands of volunteers in most major cities have started to build public wireless networks based on 802.11.

802.11 networks consist of four major physical components. The components are:

Distribution system

When several access points are connected to form a large coverage area, they must communicate with each other to track the movements of mobile stations. The distribution system is the logical component of 802.11 used to forward frames to their destination. 802.11 does not specify any particular technology for the distribution system. In most commercial products, the distribution system is implemented as a combination of a bridging engine and a distribution system medium, which is the backbone network used to relay frames between access points; it is often called simply the backbone network. In nearly all commercially successful products, Ethernet is used as the backbone network technology.

Access Points

Frames on an 802.11 network must be converted to another type of frame for delivery to the rest of the world. Devices called access points perform the wireless-to-wired bridging function. (Access Points perform a number of other functions, but bridging is by far the most important.)

Wireless medium

To move frames from station to station, the standard uses a wireless medium. Several different physical layers are defined; the architecture allows multiple physical layers to be developed to support the 802.11 MAC. Initially, two radio frequency (RF) physical layers and one infrared physical layer were standardized, though the RF layers have proven far more popular.

Stations

Networks are built to transfer data between stations. Stations are computing devices with wireless network interfaces. Typically, stations are battery-operated laptop or handheld computers. There is no reason why stations must be portable computing devices, though. In some environments, wireless networking is used to avoid pulling new cable, and desktops are connected by wireless LANs.

2.2 Types of Networks

The basic building block of an 802.11 network is the *basic service set* (BSS), which is simply a group of stations that communicate with each other. Communications take place within a somewhat fuzzy area, called the *basic service area*, defined by the propagation characteristics of the wireless medium. When a station is in the basic service area, it can communicate with the other members of the BSS. BSSs come in two flavors, both of which are illustrated in Figure. 1.2.

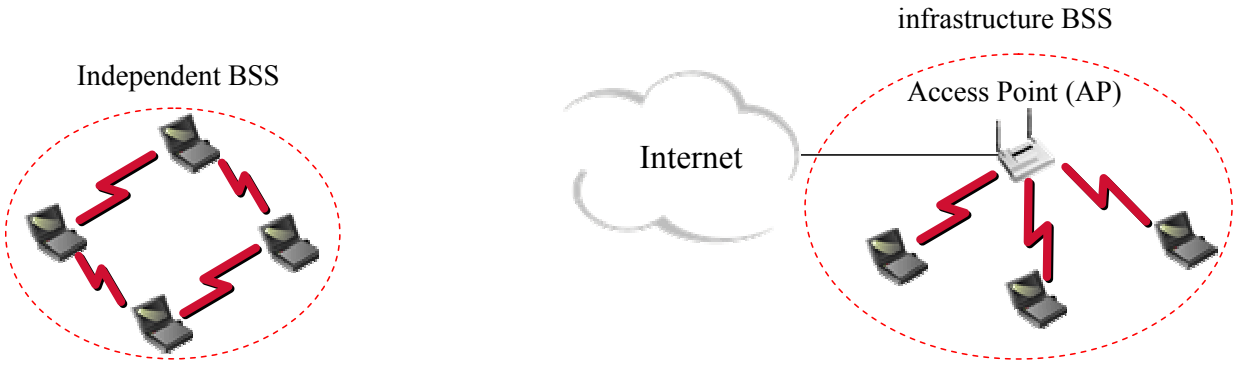

Figure 1.2. Types of Networks.

2.2.1 Independent Networks

On the left is an *independent BSS* (IBSS). Stations in an IBSS communicate directly with each other and thus must be within direct communication range. The smallest possible 802.11 network is an IBSS with two stations. Typically, IBSSs are composed of a small number of stations set up for a specific purpose and for a short period of time. One common use is to create a short-lived network to support a single meeting in a conference room. As the meeting begins, the participants create an IBSS to share data. When the meeting ends, the IBSS is dissolved. Due to their short duration, small size, and focused purpose, IBSSs are sometimes referred to as *ad hoc BSSs* or *ad hoc networks*.

2.2.2 Infrastructure Networks

On the right side of Figure 1.2 is an *infrastructure BSS* (never called an IBSS). Infrastructure networks are distinguished by the use of an access point. Access Points are used for all communications in infrastructure networks, including communication between mobile nodes in the same service area. If one mobile station in an infrastructure BSS needs to communicate with a second mobile station, the communication must take two hops. First, the originating mobile station transfers the frame to the access point. Second, the access point transfers the frame to the destination station. With all communications relayed through an access point, the basic service area corresponding to an infrastructure BSS is defined by the points in which transmissions from the access point can be received. Although the multihop transmission takes more transmission capacity than a directed frame from the sender to the receiver, it has two major advantages:

- An infrastructure BSS is defined by the distance from the access point. All mobile stations are required to be within reach of the access point, but no restriction is placed on the distance between mobile stations themselves. Allowing direct communication between mobile stations would save transmission capacity but at the cost of increased physical layer complexity because mobile stations would need to maintain neighbor relationships with all other mobile stations within the service area.
- Access Points in infrastructure networks are in a position to assist with stations attempting to save power. Access Points can note when a station enters a power saving mode and buffer

frames for it. Battery-operated stations can turn the wireless transceiver off and power it up only to transmit and retrieve buffered frames from the access point.

In an infrastructure network, stations must *associate* with an access point to obtain network services. Association is the process by which mobile station joins an 802.11 network; it is logically equivalent to plugging in the network cable on an Ethernet. It is not a symmetric process. Mobile stations always initiate the association process, and access points may choose to grant or deny access based on the contents of an association request. Associations are also exclusive on the part of the mobile station: a mobile station can be associated with only one access point. The 802.11 standard places no limit on the number of mobile stations that an access point may serve. Implementation considerations may, of course, limit the number of mobile stations an access point may serve. In practice, however, the relatively low throughput of wireless networks is far more likely to limit the number of stations placed on a wireless network.

2.2.3 Extended Service Areas

BSSs can create coverage in small offices and homes, but they cannot provide network coverage to larger areas. 802.11 allows wireless networks of arbitrarily large size to be created by linking BSSs into an *extended service set* (ESS). An ESS is created by chaining BSSs together with a backbone network. 802.11 does not specify a particular backbone technology; it requires only that the backbone provide a specified set of services (see Figure 2.2). Stations within the same ESS may communicate with each other, even though these stations may be in different basic service areas and may even be moving

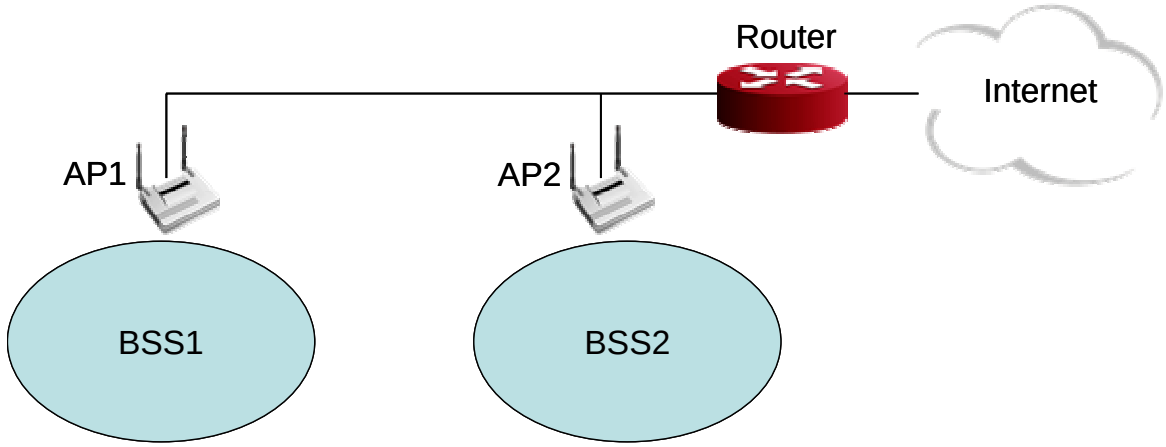

Figure 2.2. Extended Service Set.

between basic service areas. For stations in an ESS to communicate with each other, the wireless medium must act like a single layer 2 connection. Access Points act as bridges, so direct communication between stations in an ESS requires that the backbone network also be a layer 2 connection. Any link-layer connection will suffice. Several access points in a single area may be connected to a single hub or switch, or they can use virtual LANs if the link-layer connection must span a large area. Extended service areas are the highest-level abstraction supported by 802.11 networks. Access Points in an ESS operate in concert to allow the outside world to use a single MAC address to talk to a station somewhere within the ESS.

3.2 Network Services

One way to define a network technology is to define the services it offers and allow equipment vendors to implement those services in whatever way they see fit. 802.11 provides nine services. Only three of the services are used for moving data; the remaining six are management operations that allow the network to keep track of the mobile nodes and deliver frames accordingly.

The services are described in the following list:

Distribution

This service is used by mobile stations in an infrastructure network every time they send data. Once a frame has been accepted by an access point, it uses the distribution service to deliver the frame to its destination. Any communication that uses an access point travels through the distribution service, including communications between two mobile stations associated with the same access point.

Integration

Integration is a service provided by the distribution system; it allows the connection of the distribution system to a non-IEEE 802.11 network. The integration function is specific to the distribution system used and therefore is not specified by 802.11, except in terms of the services it must offer.

Association

Delivery of frames to mobile stations is made possible because mobile stations register, or associate, with access points. The distribution system can then use the registration information to determine which access point to use for any mobile station. Unassociated stations are not "on the network," much like workstations with unplugged Ethernet cables. 802.11 specifies the function that must be provided by the distribution system using the association data, but it does not mandate any particular implementation.

Reassociation

When a mobile station moves between basic service areas within a single extended service area, it must evaluate signal strength and perhaps switch the access point with which it is associated. Reassociations are initiated by mobile stations when signal conditions indicate that a different association would be beneficial; they are never initiated by the access point. After the reassociation is complete, the distribution system updates its location records to reflect the reachability of the mobile station through a different access point.

Disassociation

To terminate an existing association, stations may use the disassociation service. When stations invoke the disassociation service, any mobility data stored in the distribution system is removed. Once disassociation is complete, it is as if the station is no longer attached to the network. Disassociation is a polite task to do during the station shutdown process. The MAC is, however, designed to accommodate stations that leave the network without formally disassociating.

Authentication

Physical security is a major component of a wired LAN security solution. Network attachment points are limited, often to areas in offices behind perimeter access control devices. Network equipment can be secured in locked wiring closets, and data jacks in offices and cubicles can be connected to the network only when needed. Wireless networks cannot offer the same level of physical security, however, and therefore must depend on additional authentication routines to ensure that users accessing the network are authorized to do so. Authentication is a necessary prerequisite to association because only authenticated users are authorized to use the network. (In practice, though, many access points are configured for "open-system" authentication and will authenticate any station.)

Deauthentication

Deauthentication terminates an authenticated relationship. Because authentication is needed before network use is authorized, a side effect of deauthentication is termination of any current association.

Privacy

Strong physical controls can prevent a great number of attacks on the privacy of data in a wired LAN. Attackers must obtain physical access to the network medium before attempting to eavesdrop on traffic. On a wired network, physical access to the network cabling is a subset of physical access to other computing resources. By design, physical access to wireless networks is a comparatively simpler matter of using the correct antenna and modulation methods. To offer a similar level of privacy, 802.11 provides an optional privacy service called Wired Equivalent Privacy (WEP). WEP is not ironclad

security— in fact, it has been proven recently that breaking WEP is easily within the capabilities of any laptop. Its purpose is to provide roughly equivalent privacy to a wired network by encrypting frames as they travel across the 802.11 air interface. Depending on your level of cynicism, you may or may not think that WEP achieves its goal; after all, it's not that hard to access the Ethernet cabling in a traditional network. In any case, do not assume that WEP provides more than minimal security. It prevents other users from casually appearing on your network, but that's about all.

MSDU delivery

Networks are not much use without the ability to get the data to the recipient. Stations provide the MAC Service Data Unit (MSDU) delivery service, which is responsible for getting the data to the actual endpoint.

4.2 MAC Layer

The IEEE 802.11 standard successfully adapts Ethernet-style networking to radio links. Like Ethernet, 802.11 uses a carrier sense multiple access (CSMA) scheme to control access to the transmission medium. However, collisions waste valuable transmission capacity, so rather than the collision detection (CSMA/CD) employed by Ethernet, 802.11 uses collision avoidance (CSMA/CA). Also like Ethernet, 802.11 uses a distributed access scheme with no centralized controller. Each 802.11 station uses the same method to gain access to the medium. The major differences between 802.11 and Ethernet stem from the differences in the underlying medium.

Access to the wireless medium is controlled by coordination functions. Ethernet-like CSMA/CA access is provided by the distributed coordination function (DCF). If contention-free service is required, it can be provided by the point coordination function (PCF), which is built on top of the DCF. Contention-free services are provided only in infrastructure networks. The coordination functions are described in the following list:

DCF

The DCF is the basis of the standard CSMA/CA access mechanism. Like Ethernet, it first checks to see that the radio link is clear before transmitting. To avoid collisions, stations use a random backoff after each frame, with the first transmitter seizing the channel.

PCF

Point coordination provides contention-free services. Special stations called *point coordinators* are used to ensure that the medium is provided without contention. Point coordinators reside in access

points, so the PCF is restricted to infrastructure networks. To gain priority over standard contention-based services, the PCF allows stations to transmit frames after a shorter interval.

