DE GRUYTER OPEN

Int. J. Appl. Math. Comput. Sci., 2016, Vol. 26, No. 1, 147–160
DOI: 10.1515/amcs-2016-0010

amcs

A MATHEMATICAL MODEL FOR FILE FRAGMENT DIFFUSION AND A NEURAL PREDICTOR TO MANAGE PRIORITY QUEUES OVER BITTORRENT

CHRISTIAN NAPOLI [a,*], GIUSEPPE PAPPALARDO [a], EMILIANO TRAMONTANA [a]

[a]Department of Mathematics and Informatics
University of Catania, Viale Andrea Doria 6, 95126, Catania, Italy
e-mail: {napoli,pappalardo,tramontana}@dmi.unict.it

BitTorrent splits the files that are shared on a P2P network into fragments and then spreads these by giving the highest priority to the rarest fragment. We propose a mathematical model that takes into account several factors such as the peer distance, communication delays, and file fragment availability in a future period also by using a neural network module designed to model the behaviour of the peers. The ensemble comprising the proposed mathematical model and a neural network provides a solution for choosing the file fragments that have to be spread first, in order to ensure their continuous availability, taking into account that some peers will disconnect.

Keywords: P2P model, neural network, wavelet, diffusion, file sharing.

1. Introduction

Nowadays, a user can share files by means of several technologies, and each of them often relies on some mechanism that checks whether potential access bottlenecks will arise. Therefore, replicas are created to offload a single server. Several factors contribute to the selection of the file that will be replicated, i.e., the storage space available, the number of requests, the bandwidth, etc.

In peer to peer (P2P) systems using BitTorrent, a shared file is split into fragments and the least available ones are automatically chosen to be sent first to the users requesting the file (Cohen, 2008). Fragments availability is measured by the number of peers storing a file fragment at a given moment, and periodically computed by a *tracker* server storing peer ids, fragments held, and files requested (Cohen, 2003). In order to compute the distribution priority for each file fragment, it is of paramount importance to synchronize with the tracker and to get frequent updates since the status of the BitTorrent network is prone to rapid changes, due to the high variability of the number and availability of peers that can leave the system at any time (Kaune *et al.*, 2010). This occurs so frequently that such a fundamental BitTorrent mechanism may become ineffective, and as a result some fragments can quickly become unavailable. Moreover, the mechanism choosing fragments to spread is unaware of communication latencies among peers; as a consequence, fragment spreading occurs sooner on peers nearby the ones holding the fragment to be spread, and the furthest peers could disconnect before receiving the whole fragment.

This paper proposes a model for spreading file fragments that considers (i) latencies among peers, (ii) a time-dependent priority for a fragment to be spread, and (iii) the behaviour of peers for estimating their future availability. We take into account the fact that more time is needed to have a replica on the furthest peer ready to be served to other peers, when compared with a nearer peer. Moreover, the priority of fragments to be spread will be computed again over time, as their availability changes. The variation in priority is regulated in our model in such a way as to maximise the availability of fragments over time. To determine the dynamics of fragment spreading, we use a *diffusion model* developed by analogy to a diffusion model on a porous medium.

Moreover, we enhanced our mathematical model by using the results of an appropriate *neural predictor* as in the works of Napoli *et al.* (2014b; 2014a; 2015); Nowak *et al.* (2015), Woźniak *et al.* (2015) and Fornaia *et al.*

*Corresponding author

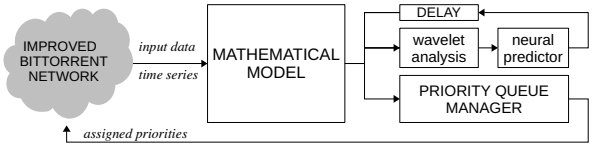


Fig. 1. Overview of the ensemble of components for the proposed solution.

(2015). This neural predictor aims at estimating the status evolution of the BitTorrent system, hence overcoming the sparse updates between peers and the tracker. Results provided by the neural network are fed to the above mentioned mathematical model, computing the fragments to be spread. As a result, BitTorrent clients could take early actions to facilitate the diffusion of file fragments, in order to cope with the availability of evolving fragments. Figure 1 shows an overall view of the proposed main components and their interactions.