4.2.1 Carrier Sensing Functions

Carrier sensing is used to determine if the medium is available. Two types of carrier sensing functions in 802.11 manage this process: the physical carrier-sensing and virtual carrier-sensing functions. If either carrier-sensing function indicates that the medium is busy, the MAC reports this to higher layers. Physical carrier-sensing functions are provided by the physical layer in question and depend on the medium and modulation used. It is difficult (or, more to the point, expensive) to build physical carrier-sensing hardware for RF-based media, because transceivers can transmit and receive simultaneously only if they incorporate expensive electronics. Virtual carrier-sensing is provided by the Network Allocation Vector (NAV). Most 802.11 frames carry a duration field, which can be used to reserve the medium for a fixed time period. The NAV is a timer that indicates the amount of time the medium will be reserved. Stations set the NAV to the time for which they expect to use the medium, including any frames necessary to complete the current operation. Other stations count down from the NAV to 0. When the NAV is nonzero, the virtual carrier-sensing function indicates that the medium is busy; when the NAV reaches 0, the virtual carrier-sensing function indicates that the medium is idle.

4.2.2 Interframe Spacing

As with traditional Ethernet, the interframe spacing plays a large role in coordinating access to the transmission medium. 802.11 uses four different interframe spaces.

We've already seen that as part of the collision avoidance built into the 802.11 MAC, stations delay transmission until the medium becomes idle. Varying interframe spacings create different priority levels for different types of traffic. The logic behind this is simple: high-priority traffic doesn't have to wait as long after the medium has become idle. Therefore, if there is any high-priority traffic waiting, it grabs the network before low-priority frames have a chance to try. To assist with interoperability between different data rates, the interframe space is a fixed amount of time, independent of the transmission speed. (This is only one of the many problems caused by having different physical layers use the same radio resources, which are different modulation techniques.) Different physical layers, however, can specify different interframe space times.

Short interframe space (SIFS)

The SIFS is used for the highest-priority transmissions, such as positive acknowledgments. High-priority transmissions can begin once the SIFS has elapsed. Once these high-priority transmissions begin, the medium becomes busy, so frames transmitted after the SIFS has elapsed have priority over frames that can be transmitted only after longer intervals.

PCF interframe space (PIFS)

The PIFS, sometimes erroneously called the priority interframe space, is used by the PCF during contention-free operation. Stations with data to transmit in the contention-free period can transmit after the PIFS has elapsed and preempt any contention-based traffic.

DCF interframe space (DIFS)

The DIFS is the minimum medium idle time for contention-based services. Stations may have immediate access to the medium if it has been free for a period longer than the DIFS.

Extended interframe space (EIFS)

It is not a fixed interval. It is used only when there is an error in frame transmission.

4.2.3 The Hidden Node Problem

In Ethernet networks, stations depend on the reception of transmissions to perform the carrier sensing functions of CSMA/CD. Wires in the physical medium contain the signals and distribute them to network nodes. Wireless networks have fuzzier boundaries, sometimes to the point where each node may not be able to communicate with every other node in the wireless network, as in Figure 3.2.

In the figure, node 2 can communicate with both nodes 1 and 3, but something prevents nodes 1 and 3 from communicating directly. (The obstacle itself is not relevant; it could be as simple as nodes 1 and 3 being as far away from 2 as possible, so the radio waves cannot reach the full distance from 1 to 3.)

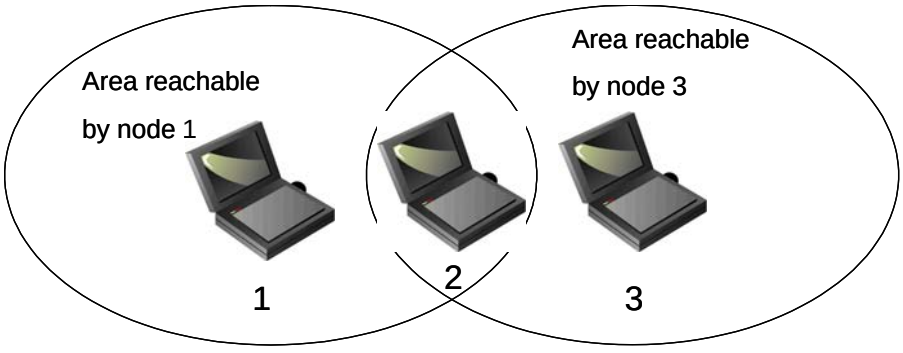

Figure 3.2. Nodes 1 and 3 are hidden.

From the perspective of node 1, node 3 is a "hidden" node. If a simple transmit-and-pray protocol was used, it would be easy for node 1 and node 3 to transmit simultaneously, thus rendering node 2 unable to make sense of anything. Furthermore, nodes 1 and 3 would not have any indication of the error because the collision was local to node 2.

Collisions resulting from hidden nodes may be hard to detect in wireless networks because wireless transceivers are generally half-duplex; they do not transmit and receive at the same time. To prevent collisions, 802.11 allows stations to use Request to Send (RTS) and Clear to Send (CTS) signals to clear out an area. Figure 4.2 illustrates the procedure.

In Figure 4.2, node 1 has a frame to send; it initiates the process by sending an RTS frame. The RTS frame serves several purposes: in addition to reserving the radio link for transmission, it silences any stations that hear it. If the target station receives an RTS, it responds with a CTS. Like the RTS frame, the CTS frame silences stations in the immediate vicinity. Once the RTS/CTS exchange is complete, node 1 can transmit its frames without worry of interference from any hidden nodes. Hidden nodes beyond the range of the sending station are silenced by the CTS from the receiver. When the RTS/CTS clearing procedure is used, any frames must be positively acknowledged. The multiframe RTS/CTS transmission procedure consumes a fair amount of capacity, especially because of the additional latency incurred before transmission can commence. As a result, it is used only in high-capacity environments and environments with significant contention on transmission. For lower-capacity environments, it is not necessary.

It is possible to control the RTS/CTS procedure by setting the *RTS threshold* if the device driver for the

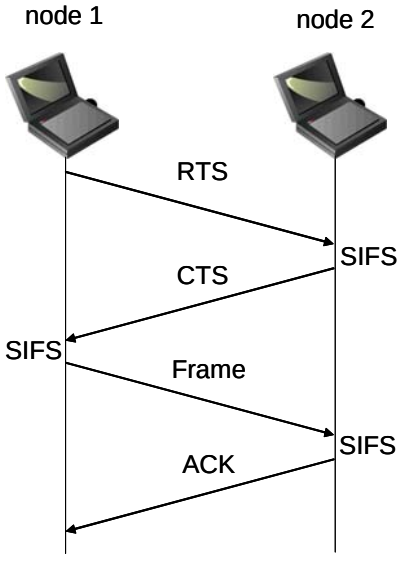

Figure 4.2. RTS/CTS clearing

802.11 card allows to adjust it. The RTS/CTS exchange is performed for frames larger than the threshold. Frames shorter than the threshold are simply sent.

4.2.4 Contention-Based Access Using the DCF

Most traffic uses the DCF, which provides a standard Ethernet-like contention-based service. The DCF allows multiple independent stations to interact without central control, and thus may be used in either IBSS networks or in infrastructure networks. Before attempting to transmit, each station checks whether the medium is idle. If the medium is not idle, stations defer to each other and employ an orderly exponential backoff algorithm to avoid collisions.

In distilling the 802.11 MAC rules, there is a basic set of rules that are always used, and additional rules may be applied depending on the circumstances. Two basic rules apply to all transmissions using the DCF:

1. If the medium has been idle for longer than the DIFS, transmission can begin immediately. Carrier sensing is performed using both a physical medium dependent method and the virtual (NAV) method.
    a. If the previous frame was received without errors, the medium must be free for at least the DIFS.
    b. If the previous transmission contained errors, the medium must be free for the amount of the EIFS.
2. If the medium is busy, the station must wait for the channel to become idle. 802.11 refers to the wait as *access deferral*. If access is deferred, the station waits for the medium to become idle for the DIFS and prepares for the exponential backoff procedure.

Additional rules may apply in certain situations. Many of these rules depend on the particular situation "on the wire" and are specific to the results of previous transmissions.

1. Error recovery is the responsibility of the station sending a frame. Senders expect acknowledgments for each transmitted frame and are responsible for retrying the transmission until it is successful.
    a. Positive acknowledgments are the only indication of success. If an acknowledgment is expected and does not arrive, the sender considers the transmission lost and must retry.
    b. All unicast data must be acknowledged.
    c. Any failure increments a retry counter, and the transmission is retried. A failure can be due to a failure to gain access to the medium or a lack of an acknowledgment. However, there is a longer congestion window when transmissions are retried.

2. Multiframe sequences may update the NAV with each step in the transmission procedure. When a station receives a medium reservation that is longer than the current NAV, it updates the NAV. Setting the NAV is done on a frame-by-frame basis and is discussed in much more detail in the next chapter.

3. The following types of frames can be transmitted after the SIFS and thus receive maximum priority: acknowledgments, the CTS in an RTS/CTS exchange and fragments in fragment sequences.

   a. Once a station has transmitted the first frame in a sequence, it has gained control of the channel. Any additional frames and their acknowledgments can be sent using the short interframe space, which locks out any other stations.

   b. Additional frames in the sequence update the NAV for the expected additional time the medium will be used.

4. Extended frame sequences are required for higher-level packets that are larger than configured thresholds.

   a. Packets larger than the RTS threshold must have RTS/CTS exchange.

   b. Packets larger than the fragmentation threshold must be fragmented.

4.2.5 Error Recovery with the DCF

Error detection and correction is up to the station that begins an atomic frame exchange. When an error is detected, the station with data must resend the frame. Errors must be detected by the sending station. In some cases, the sender can infer frame loss by the lack of a positive acknowledgment from the receiver. Retry counters are incremented when frames are retransmitted. Each frame or fragment has a single retry counter associated with it. Stations have two retry counters: the *short retry count* and the *long retry count*. Frames that are shorter than the RTS threshold are considered to be short; frames longer than the threshold are long. Depending on the length of the frame, it is associated with either a short or long retry counter. Frame retry counts begin at 0 and are incremented when a frame transmission fails.

The short retry count is reset to 0 when:

- A CTS frame is received in response to a transmitted RTS
- A MAC-layer acknowledgment is received after a nonfragmented transmission
- A broadcast or multicast frame is received

The long retry count is reset to 0 when:

- A MAC-layer acknowledgment is received for a frame longer than the RTS threshold
- A broadcast or multicast frame is received

In addition to the associated retry count, fragments are given a maximum "lifetime" by the MAC. When the first fragment is transmitted, the lifetime counter is started. When the lifetime limit is reached, the frame is discarded and no attempt is made to transmit any remaining fragments.

4.2.6 Using the retry counters

Like most other network protocols, 802.11 provides reliability through retransmission. Data transmission happens within the confines of an atomic sequence, and the entire sequence must complete for a transmission to be successful. When a station transmits a frame, it must receive an acknowledgment from the receiver or it will consider the transmission to have failed. Failed transmissions increment the retry counter associated with the frame (or fragment). If the retry limit is reached, the frame is discarded, and its loss is reported to higher-layer protocols. One of the reasons for having short frames and long frames is to allow network administrators to customize the robustness of the network for different frame lengths. Large frames require more buffer space, so one potential application of having two separate retry limits is to decrease the long retry limit to decrease the amount of buffer space required.

4.2.7 Backoff with the DCF

After frame transmission has completed and the DIFS has elapsed, stations may attempt to transmit congestion-based data. A period called the *Contention Window* or *backoff window* follows the DIFS. This window is divided into slots. Slot length (T_{slot}) is medium dependent; higher-speed physical layers use shorter slot times. Stations pick a random slot and wait for that slot before attempting to access the medium; all slots are equally likely selections. When several stations are attempting to transmit, the station that picks the first slot (the station with the lowest random number) wins.
With the DCF access method, an active station with frames to be transmitted senses the channel. If the channel is sensed idle for a period greater than the distributed interframe space (DIFS), the station transmits its frames; otherwise defers its transmission till the end of the ongoing transmission. Then, the station randomly selects a *backoff interval* used to initialize the *backoff timer*; this timer is decremented of a *backoff slot* as long as the channel is sensed idle, stopped when the channel is busy, and restarted when the channel becomes idle after a DIFS again. When the backoff timer expires, the station is allowed to transmit. As in Ethernet, the backoff time is selected from a larger range each time a transmission fails. Contention window sizes are always 1 less than a power of 2 (e.g., 31, 63, 127,

255). Each time the retry counter increases, the contention window moves to the next greatest power of two. The size of the contention window is limited by the physical layer. For example, the DS physical layer limits the contention window to 1023 transmission slots. For simplicity of notation in the following, we consider contention windows with size of power of 2.

When a station starts the collision avoidance mechanism, randomly selects the backoff counter (hereafter BC) in the range $[0, CW(j)]$, where $CW(j)$ is the contention window after j unsuccessful transmission attempts. For the first attempt the contention window is set to the minimum value $CW_{\min}$; this value is doubled for each collision till the maximum value $CW_{\max} = 2^{w} CW_{\min}$ is reached. Allowing long contention windows when several competing stations are attempting to gain access to the medium keeps the MAC algorithms stable even under maximum load. The contention window is reset to its minimum size when frames are transmitted successfully, or the associated retry counter is reached, and the frame is discarded. The entire BC has to expire before a station transmits. If another station uses the channel before the BC of the tagged station expires, the counter is stopped and restarted at the end of the next DIFS interval. The station is able to transmit after the residual value of the BC is decremented to zero. Then the frames exchange between the two stations finally begins: the IEEE 802.11 standard states that when a station receives a frame with no error, has to send an ACK frame after a SIFS.

4.2.8 Frame Format

To meet the challenges posed by a wireless data link, the MAC was forced to adopt several unique features, not the least of which was the use of four address fields. Not all frames use all the address fields, and the values assigned to the address fields may change depending on the type of MAC frame being transmitted. Figure 5.2 shows the generic 802.11 MAC frame. All diagrams in this section follow the IEEE conventions in 802.11. Fields are transmitted from left to right, and the most significant bits appear last. 802.11 MAC frames do not include some of the classic Ethernet frame features, most notably the type/length field and the preamble. The preamble is part of the physical layer, and encapsulation details such as type and length are present in the header on the data carried in the 802.11 frame.

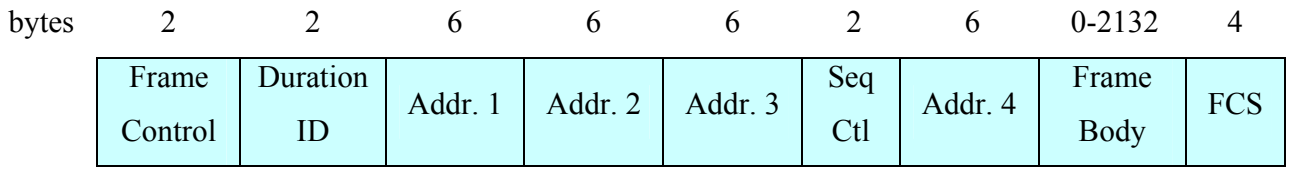

Figure 5.2. Generic 802.11 MAC Frame.

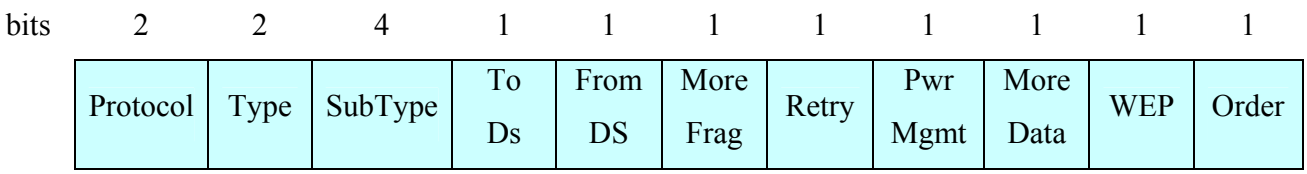

Figure 6.2. Frame Control Field.

Table 2.2. Type and subtype identifiers.

Subtype Value	**Subtype Name**
Management Frames (Type 00)	
0000	Association request
0001	Association response
0010	Reassociation request
0011	Reassociation response
0100	Probe request
0101	Probe response
1000	Beacon
1001	Announcement traffic indication message (ATIM)
1010	Disassociation
1011	Authentication
1100	Deauthentication
Control Frames (Type 01)	
1010	Power Save (PS)-Poll
1011	RTS
1100	CTS
1101	Acknowledgment (ACK)
1110	Contention Free (CF)-End
1111	CF-End+CF-Ack
Data Frames (Type 10)	
0000	Data
0001	Data+CF-Ack
0010	Data+CF-Poll
0011	Data+CF-Ack+CF-Poll

Frame Control

Each frame starts with a two-byte Frame Control subfield, shown in Figure 5.2. The components of the Frame Control subfield as shown in Figure 6.2, are:

Protocol version

Two bits indicate which version of the 802.11 MAC is contained in the rest of the frame. At present, only one version of the 802.11 MAC has been developed; it is assigned the protocol number 0. Other values will appear when the IEEE standardizes changes to the MAC that render it incompatible with the initial specification.

Type and subtype fields

Type and subtype fields identify the type of frame used. To cope with noise and unreliability, a number of management functions are incorporated into the 802.11 MAC. Some, such as the RTS/CTS operations and the acknowledgments, have already been discussed.

In Table 2, bit strings are written most-significant bit first, which is the reverse of the order used in Figure 6.2. Therefore, the frame type is the third bit in the frame control field followed by the second bit (b3 b2), and the subtype is the seventh bit, followed by the sixth, fifth, and fourth bits (b7 b6 b5 b4).

ToDS and FromDS bit

These bits indicate whether a frame is destined for the distribution system. All frames on infrastructure networks will have one of the distribution system's bits set. Table 3 shows how these bits are interpreted. The interpretation of the address fields depends on the setting of these bits.

More fragments bit

This bit functions much like the "more fragments" bit in IP. When a higher-level packet has been fragmented by the MAC, the initial fragment and any following non final fragments set this bit to 1. Some management frames may be large enough to require fragmentation; all other frames set this bit to 0.

Table 3.2. Interpreting the ToDS and FromDS bits.

	To DS = 0	To DS = 1
From DS = 0	All management and control frames Data frames within an IBSS (never infrastructure data frames)	Data frames transmitted from a wireless station in an infrastructure network
From DS = 1	Data frames received for a wireless station in an infrastructure network	Data frames on a "wireless bridge"

Retry bit

From time to time, frames may be retransmitted. Any retransmitted frames set this bit to 1 to aid the receiving station in eliminating duplicate frames.

Power management bit

Network adapters built on 802.11 are often built to the PC Card form factor and used in battery-powered laptop or handheld computers. To conserve battery life, many small devices have the ability to power down parts of the network interface. This bit indicates whether the sender will be in a power-saving mode after the completion of the current atomic frame exchange. One indicates that the station will be in power-save mode, and 0 indicates that the station will be active. Access Points perform a number of important management functions and are not allowed to save power, so this bit is always 0 in frames transmitted by an access point.

More data bit

To accommodate stations in a power-saving mode, access points may buffer frames received from the distribution system. An access point sets this bit to indicate that at least one frame is available and is addressed to a dozing station.

WEP bit

Wireless transmissions are inherently easier to intercept than transmissions on a fixed network. 802.11 defines a set of encryption routines called Wired Equivalent Privacy (WEP) to protect and authenticate data. When a frame has been processed by WEP, this bit is set to 1, and the frame changes slightly.

Order bit

Frames and fragments can be transmitted in order at the cost of additional processing by both the sending and receiving MACs. When the "strict ordering" delivery is employed, this bit is set to 1.

Duration/ID Field

The Duration/ID field follows the frame control field. This field has several uses and takes one of the following three forms:

- When bit 15 is 0, the duration/ID field is used to set the NAV. The value represents the number of microseconds that the medium is expected to remain busy for the transmission currently in progress. All stations must monitor the headers of all frames they receive and update the NAV accordingly. Any value that extends the amount of time the medium is busy updates the NAV and blocks access to the medium for additional time.

- During the contention-free periods, bit 14 is 0 and bit 15 is 1. All other bits are 0, so the duration/ID field takes a value of 32,768. This value is interpreted as a NAV. It allows any stations that did not receive the Beacon announcing the contention-free period to update the NAV with a suitably large value to avoid interfering with contention-free transmissions.

- Bits 14 and 15 are both set to 0 in PS-Poll frames. Mobile stations may elect to save battery power by turning off antennas. Dozing stations must wake up periodically. To ensure that no frames are lost, stations awaking from their slumber transmit a PS-Poll frame to retrieve any buffered frames from the access point. Along with this request, waking stations incorporate the association ID (AID) that indicates which BSS they belong to. The AID is included in the PS-Poll frame and may range from 1-2,007. Values from 2,008-16,383 are reserved and not used.

Address Fields

An 802.11 frame may contain up to four address fields. The address fields are numbered because different fields are used for different purposes depending on the frame type. The general rule of thumb is that Address 1 is used for the receiver, Address 2 for the transmitter, with the Address 3 field used for filtering by the receiver.

Addressing in 802.11 follows the conventions used for the other IEEE 802 networks, including Ethernet. Addresses are 48 bits long. If the first bit sent to the physical medium is a 0, the address represents a single station (unicast). When the first bit is a 1, the address represents a group of physical stations and is called a *multicast* address. If all bits are 1s, then the frame is a *broadcast* and is delivered to all stations connected to the wireless medium.

48-bit addresses are used for a variety of purposes:

Destination address

As in Ethernet, the destination address is the 48-bit IEEE MAC identifier that corresponds to the final recipient: the station that will hand the frame to higher protocol layers for processing.

Source address

This is the 48-bit IEEE MAC identifier that identifies the source of the transmission. Only one station can be the source of a frame, so the Individual/Group bit is always 0 to indicate an individual station.

Receiver address

This is a 48-bit IEEE MAC identifier that indicates which wireless station should process the frame. If it is a wireless station, the receiver address is the destination address. For frames destined to a node on an Ethernet connected to an access point, the receiver is the wireless interface in the access point, and the destination address may be a router attached to the Ethernet.

Transmitter address

This is a 48-bit IEEE MAC address to identify the wireless interface that transmitted the frame onto the wireless medium. The transmitter address is used only in wireless bridging.

Basic Service Set ID (BSSID)

To identify different wireless LANs in the same area, stations may be assigned to a BSS. In infrastructure networks, the BSSID is the MAC address used by the wireless interface in the access point. Ad hoc networks generate a random BSSID with the Universal/Local bit set to 1 to prevent conflicts with officially assigned MAC addresses.

The number of address fields used depends on the type of frame. Most data frames use three fields for source, destination, and BSSID. The number and arrangement of address fields in a data frame depends on how the frame is travelling relative to the distribution system. Most transmissions use three addresses, which is why only three of the four addresses are contiguous in the frame format.

Sequence Control Field

This 16-bit field is used for both defragmentation and discarding duplicate frames. It is composed of a 4-bit fragment number field and a 12-bit sequence number field. Higher-level frames are each given a sequence number as they are passed to the MAC for transmission. The sequence number subfield operates as a modulo-4096 counter of the frames transmitted. It begins at 0 and increments by 1 for each higher-level packet handled by the MAC. If higher-level packets are fragmented, all fragments will have the same sequence number. When frames are retransmitted, the sequence number is not changed.

What differs between fragments is the fragment number. The first fragment is given a fragment number of 0. Each successive fragment increments the fragment number by one. Retransmitted fragments keep their original sequence numbers to assist in reassembly.

Frame Body

The frame body, also called the Data field, moves the higher-layer payload from station to station. 802.11 can transmit frames with a maximum payload of 2,304 bytes of higher-level data. (Implementations must support frame bodies of 2,312 bytes to accommodate WEP overhead.) 802.2 LLC headers use 8 bytes for a maximum network protocol payload of 2,296 bytes. Preventing fragmentation must be done at the protocol layer. On IP networks, Path MTU Discovery (RFC 1191) will prevent the transmission of frames with Data fields larger than 1,500 bytes.

Frame Check Sequence

As with Ethernet, the 802.11 frame closes with a frame check sequence (FCS). The FCS is often referred to as the cyclic redundancy check (CRC) because of the underlying mathematical operations. The FCS allows stations to check the integrity of received frames. All fields in the MAC header and the body of the frame are included in the FCS. Although 802.3 and 802.11 use the same method to calculate the FCS, the MAC header used in 802.11 is different from the header used in 802.3, so the FCS must be recalculated by access points.

When frames are sent to the wireless interface, the FCS is calculated before those frames are sent out over the RF or IR link. Receivers can then calculate the FCS from the received frame and compare it to the received FCS. If the two match, there is a high probability that the frame was not damaged in transit. On Ethernets, frames with a bad FCS are simply discarded, and frames with a good FCS are passed up the protocol stack. On 802.11 networks, frames that pass the integrity check may also require the receiver to send an acknowledgment. For example, data frames that are received correctly must be positively acknowledged, or they are retransmitted. 802.11 does not have a negative acknowledgment for frames that fail the FCS; stations must wait for the acknowledgment timeout before retransmitting.

Power-Save Poll (PS-Poll)

When a mobile station wakes from a power-saving mode, it transmits a PS-Poll frame to the access point to retrieve any frames buffered while it was in power-saving mode.

Four fields make up the MAC header of a PS-Poll frame:

Frame Control

The frame subtype is set to 1010 to indicate a PS-Poll frame.

Association ID (AID)

Instead of a Duration field, the PS-Poll frame uses the third and fourth bytes in the MAC header for the association ID. This is a numeric value assigned by the access point to identify the association. Including this ID in the frame allows the access point to find any frames buffered for the now-awakened mobile station.

When mobile stations associate with an access point, the access point assigns the AID from the range 1-2,007.

Address 1: BSSID

This field contains the BSSID of the BSS created by the access point that the sender is currently associated with.

Address 2: Transmitter Address

This is the address of the sender of the PS-Poll frame. The PS-Poll frame does not include duration information to update the NAV. However, all stations receiving a PS-Poll frame update the NAV by the short interframe space plus the amount of time required to transmit an ACK. The automatic NAV update allows the access point to transmit an ACK with a small probability of collision with a mobile station.

5.2 Power Conservation

The major advantage of wireless networks is that network access does not require nodes to be in any particular location. To take full advantage of mobility, nothing can constrain the location of a node, including the availability of electrical power. Mobility therefore implies that most mobile devices can run on batteries. But battery power is a scarce resource; batteries can run only so long before they need to be recharged. Requiring mobile users to return frequently to commercial power is inconvenient, to say the least. Many wireless applications require long battery life without sacrificing network connectivity.

As with any other network interface, powering down the transceiver can lead to great power savings in wireless networks. When the transceiver is off, it is said to be *sleeping*, *dozing*, or in *power-saving mode* (PS). When the transceiver is on, it is said to be *awake*, *active*, or simply *on*. Power conservation in 802.11 is achieved by minimizing the time spent in the latter stage and maximizing the time in the former. However, 802.11 accomplishes this without sacrificing connectivity.

5.2.1 Power Management in Infrastructure Networks

Power management can achieve the greatest savings in infrastructure networks. All traffic for mobile stations must go through access points, so they are an ideal location to buffer traffic. There is no need to work on a distributed buffer system that must be implemented on every station; the bulk of the work is left to the access point. By definition, access points are aware of the location of mobile stations, and a mobile station can communicate its power management state to its access point. Furthermore, access points must remain active at all times; it is assumed that they have access to continuous power. Combining these two facts allows access points to play a key role in power management on infrastructure networks.

Access points have two power management-related tasks. First, because an access point knows the power management state of every station that has associated with it, it can determine whether a frame should be delivered to the wireless network because the station is active or buffered because the station is asleep. But buffering frames alone does not enable mobile stations to pick up the data waiting for them. An access point's second task is to announce periodically which stations have frames waiting for them. The periodic announcement of buffer status also helps to contribute to the power savings in infrastructure networks. Powering up a receiver to listen to the buffer status requires far less power than

periodically transmitting polling frames. Stations only need to power up the transmitter to transmit polling frames after being informed that there is a reason to expend the energy. Power management is designed around the needs of the battery-powered mobile stations. Mobile stations can sleep for extended periods to avoid using the wireless network interface. Part of the association request is the Listen Interval parameter, which is the number of Beacon periods for which the mobile station may choose to sleep. Longer listen intervals require more buffer space on the access point; therefore, the Listen Interval is one of the key parameters used in estimating the resources required to support an association. The Listen Interval is a contract with the access point. In agreeing to buffer any frames while the mobile station is sleeping, the access point agrees to wait for at least the listen interval before discarding frames. If a mobile station fails to check for waiting frames after each listen interval, they may be discarded without notification.

Unicast frame buffering and delivery using the Traffic Indication Map (TIM)

When frames are buffered, the destination node's AID provides the logical link between the frame and its destination. Each AID is logically connected to frames buffered for the mobile station that is assigned that AID. Multicast and broadcast frames are buffered and linked to an AID of zero.
Buffering is only half the battle. If stations never pick up their buffered frames, saving the frames is a rather pointless exercise. To inform stations that frames are buffered, access points periodically assemble a traffic indication map (TIM) and transmit it in Beacon frames. The TIM is a virtual bitmap composed of 2,008 bits; offsets are used so that the access point needs to transmit only a small portion of the virtual bitmap. This conserves network capacity when only a few stations have buffered data. Each bit in the TIM corresponds to a particular AID; setting the bit indicates that the access point has buffered unicast frames for the station with the AID corresponding to the bit position. Mobile stations must wake up and enter the active mode to listen for Beacon frames to receive the TIM. By examining the TIM, a station can determine if the access point has buffered traffic on its behalf. To retrieve buffered frames, mobile stations use PS-Poll Control frames. When multiple stations have buffered frames, all stations with buffered data must use the random backoff algorithm before transmitting the PS-Poll. Each PS-Poll frame is used to retrieve one buffered frame. That frame must be positively acknowledged before it is removed from the buffer. Positive acknowledgment is required to keep a second, retried PS-Poll from acting as an implicit acknowledgment. Figure 7.2 illustrates the process.

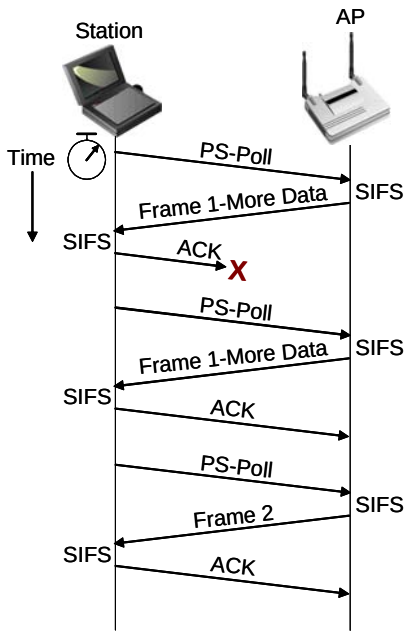

Figure 7.2. PS-Poll frame retrieval.

If multiple frames are buffered for a mobile station, then the More Data bit in the Frame Control field is set to 1. Mobile stations can then issue additional PS-Poll requests to the access point until the More Data bit is set to 0, though no time constraint is imposed by the standard. After transmitting the PS-Poll, a mobile station must remain awake until either the polling transaction has concluded or the bit corresponding to its AID is no longer set in the TIM. The reason for the first case is obvious: the mobile station has successfully polled the access point; part of that transaction was a notification that the mobile station will be returning to a sleeping mode. The second case allows the mobile station to return to a power conservation mode if the access point discards the buffered frame. Once all the traffic buffered for a station is delivered or discarded, the station can resume sleeping. The buffering and delivery process is illustrated in Figure 8.2, which shows the medium as it appears to an access point and two associated power-saving stations. The hash marks on the timeline represent the beacon interval. Every beacon interval, the access point transmits a Beacon frame with a TIM information element. The lines above the station base lines indicate the ramp-up process of the receiver to listen for the TIM.

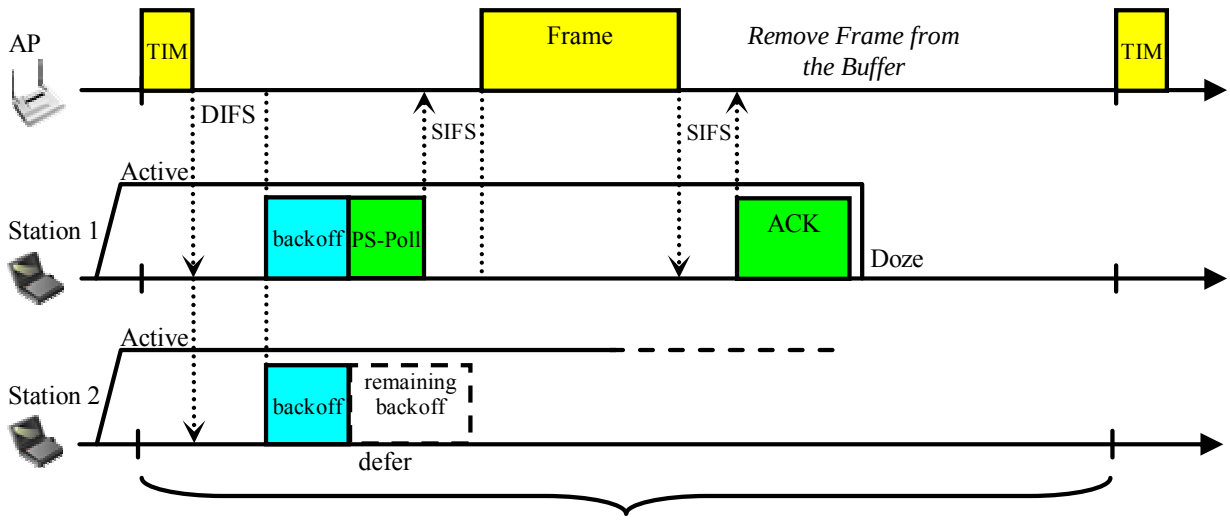

Figure 8.2. Buffered frame retrieval process.

At the beginning of the beacon, both stations wake up to listen to the TIM, which indicates that there are frames buffered for both. Both station 1 and station 2 prepare to transmit PS-Poll frames after the expiration of a contention window countdown as described above. Station 1 wins because its random delay was shorter. Station 1 issues a PS-Poll and receives its buffered frame in response. During the transmission, station 2 defers. If, at the end of that frame transmission, a third station, which is not illustrated, seizes the medium for transmission, station 2 must continue to stay awake until the next TIM. If the access point has run out of buffer space and has discarded the buffered frame for station 2, the TIM at the successive beacon indicates that no frames are buffered, and station 2 can finally return to a low-power mode. Stations may switch from a power conservation mode to active mode at any time. It is common for laptop computers to operate with full power to all peripherals when connected to AC power and conserve power only when using the battery. If a mobile station switches to the active mode from a sleeping mode, frames can be transmitted without waiting for a PS-Poll. PS-Poll frames indicate that a power-saving mobile station has temporarily switched to an active mode and is ready to receive a buffered frame. By definition, active stations have transceivers operating continuously. After a switch to active mode, the access point can assume that the receiver is operational, even without receiving explicit notification to that effect. Access points must retain frames long enough for mobile stations to pick them up, but buffer memory is a finite resource. 802.11 mandates that access points use an *aging function* to determine when buffered frames are old enough to be discarded. The standard leaves a great deal to the discretion of the developer because it specifies only one constraint. Mobile stations depend on access points to buffer traffic for at least the listen interval specified with the

association, and the standard forbids the aging function from discarding frames before the listen interval has elapsed. Beyond that, however, there is a great deal of latitude for vendors to develop different buffer management routines.

Delivering multicast and broadcast frames: the Delivery TIM (DTIM)

Frames with a group address cannot be delivered using a polling algorithm because they are, by definition, addressed to a group. Therefore, 802.11 incorporates a mechanism for buffering and delivering broadcast and multicast frames. Buffering is identical to the unicast case, except that frames are buffered whenever any station associated with the access point is sleeping. Buffered broadcast and multicast frames are saved using AID 0. Access points indicate whether any broadcast or multicast frames are buffered by setting the first bit in the TIM to 0; this bit corresponds to AID 0. Each BSS has a parameter called the DTIM Period. TIMs are transmitted with every Beacon. At a fixed number of Beacon intervals, a special type of TIM, a Delivery Traffic Indication Map (DTIM), is sent. The TIM element in Beacon frames contains a counter that counts down to the next DTIM; this counter is zero in a DTIM frame. Buffered broadcast and multicast traffic is transmitted after a DTIM Beacon. Multiple buffered frames are transmitted in sequence; the More Data bit in the Frame Control field indicates that more frames must be transmitted. Normal channel acquisition rules apply to the transmission of buffered frames. The access point may choose to defer the processing of incoming PS-Poll frames until the frames in the broadcast and multicast transmission buffers have been transmitted.

To receive broadcast and multicast frames, a mobile station must be awake for DTIM transmissions. Nothing in the specification, however, keeps power-saving stations in infrastructure networks from waking up to listen to DTIM frames. Some products that implement power-saving modes will attempt to align their awakenings with DTIM transmissions. If the system administrator determines that battery life is more important than receiving broadcast and multicast frames, a station can be configured to sleep for its listen period without regard to DTIM transmissions. Some documentation may refer to this as *extremely low power*, *ultra power-saving mode*, *deep sleep*, or something similar. Several products allow configuration of the DTIM interval. Lengthening the DTIM interval allows mobile stations to sleep for longer periods and maximizes battery life at the expense of timely delivery. Shorter DTIM intervals emphasize quick delivery at the expense of more frequent power-up and power-down cycles. You can use a longer DTIM when battery life is at a premium and delivery of broadcast and multicast

frames is not important. Whether this is appropriate depends on the applications you are using and how they react to long link-layer delays.

Power-Saving Sequences

The most power-hungry components in RF systems are the amplifiers used to boost a signal immediately prior to transmission and to boost the received signal to an intelligible level immediately after its reception. As already mentioned, 802.11 stations can maximize battery life by shutting down the radio transceiver and sleeping periodically. We have seen that during sleeping periods, access points buffer any unicast frames for sleeping stations. These frames are announced by subsequent Beacon frames. To retrieve buffered frames, newly awakened stations use PS-Poll frames.

Immediate response

Access points can respond immediately to the PS-Poll. After a short interframe space, an access point may transmit the frame. The PS-Poll frame contains an Association ID in the Duration/ID field so that the access point can determine which frames were buffered for the mobile station. However, the MAC specification requires all stations receiving a PS-Poll to update the NAV with an implied value equal to a short interframe space and one ACK. Although the NAV is too short for the data frame, the access point acquires that the medium and all stations defer access for the entire data frame. At the conclusion of the data frame, the NAV is updated to reflect the value in the header of the data frame.
If the buffered frame is large, it may require fragmentation. Like all other stations, access points typically have a configurable fragmentation threshold.

Deferred response

Instead of an immediate response, access points can also respond with a simple acknowledgment. This is called a *deferred response* because the access point acknowledges the request for the buffered frame but does not act on it immediately. A station requesting a frame with a PS-Poll must stay awake until it is delivered. Under contention-based service, however, the access point can deliver a frame at any point. A station cannot return to a low-power mode until it receives a Beacon frame in which its bit in the traffic indication map (TIM) is clear. However, the access point may choose to defer its response by transmitting only an ACK. At this point, the access point has acknowledged the station's request for

buffered frames and promised to deliver them at some point in the future. The station must wait in active mode, perhaps through several atomic frame exchanges, before the access point delivers the data. Furthermore, a buffered frame may be subject to fragmentation.
After receiving a data frame, the station must remain awake until the next Beacon is transmitted. Beacon frames only note whether frames are buffered for a station and have no way of indicating the number of frames. Once the station receives a Beacon frame indicating that no more traffic is buffered, it can conclude that it has received the last buffered frame and return to a low-power mode.

6.2 Timer Synchronization

Like other wireless network technologies, 802.11 depends a great deal on the distribution of timing information to all the nodes. It is especially important in frequency-hopping networks because all stations on the network must change frequency channels in a coordinated pattern. Timing information is also used by the medium reservation mechanisms.
In addition to local station timing, each station in a basic service area maintains a copy of the *timing synchronization function* (TSF), which is a local timer synchronized with the TSF of every other station in the basic service area. The TSF is based on a 1-MHz clock and "ticks" in microseconds. Beacon frames are used to periodically announce the value of the TSF to other stations in the network. The "now" in a timestamp is when the first bit of the timestamp hits the PHY for transmission.

6.2.1 Infrastructure Timing Synchronization

The ease of power management in an infrastructure network is based on the use of access points as central coordinators for data distribution and power management functions. Timing in infrastructure networks is quite similar. Access points are responsible for maintaining the TSF time, and any stations associated with an access point must simply accept the access point's TSF as valid.
When access points prepare to transmit a Beacon frame, the access point timer is copied into the Beacon's timestamp field. Stations associated with an access point accept the timing value in any received Beacons, but they may add a small offset to the received timing value to account for local processing by the antenna and transceiver. Associated stations maintain local TSF timers so they can miss a Beacon frame and still remain roughly synchronized with the global TSF. The wireless medium

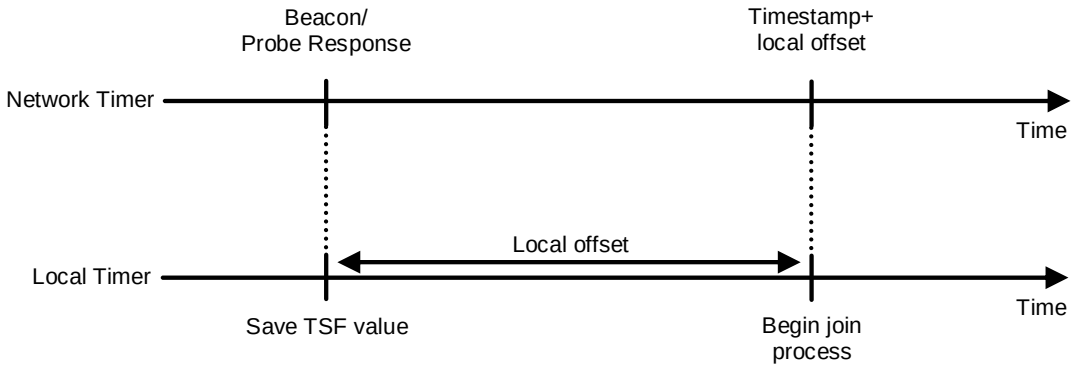

Figure 9.2. Matching the local timer to a network timer.

is expected to be noisy, and Beacon frames are unacknowledged. Therefore, missing a Beacon here and there is to be expected, and the local TSF timer mitigates against the occasional loss of Beacon frames.

To assist active scanning stations in matching parameters with the BSS, timing values are also distributed in Probe Response frames. When a station finds a network by scanning, it saves the timestamp from the Beacon or Probe Response and the value of the local timer when it was received. To match the local timer to the network timer, a station then takes the timestamp in the received network advertisement and adds the number of microseconds since it was received. Figure 9.2 illustrates this process.

7.2 Adaptive Power Management in Infrastructure WLANs

The main drawback of the wireless technology is the finite time battery energy of mobile computers that negatively affects performance. Given that activities such as transmission, reception and medium access are highly power consuming for mobile hosts, it is necessary to improve the efficiency of protocols that promotes Power Saving strategies to minimize power consumption. The IEEE 802.11 standard specifies a Power Management (PM) mechanism that allows a mobile station to enter in a state of low power consumption (*doze*) when its interface is idle. Much research has been conducted on PM and some inefficiencies and limitations of this mechanism have been revealed. Several solutions have been proposed to overcome such problems, and most of these refer to the IBSS mode. Among these, the work in [BCD01] is based on a Distributed Contention Control mechanism to guarantee the optimal power consumption by controlling the accesses to the shared transmission channel of a wireless LAN. The problem of dynamically selecting the optimal ATIM (Announcement Traffic Indication Map) window size is addressed in [JV02]. In [THH02], authors deal with the power management in MANET (Multi Hop Ad Hoc Networks). Differently from the IBSS architecture, only few works consider the iBSS system. In [CSE04] and [T00], application dependent solutions are described to improve the efficiency in power management. In [CSE04], the authors propose a technique called UPSD for voice services (Unscheduled Power-Save Delivery). The main idea is to define an unscheduled service period, which is a series of contiguous intervals of time during which a station is expected to be awake. For an given downlink flow (i.e., reception of voice streaming packets), the station specifies UPSD power management and triggers an unscheduled service period by notifying this to the AP. The AP then sends the frames of the downlink flow during this period. This solution is advantageous in case of traffic flows with smooth rate variations, as in case of voice services, while it may be too complex for bursty traffic. Author in [T00] proposes to separate the transmissions of hosts subject to power management constraints from those of hosts without such constraints. Additionally, for the first group of hosts an explicit traffic re-shaper function is added at the AP scheduler. Not only does such a scheduler operate on the basis of the packet arrival times, length, and deadlines, but it also takes queue sizes into account. This algorithm works only with the Point Coordination Function (PCF) mechanism. To reduce the number of collisions when the stations poll the AP to retrieve the buffered traffic, authors in [LN05] propose an algorithm which lets the AP schedule the sequence of poll messages from the stations in doze within each beacon interval. The AP uses a bit map to provides such a sequence, which is piggybacked either in TIM (Traffic Indication Map) messages at the beginning of

the beacon interval or in data frames within the beacon interval. In [CH04], the listen interval (period during which a station in doze wakes up to check if the AP has pending frames) is adjusted according to a power management policy that keeps into account frame loss and delay.
We believe that the PM algorithm used in iBSS may still be improved by allowing the AP to defer the waking of stations in doze to future beacon intervals on the basis of the actual network traffic and we also think that these improvements should go towards an application-independent approach so as not to increase the algorithm complexity. In this work, we then propose a strategy that introduces significant changes in the AP behavior, which acquires a high decisional power. We start analyzing the energy required for the successful reception of a frame from the stations that, when in doze state, are woken up by the AP. This analysis puts in evidence that such energy strongly depends on the number of stations contending the channel: this means that in case of high traffic, the stations to be woken up could incur into long waiting times that can negatively affect the performance of the system, introducing delays and leading to high energy consumption. Three simple but effective PM algorithms are proposed, which are based on giving the AP the authority to decide whether a station in doze mode with pending frames should be woken up or not. According to the first one, the decision is taken on the basis of the probability to find lower traffic in future beacon intervals before a maximum allowed delay. The second and the third ones are based on a cost function that weighs both energy consumption and introduced latency; the solution is found so that the cost function is minimized.

7.2.1 Evaluation of energy consumption

A major point in power management is the evaluation of the energy consumption for the successful reception of a frame by a station woken by the AP. In this Section, we provide the formulas used for the computation of this measure. To this, we mainly resort to the modeling and results in [WA04], [TC01] and [B00], with some key differences that are underlined in the following.
Consider a wireless cell with r contending stations with at least one frame to transmit/receive. In this analysis we focus on a generic station that was in doze mode, has been woken by the AP, and has to contend the use of the channel with other r-1 stations (among which some are active and some other have just been woken up) to receive the frames buffered at the AP. The average energy consumed for the reception of a frame from the AP is given by the sum of four main factors: the energy used during the backoff stages E_{BC}, the energy waste due to collisions E_C, the energy spent overhearing the other transmissions E_{fr}, and the energy necessary to successfully receive a frame from the AP E_{tx}. To define

these terms, we firstly need to introduce some useful expressions. The probability that a transmission incurs a collision can be approximated as in [TC01] with $p = 1-(1-1/E[BC])^{r-1}$, where $E[BC]$ is the expected value of the backoff counter:

$$E[BC] = 0.5 \cdot CW_{\min}\left(1 - p - p(2p)^{w}\right) \cdot (1-2p)^{-1}. \quad (1.2)$$

It's easy to see that the average probability of each station to send a frame is the inverse of this expectation. The probability that a woken station suffers a certain number N_C of collisions before successfully transmitting can then be written as:

$$\Pr\{N_C = i\} = p^{i}(1-p), \quad (2.2)$$

with expected value:

$$E[N_C] = \sum_{i=0}^{\infty} i \cdot p^{i}(1-p) = \frac{p}{1-p}. \quad (3.2)$$

When a collision occurs, the contending stations start decrementing their BCs again after a DIFS interval; given that we are considering the basic access mechanism, the time T_C during which the channel remains busy for a collision is given by DIFS+T_{frame}, where T_{frame} is the transmission time of a data frame [B00]. We are now able to define the first two contributes to the total average energy:

$$E_{BC} = P_{list} \cdot (E[N_C]+1) \cdot E[BC] \cdot T_{slot}, \quad (4.2)$$

$$E_C = P_{list} \cdot E[N_C] \cdot T_C, \quad (5.2)$$

where P_{list} is the power consumed by the station in the listening state. To derive E_{fr}, we need to define the average number of overheard transmissions by a woken station during its backoff stages:

$$\overline{N}_t = (E[N_C]+1) \cdot E[BC] \cdot P_{tr}, \quad (6.2)$$

where P_{tr} is the probability that at least one of the r-1 stations occupies the channel for a transmission. Using the modeling in [B00], it results to be $P_{tr} = 1-(1-1/E[BC])^{r-1}$, which is the same expression of p.

Among these overheard transmissions, $\overline{N}_t \cdot P_s$ will be successful and $\overline{N}_t \cdot (1-P_s)$ will be unsuccessful due to collisions, where P_s is the probability that any of these transmissions is successful:

$$P_s = P_{tr}^{-1} \cdot (r-1) \cdot E[BC]^{-1}\left(1-E[BC]^{-1}\right)^{r-2}. \quad (7.2)$$

The energy spent in overhearing the transmissions is thus given by:

$$E_{fr} = \overline{N}_t \left[P_s T_s + (1 - P_s) T_C \right] \cdot P_{list}, \tag{8.2}$$

where

$$T_s = \begin{cases} DIFS + T_{frame} + SIFS + T_{ACK} & \text{for an active station} \\ DIFS + T_{POLL} + 2SIFS + T_{frame} + T_{ACK} & \text{for a woken station} \end{cases} \tag{9.2}$$

is the duration of a successful transmission. T_{POLL} and T_{ACK} are the transmission times for the PS-Poll and the ACK frames, respectively. Differently from [WA04], in this work we keep also into account the duration of a successful transmission for a woken station that has to retrieve a buffered frame from the AP. The energy consumption for a successful transmission is:

$$E_{tx} = P_{tx} \cdot T_{POLL} + P_{list} \cdot (DIFS + 2SIFS) + P_{rx} \cdot (T_{frame} + T_{ACK}), \tag{10.2}$$

where P_{tx} and P_{rx} represent the power consumed by the woken station in transmission and receipt states, respectively.

We have now all the elements to define the total average energy required for a woken station to successfully receive a frame from the AP:

$$\overline{E} = E_{BC} + E_C + E_{fr} + E_{tx}. \tag{11.2}$$

This term strongly depends on the number of contending stations and, in an environment with high power constraints, has to be as low as possible. Figure 10.2 draws $\overline{E}$ as a function of the number of contending stations, with system parameters set according to Table 4.2. It is worth noting that this curve has a linear behavior. Applying a least-squares regression to $\overline{E}$, we have obtained a straight line with slope *a*=0.006 and offset *b*=-0.0078, with residual error equal to 0.0251.

Table 4.2. Values of parameters used in experiments.

Parameter	**Value**
T_b	100 msec
Tx Rate	2 Mbps
T_{slot}	20 μsec
T_{POLL}	80 μsec
CW_{max}	1024
$CW_{\min}$	128
P_{tx}	1.65 W
P_{list}	1.15 W
P_{rx}	1.4 W
T_{frame}	4 msec
SIFS	10 μsec
DIFS	50 μsec
T_{ACK}	56 μsec

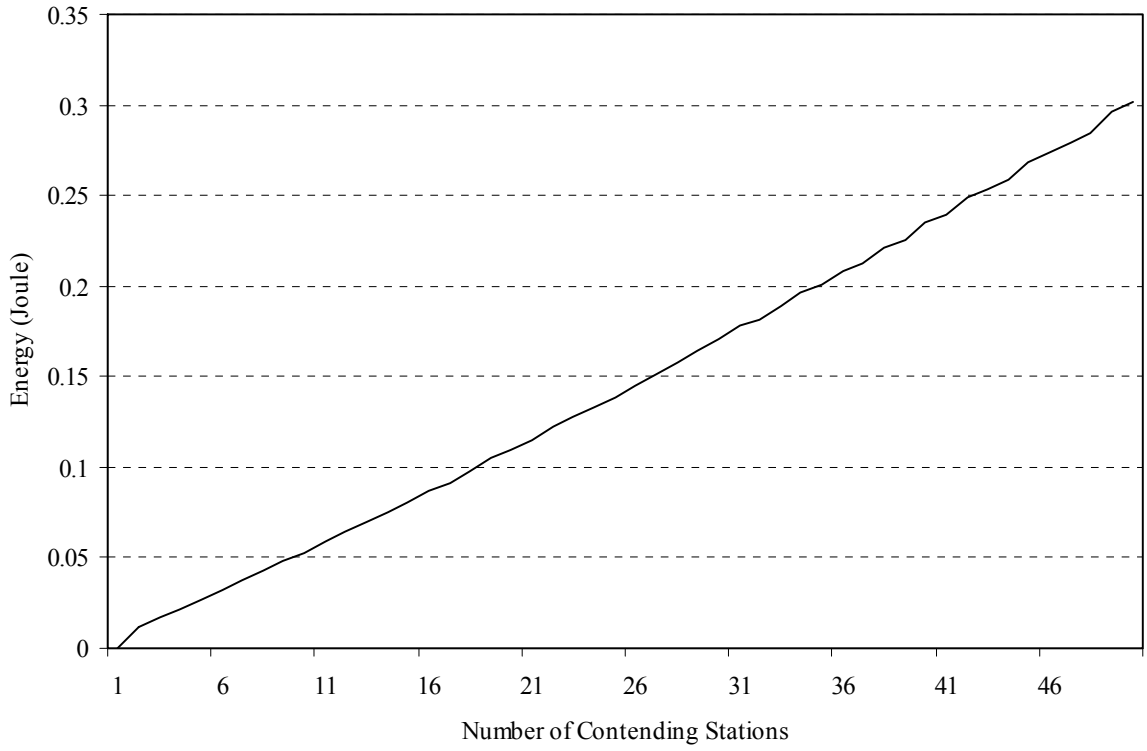

Figure 10.2. Average energy dissipation for frame transmission varying network traffic.

7.2.2 Adaptive waking of stations in doze mode

The proposed adaptive waking of stations in doze mode is aimed at selecting the stations to be notified of pending frames on the basis of the current traffic on which the frame transmission energy consumption depends and waiting time of pending frames. At the beginning of each beacon interval, the AP observes the queues of frames directed to doze stations and decides which stations to wake up, putting the correspondent AIDs in the TIM. This means that, differently from the IEEE 802.11 protocol, some stations with pending frames could be not woken up, deferring the transmission to successive beacon intervals. Obviously, this approach introduces latency: if the AP decides to keep in doze state a station with a pending frame, this will suffer an additional latency equal to the length of a beacon interval. Several strategies may be devised to take into account energy dissipation and latency. Herein, we propose three variants which are described in next subsections using the notation described next.

Consider a wireless cell controlled by an AP with *M* stations implementing the PM. Let *i* and *k* index the stations and the beacon intervals, respectively. At the beginning of each beacon interval *k*, the AP lists in the TIM the AID's of the stations to be woken up. Let vector $\mathbf{\Lambda}_k = \left\{\lambda_k^i, i = 1,..., M\right\}$ represent the PM decision of the AP at the beginning of interval *k*:

$$\lambda_k^i = \begin{array}{l} 1 \text{ if the AP wakes up station } i \\ 0 \text{ otherwise} \end{array} .$$

Accordingly, the number of stations woken up is $n_k = \left\|\mathbf{\Lambda}_k\right\|_1$. Note that λ_k^i has to be zero for the active stations and for stations in doze state without pending frames. To take into account the state of stations at the beginning of interval *k* vector $\mathbf{B}_k = \left\{\beta_k^i, i = 1,..., M\right\}$ has been introduced:

$$\beta_k^i = \begin{array}{l} 1 \text{ if station } i \text{ is in doze with pending frames} \\ 0 \text{ otherwise} \end{array} .$$

The number of stations that the AP wakes up in the IEEE 802.11 PM is $m_k = \left\|\mathbf{B}_k\right\|_1$. In fact, the standard states that all the stations in doze with pending frames have to be woken up.

Additionally, let $\mathbf{A}_k = \left\{\alpha_k^i, i = 1,..., M\right\}$ represent the evolution of the state within a beacon interval:

$$\alpha_k^i = \begin{cases} 1 & \text{if the AP receives frames for station } i \text{ in doze} \\ -1 & \text{if station } i \text{ wakes up by itself} \\ 0 & \text{if station } i \text{ maintains its state} \end{cases}.$$

To take into account the latency introduced by the proposed algorithm, we also make use of vector $\mathbf{D}_k = \left\{ d_k^i, i = 1,...,M \right\}$, which represents the number of beacon intervals during which the transmission of pending frames have been delayed.

Based on the previous definitions, the system evolves according to the following expressions:

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \mathbf{A}_k - \mathbf{\Lambda}_k \tag{12.2}$$

$$d_{k+1}^i = \begin{cases} d_k^i + \beta_k^i & \text{if } \lambda_k^i = 0 \\ 0 & \text{if } \lambda_k^i = 1 \;\middle|\; \alpha_k^i = -1 \end{cases}, \tag{13.2}$$

with the following constraints:

$$\begin{aligned} &\mathbf{\Lambda}_k \le \mathbf{B}_k \\ &\mathbf{0} \le \mathbf{A}_k + \mathbf{B}_k \le \mathbf{1}, \\ &\mathbf{D}_k \le \mathbf{1} \cdot d_{\max} \end{aligned} \tag{14.2}$$

where $d_{\max}$ is an integer parameter representing the maximum tolerable latency (in number of beacon intervals) which may be linked to specific applications. Finally, we denote with μ_k the number of active stations in the cell with frames to transmit.

Next subsections illustrate three alternative algorithms that we propose for the control of vector $\mathbf{\Lambda}_k = \left\{ \lambda_k^i, i = 1,...,M \right\}$, which represents the solution of the power control algorithm.

A. Algorithm 1

As described in Section II, woken stations contend the channel with active stations for the reception of pending frames and, accordingly, spend an amount of energy strictly linked with the number of active stations, which varies over the time. A simple but effective strategy is then to monitor the temporal evolution of the number of active users and to defer the waking of stations in doze mode in case this number is expected to decrease during next beacon intervals.

To implement this strategy we construct a *transmitting stations distribution* (TSD), which is a statistical representation of the number of active stations at the beginning of any beacon interval. There

are several approaches that can be used to build it on the basis of observed measurements. The adopted approach is to store and track the observed values using a measured histogram to approximate the distribution. Each value in the histogram counts the number of occurrences of a certain number of active stations. Since network characteristics vary with time, these statistical trends vary, requiring an update of the information to take into account the evolutions. The flush and refresh approach is used here, that is, the TSD is built on the basis of the past observed L intervals so that the contribution of observations older than L are not considered anymore in the histogram.

Let $f_k(x)$ and $F_k(x)$ denote the TSD and its cumulative distribution function at beacon k, respectively. Defined $\Phi_k(x)=1-F_k(x)$, at the beginning of beacon interval k the AP implements the following solution $\mathbf{\Lambda}_k^*$:

$$\text{if } \Phi_k(\mu_k) \geq \Phi_{th} \;\; \mathbf{\Lambda}_k^* = \mathbf{B}_k;$$

else

$$\text{if } D_k^i = d_{\max} \;\; \lambda_k^{i*} = 1$$

$$\text{else } \lambda_k^{i*} = 0$$

Φ_{th} is a threshold usually set to a value in the range 0.4÷0.5. This is based on the observation that if $\Phi_k(\mu_k)$ is low, it is advantageous from an energy saving point of view to defer the transmissions towards stations in doze, which should likely encounter lower traffic in the future with a reduction of energy consumption. Contrarily, when this probability is high, there is no sense to still keep in the buffer the frames, and the relevant stations are then notified. Note that this is also applied in case of low values of $\Phi_k(\mu_k)$ when pending frames have reached the maximum delay. Figure 11.2 shows the typical shape of $\Phi_k(x)$ when using the system configuration adopted during the experiments.

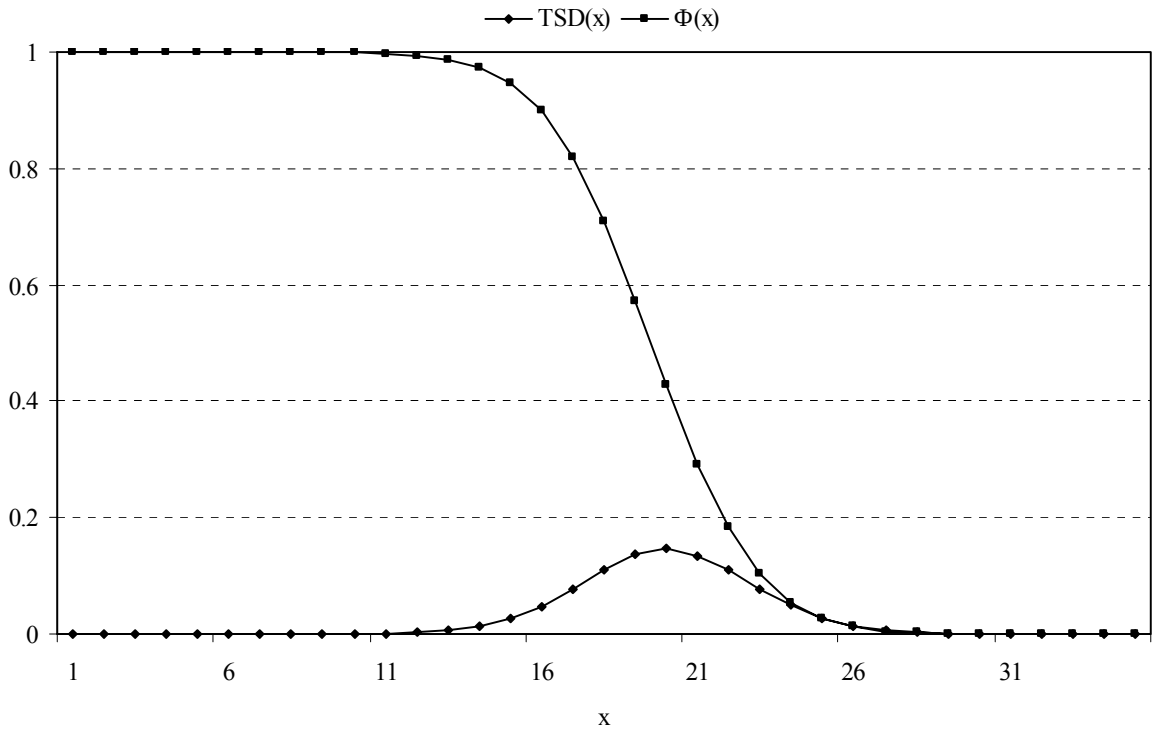

Figure 11.2. Shape of the transmitting stations distribution (TSD) and the corresponding Φ_k function, which represents the probability of having a number of active stations at the beginning of the beacon interval greater than x.

Notwithstanding its simplicity, this strategy is quite powerful as shown in the experiments Section. However, it has the main drawback of not taking into account the latency suffered by the deferred stations: in fact, with this algorithm, the sole control on delay is given by the constraint on the maximum allowable value.

B. Algorithm 2

To overcome the mentioned drawback of algorithm 1, we propose to evaluate the expected energy consumption and the latency associated to each AP decision and to find the optimal balance between these two contributions according to a cost function $F_k(.)$:

$$F_k(\mathbf{\Lambda}_k) = X_k(\mathbf{\Lambda}_k) + \vartheta \cdot Y_k(\mathbf{\Lambda}_k), \tag{15.2}$$

where $X_k(.)$ and $Y_k(.)$ measure energy consumption and frame latency, respectively. The parameter $\vartheta > 0$ is a weight parameter whose importance will be discussed in the following. At the beginning of every interval *k*, the AP seeks for the solution $\mathbf{\Lambda}_k^*$ that minimizes (15.2) subject to the constraints in

(14.2). The term $X_k(.)$ measures the energy consumption of the woken stations at the beacon k according to (11.2). Clearly, this term weights the choice of the AP to wake up a certain number n_k of stations in doze. Accordingly, the used expression follows:

$$X_k(\mathbf{\Lambda}_k) = n_k \cdot \overline{E}(n_k + \mu_k). \tag{16.2}$$

Note that $n_k = \|\mathbf{\Lambda}_k\|_1$ and that $\overline{E}(.)$ is given by (11.2).

As to $Y_k(.)$, it measures the latency of pending frames whose transmission is deferred by the AP:

$$Y_k(\mathbf{\Lambda}_k) = T_b \cdot \|\mathbf{D}_{k+1}\|_1, \tag{17.2}$$

where $\mathbf{D}_{k+1}$ evolves according to (13.2).

In case any stations with pending frames are notified, the resulting cost function is the following:

$$F_k^{PM} = m_k \cdot \overline{E}(m_k + \mu_k). \tag{18.2}$$

This is the expression associated to the standard strategy implemented in the IEEE 802.11 standard and will be used for comparison purposes during experiments.

The parameter ϑ allows the control procedure to balance energy consumption versus latency. When set to high values, more importance is given to the latency and the behavior of the algorithm converges toward the standard IEEE strategy, where the stations in doze with pending frames are always woken. Contrarily, low values bring to frequent postponements of frame transmissions with the aim of reducing station energy consumption. A way to make this setting easier is to think of the importance of the energy dissipated during a beacon interval when in receipt state (given by $P_{rx}T_b$) in number s of one-frame one-beacon interval delays. The parameter ϑ should then be set to sP_{rx}.

In what follows, we further investigate the effects of ϑ on the proposed strategy by writing an approximation of the relationship between the number of stations woken by the AP and the setting of the weight parameter at varying network traffic. We start by considering a linear approximation for $\overline{E}$ and assuming that for each stations with pending frames $d_{k+1}^i = D < d_{\max}$. Since the context is clear, we drop the dependence on k from the notation. The cost function for the proposed strategy can be approximated as follows:

$$F(\mathbf{\Lambda}) \cong n \cdot [a(n+\mu)+b] + \vartheta(T_b D) \cdot (m-n), \tag{19.2}$$

where $0 \le n \le m$ represents the number of woken stations and the pair a and b come from the linear approximation of the energy function. Expression (19) can be thus written as follows:

$$F(\Lambda) \cong an^2 + (a\mu + b - \vartheta T_b D)n + \vartheta T_b Dm \,. \qquad (20.2)$$

The AP seeks for the solution n^* that minimizes equation (20.2), which is the abscissa corresponding to the vertex of the parabola. This is the number of transmissions of stations in doze that the AP decides to wake up. As already noted, the greater ϑ, the greater n^* is.

We can now obtain the range of values for the weight parameter imposing that n^* is included in the interval [0,*m*]: when the vertex is located at the origin, the AP defers all the stations in doze with pending frames; this happens when the weight parameter is set to its minimum value:

$$\vartheta_{\min} = (T_b D)^{-1} \cdot (a\mu + b). \qquad (21.2)$$

For values of ϑ lower than $\vartheta_{\min}$ the AP defers all stations in doze.

The AP wakes up all the stations in doze when n^* is equal to *m*; this happens for a value of the weight parameter equal to:

$$\vartheta_{\max} = (T_b D)^{-1} \cdot (2am + a\mu + b) \qquad (22.2)$$

For values of ϑ higher than $\vartheta_{\max}$ the AP wakes all stations in doze.

We have then found an approximation of the relationship between the number of the woken stations and the weight parameter:

$$n^*(\vartheta) = \begin{cases} 0 & \text{for } \vartheta \le \vartheta_{\min} \\ \dfrac{\vartheta T_b D - a\mu - b}{2a} & \text{for } \vartheta_{\min} < \vartheta < \vartheta_{\max} \\ m & \text{for } \vartheta \ge \vartheta_{\max} \end{cases} . \qquad (23.2)$$

Figure 12.2 draws (23.2) for various combinations of μ and *D*, which are shown between square brackets. For a couple μ-*D*, $\vartheta_{\min}$ and $\vartheta_{\max}$ are given by the abscissas corresponding to an ordinate equal to 0 and *m* (5 in this case), respectively. While the resulting values are quite close each other, they vary a lot from a combination of μ-*D* to another. It means that for a given weight parameter, the decision of the AP most of the time results in either waking or deferring all the stations in doze. As expected, the curve moves to the right as the number of woken stations increases, while moves to the left as the average latency increases. This graph helps in deciding on which parameter value to use by selecting the combination μ-*D* that represents the threshold between waking and deferring actions. For instance, a value around 0.5 should be selected in case we prefer transmitting when the average delay is 0.3 seconds and the number of active stations is at most 20.

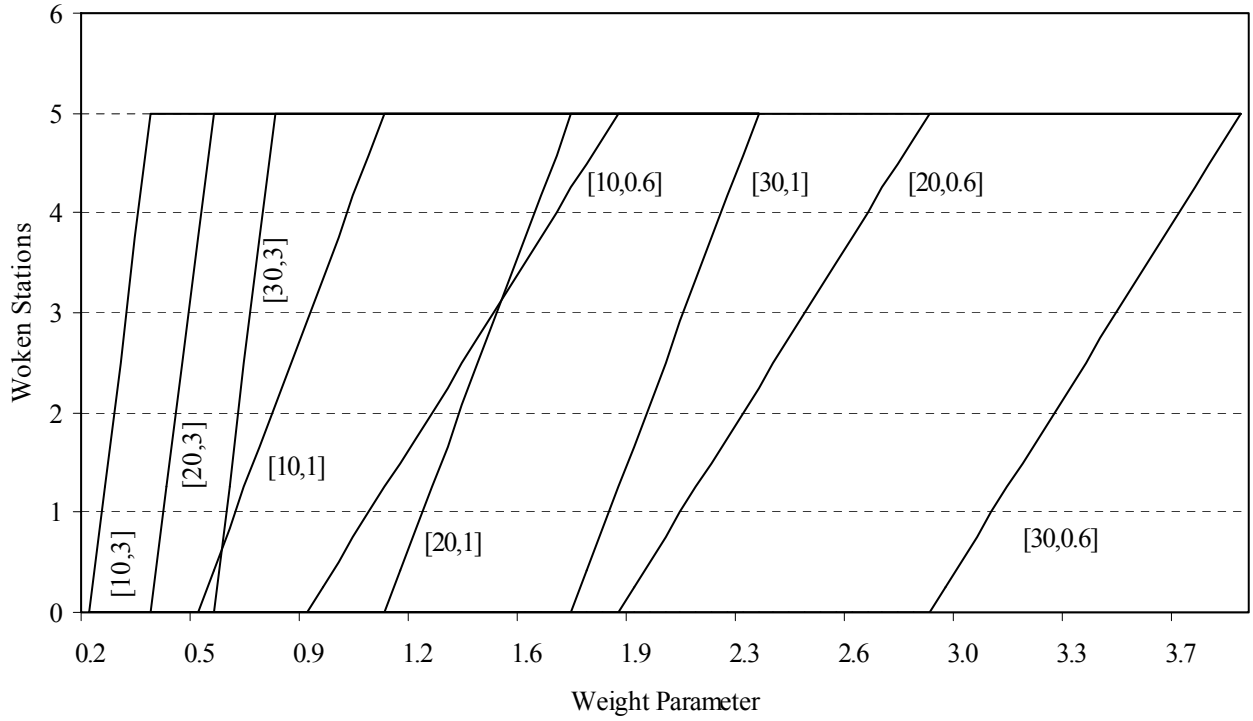

Figure 12.2. Number of woken stations versus the weight parameter at various values of number of active stations and average latency shown between square brackets in the graph. Note that the number of stations in doze mode is 5.

C. Algorithm 3

For each possible solution $\mathbf{\Lambda}_k$, (15.2) weights both the energy spent by the stations woken up by the AP for frame reception and the latency suffered by the frames whose transmission is deferred. However, it doesn't take into account the amount of energy that will be consumed in the future for the frames that remain in the buffer and need to be transmitted in a next interval. In case such deferred transmissions encounter low traffic level with minor energy consumption, this aspect is insignificant. It would mean that delaying the transmissions causes only an increase in the delay and a saving of the energy that would be consumed waking the stations in doze with the current network traffic. But this assumption may be inappropriate for some network circumstances, i.e., when the network traffic is not characterized by a high temporal variability with respect to intervals of time equal to d_{max} beacon intervals. In view of this observation, we have proposed a third algorithm which instead of evaluating the energy consumption considers the loss of energy in case of immediate waking (respect with deferring). Accordingly, the first addend in (15.2) is defined as follows:

$$X_k(\mathbf{\Lambda}_k) = \mathbf{\Lambda}_k \cdot \left[\overline{E}(n_k + \mu_k) \cdot \mathbf{1}^{\mathbf{r}} - \mathbf{E}_{\min,k}\right] , \qquad (24.2)$$

where $\mathbf{E}_{\min,k} = \left\{E^{i}_{\min,k}, i = 1,..., M\right\}$ is a vector where each element $E^{i}_{\min,k}$ represents the minimum energy consumption for station *i* that can be reached by appropriate deferring choices. Since $E^{i}_{\min,k}$ cannot be predicted deterministically, we resort to an empirical approach which is based on approximating this value with the energy that is consumed with a number of active stations equal to the minimum amount that is observed in a window of $d_{\max} - d^{i}_{k}$ beacon intervals. Accordingly, let $\Psi_k(x)$ represent the minimum number of active stations in a window of length *x*, then:

$$E^{i}_{\min,k} = \overline{E}\left(\Psi_k\left(d_{\max} - d^{i}_{k}\right)\right). \tag{25.2}$$

$\Psi_k(x)$ is estimated online on the basis of the observed number of active stations. In particular, the AP evaluates the average of the minimum number of active stations that have been observed over any windows of length *x* during the past *N* beacon intervals. Note that whatever the length of *x* is, the number of windows considered is *N*-1. As for $\Phi_k(x)$, $\Psi_k(x)$ is updated by means of a flush and refresh approach, so that the values observed during intervals older that *N* intervals are discarded.

Figure 13.2 shows a representative behavior of function $\Psi_k(x)$ when using the system parameters considered in the experiments discussed in next section. As expected, it is a function with a monotone decreasing behavior showing that postponing the transmission of even few beacon intervals allows the AP to wake up the stations in doze with quite lower traffic level.

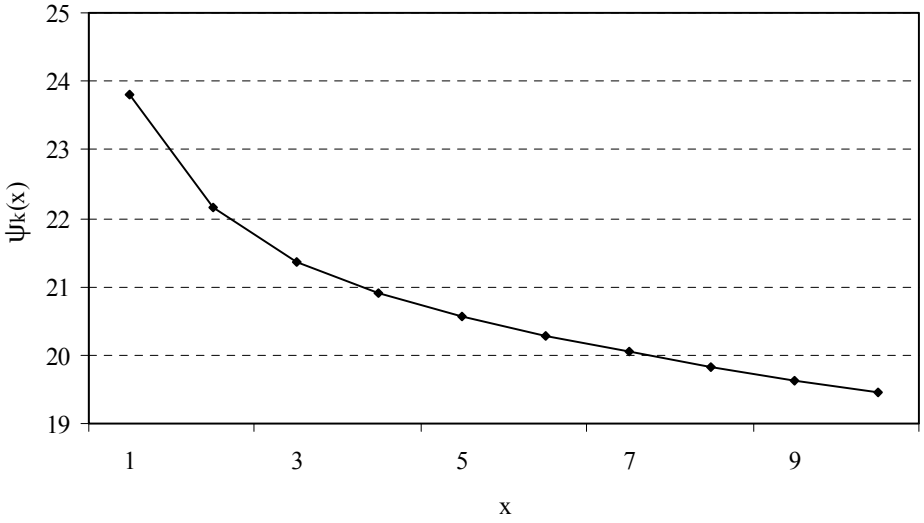

Figure 13.2. Function $\Psi_k(x)$ representing the expected minimum number of active stations over a window of x beacon intervals.

Also in this case, when setting the weight parameter the same analysis conducted for algorithm 2 can also be applied. Assuming that $E^{i}_{\min,k} = E_{\min}$ for any stations in doze, the maximum and minim value of the weight parameter become:

$$\vartheta_{\max} = (T_b D)^{-1} \cdot (2am + a\mu + b - E_{\min}). \tag{26.2}$$

$$\vartheta_{\min} = (T_b D)^{-1} \cdot (a\mu + b - E_{\min}). \tag{27.2}$$

While the expression of the number of woken stations is given by the following formula:

$$n^*(\vartheta) = \begin{cases} 0 & \text{for } \vartheta \le \vartheta_{\min} \\ \dfrac{\vartheta T_b D + E_{\min} - a\mu - b}{2a} & \text{for } \vartheta_{\min} < \vartheta < \vartheta_{\max} \\ m & \text{for } \vartheta \ge \vartheta_{\max} \end{cases} \tag{28.2}$$

7.2.3 Experiments

The performance of the proposed strategies have been evaluated in terms of average energy saving and latency increase with respect to the standard PM algorithm. In this section, we refer to the results obtained with a wireless cell of *M*=40 stations and have been obtained by averaging the outcomes for *N*=10,000 beacon intervals. The system parameters described throughout the paper have been set according to Table 4.2. Additionally, the evolution of the system, which depends on the observed vectors $\mathbf{A}_k$ other than the decision vectors $\mathbf{\Lambda}_k$, is driven by the following modeling: the arrival process of packets addressed to each station is modeled with a Poisson point process with an average arrival rate of 5 packets per second; the number of active stations has been modeled as a birth-death model, with birth and death processes represented with Poisson and Erlang distributions [FZ02], respectively, with an average number of active stations equal to 20. As to $d_{\max}$, it has to be set according to the delay tolerance of specific applications; in our simulations we have used a value of 5 beacon intervals, corresponding to 500 msec. The term T_S in (9.2), required in algorithms 2 and 3, is computed as a weighted average of the values in case the transmission is done by an active station or a station just woken up.

In our analysis the average energy saving for the woken stations with respect to the standard PM and the average latency are computed as follows:

$$\text{Energy Saving}(\%) = 100 \cdot \left(1 - \frac{\frac{1}{N}\sum_{k=1}^{N} n_k \cdot \overline{E}(n_k + \mu_k)}{\frac{1}{N}\sum_{k=1}^{N} m_k \cdot \overline{E}(m_k + \mu_k)}\right), \tag{29.2}$$

$$\text{Latency} = T_b \frac{\sum_{k=1}^{N} \mathbf{D}_k \cdot \mathbf{\Lambda}_k}{\sum_{k=1}^{N} \|\mathbf{\Lambda}_k\|_1}. \tag{30.2}$$

A good power management algorithm should be able to maximize energy saving for a given latency. We can get a trade-off curve for a given algorithm by plotting these two quantities for the entire range of values of the algorithm control parameters. Once we have this curve, we can better judge which algorithm performs better: if the curve achieved with one algorithm falls below the curve achieved with a second, the latter performs better. Figure 14.2 shows the resulting trade-off curves for the three proposed algorithms varying Φ_{th} for algorithm 1 and ϑ for algorithms 2 and 3. The first consideration we may draw is that algorithm 2 outperforms the others. Consider at first the second and third algorithms which are characterized by a quite similar underlying principle. It seems that the estimated $\mathbf{E}_{\min,k}$, added in the third algorithm, doesn't improve the evaluation of the estimated energy consumption. This is opposite to what expected; it should be concluded that the estimation of $\mathbf{E}_{\min,k}$

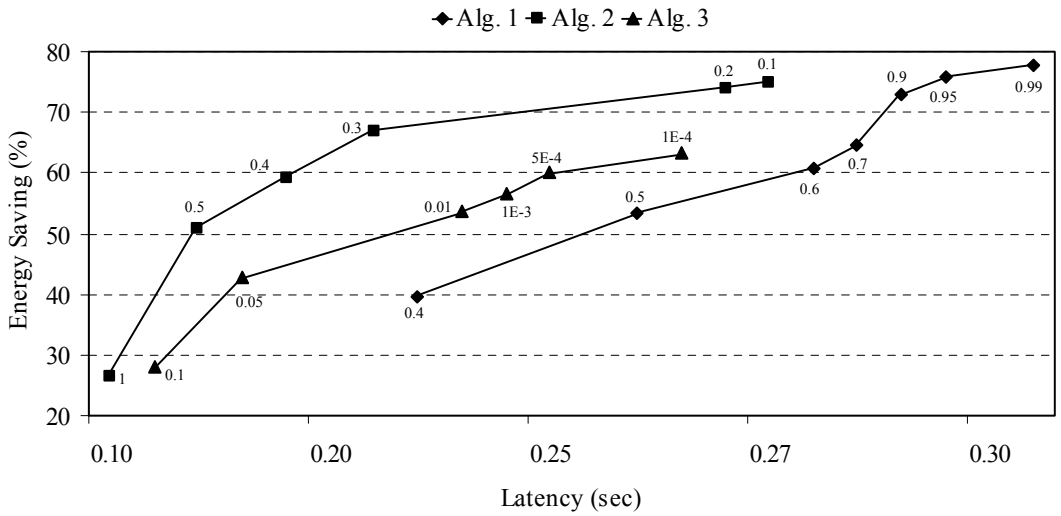

Figure 14.2. Comparison of the three proposed algorithms in terms of trade-off curves energy saving-latency. The different points in the graph have been obtained varying Φ_{th} for algorithm 1 and ϑ for algorithms 2 and 3.

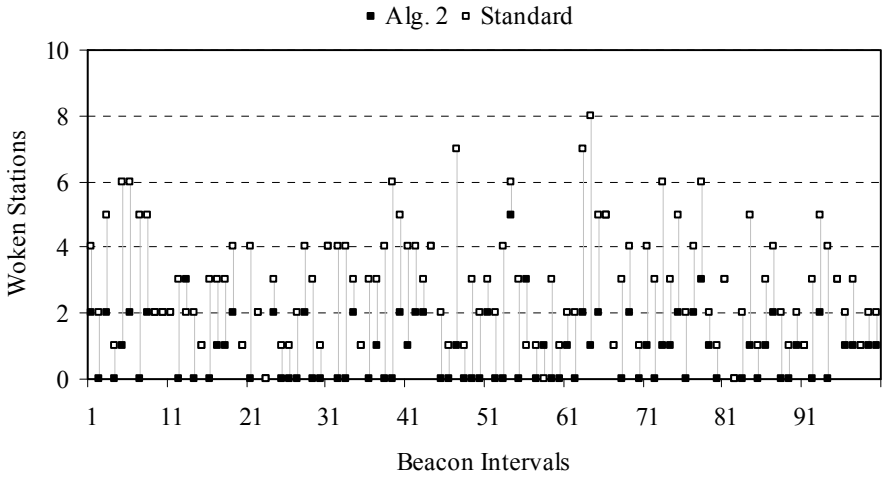

Figure 15.2. Number of woken stations when adopting the standard PM and algorithm 2 with ϑ=0.3.

through $\Psi_k(x)$ is not accurate as desired and the error introduced makes the approach proposed ineffective. As to the first algorithm, it is quite simple and as expected provides the worst results in terms of energy saving and latency ratio. However, in case the latency is not a problem, it is a good solution since allows for high energy saving levels when Φ_{th} is around 0.9. Note that even if we increase Φ_{th}, we don't get a significant improvement; in fact, an energy saving of around 78% seems to be a limit for the considered system configuration. It is important to note that the latency we are computing is the one related to the only stations in doze.

To better evaluate the benefits introduced by the proposed approach, we now examine the behavior of algorithm 2 with $\vartheta=0.3$ and compare it with those of the standard PM during the same run of 100 beacon intervals. In Figure 15.2, m_k and n_k are compared: as expected, our strategy tends to wake up a number of stations lower than that in case of the standard PM for almost all the beacon intervals. This fact can be easily explained by means of Figure 16.2, where we compare the number of frames directed to woken stations and sent by the AP using the two algorithms. Note that with the proposed strategy, the AP aggregates more frames directed to the same station in doze and transmits these during the same beacon interval. This allows the AP to wake up a lower number of stations, which however receive a higher number of frames per interval on average. Note that this partially contributes to the energy saving. This fact leads to the generation of more bursty traffic than the case of standard PM with the same average value of sent frames per interval equal to 4. Such trend is strictly linked to the behavior of the active stations. In Figure 17.2, we compare the number of active and woken stations: it can be noted that when the number of active stations is low, the number of woken stations reaches high values,

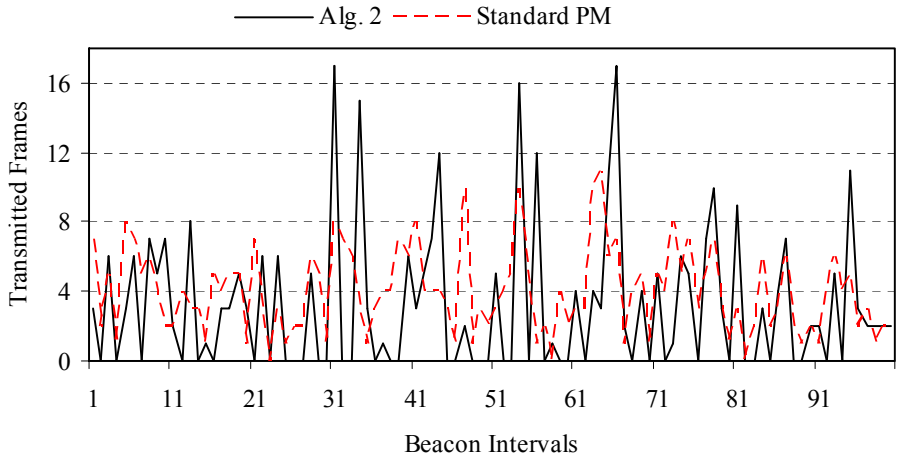

Figure 16.2. Number of frames transmitted to woken stations by the access point when using the standard PM and algorithm 2 with ϑ =0.3.

and vice versa. This means that this strategy adequately reacts to the temporal changes of the system and tends to wake up the stations in doze when the network load is low. Figure 18.2 provides a comparison of the total average energy $\overline{E}$ consumed by the woken stations between the proposed and the standard PM as computed by means of expressions (16.2) and (18.2), respectively. Note that there is a significant mismatch between the two functions for almost all beacon intervals; in fact, we have obtained an average reduction of energy consumption of about 67%. The drawback of this reduction is the increase in frame latency in the AP queues. The average latency obtained in this run is equal to 0.22 sec. A further aspect that has to be considered is the high level of buffer occupancy introduced by the proposed algorithm, which is drawn in Figure 19.2. Note that there are no deferred frame transmissions when using the standard PM.

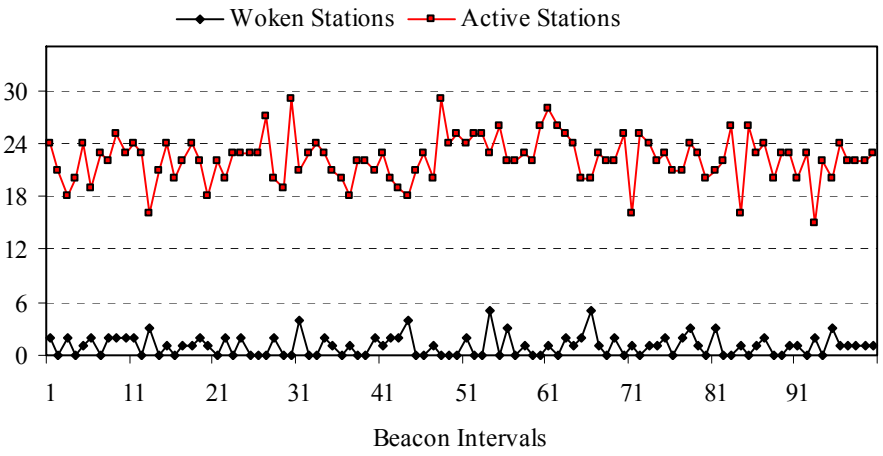

Figure 17.2. Temporal evolution of the number of woken stations and the number of active stations using algorithm 2 with ϑ=0.3.

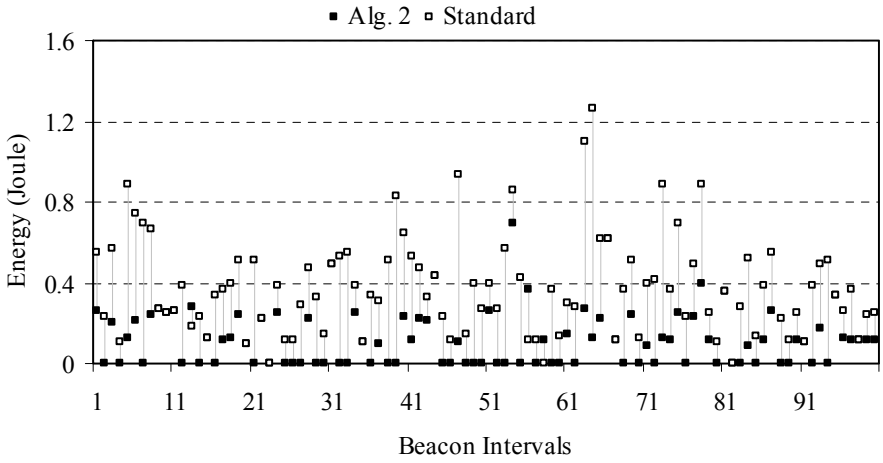

Figure 18.2. Energy consumption for frame transmission to woken stations when using the standard PM and algorithm 2 with ϑ=0.3.

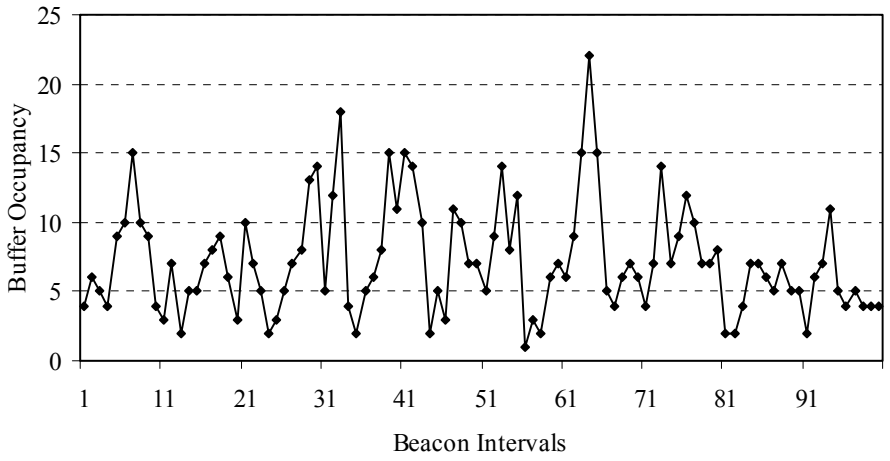

Figure 19.2. Temporal evolution of the buffer occupancy (in number of frames) using algorithm 2 with ϑ=0.3.

7.2.4 Conclusions

Stations in doze mode woken by the AP contend the channel with the active stations and then consume an amount of energy to receive the pending frame that is dependent on the current channel. On the basis of this observation, we have proposed an adaptive approach for the selection of the stations in doze to be woken up. The aim is to make the AP notify the presence of pending frames during beacon intervals with low traffic. Experimental results have confirmed the effectiveness of the proposed solution with an overall energy saving of about 75% at the expense of an increase in the latency of about 0.3 sec, which is an acceptable delay for the stations with energy constraints