The rest of the paper is structured as follows. Section 2 introduces the formalism used. Section 3 describes our diffusion model. Section 4 details the design of our neural predictor. Section 5 shows how the neural predictor has been used. Section 6 provides the results of our experiments. Finally, related works are discussed in Section 7, and Section 8 draws our conclusions.

2. Mathematical representations

In order to develop our diffusion model for BitTorrent based on a physical porous medium, some conventions must be chosen and some extrapolations are needed. We first describe a continuum system using a continuum metric; however, later on we will single out a few interesting discrete points of the continuum. Due to the analogy we make between a physical system and BitTorrent, we use a distance metric (named δ), which will be defined as the network *latency* among nodes, i.e., the hosts on a network holding *peers*, playing as *seeds* (peers providing fragments) or *leeches* (peers downloading fragments). Table 1 lists the symbols we use along with their explanation for quick reference, whereas the whole description is in the following.

For the nodes we use n^i or n^i_α: the first indicates a generic i-esime node on the BitTorrent network, the second indicates the α-esime node as seen from the i-esime node. Of course, n^i_α and n^j_α could be different nodes when $i \neq j$. Double indexing is needed since, when we use something like δ^{ij}, it will represent the distance of the j-esime node as measured by the i-esime node. Moreover, let us denote by P^{ij}_k the *probability of diffusion* for the k-esime file fragment from the i-esime node to the j-esime node. Finally, we distinguish between time and time steps: the first will be used for a continuum measure of temporal intervals and will be expressed by the Latin letter t, the second will indicate time steps (e.g., the steps of an iterative cycle) and we will use for it the Greek letter τ. Therefore, while $\delta^{ij}(t)$ will represent the continuous evolution during time t of the network latency δ, which measures the distance from the i-esime node to the j-esime node, the notation $\delta^{ij}(\tau)$ represents the same measure at the τ-esime step, i.e., the time taken by a ping from the i-esime node to the j-esime node, only for the specific time step τ. Finally, we will suppose that each node has the fragment z_k of a file z and is interested in sharing or obtaining other portions of the same file; hence, we will compute the probability-like function that expresses how easily the k-esime shared fragment is copied from the i-esime node to the j-esime node at a certain step τ, and we will call it $P^{ij}_k(\tau)$.

Eventually, we are interested in an analytical computation for the urgency to share a fragment z_k from n^i to n^j for a time step τ, and we will call it $\chi^{i,j}_k(\tau)$. In the following sections, we will distinguish between a measured value and a value predicted by a neural network using a tilde for predicted values as in $\tilde{x}$.

Table 1. Symbols used and their meaning.

n^i,n^j,n^α	Nodes of the network
z_k	File fragment
n^i_α	Node in the list of nodes held by n^α
δ^{ij}	Distance among nodes n^i and n^j
P^{ij}_k	Diffusion probability from n^i to n^j of z_k
Ω^i	Node list ordered based on distance from n^i
Φ	Concentration of file fragments
D	Diffusion coefficient
T_k	Total users sharing or requesting z_k
S_k	Seeds of z_k
ρ_k	Share ratio of z_k
$\chi^{i,j}_k$	Urgency to share fragment z_k from n^i to n^j
s	Data time series (signal)
$\hat{\mathbf{W}}$	Wavelet transform
ψ	Wavelet function
φ	Wavelet dual scaling function
$d_{j,l}$	Wavelet coefficients
a_M	Wavelet residuals
$\hat{N}$	Neural network
$\mathbf{u}$	Neural network input
t	Time (continuous)
τ	Time step (discrete)

3. Fragment diffusion on a P2P network

In our work, we compare the spreading of file fragments for a shared file to the diffusion of mass through a porous means. To embrace this view, it is mandatory to develop some mathematical tools, which are explained in the following.

3.1. Spaces and metrics. Users in a P2P BitTorrent network can be represented as points spread on a unidimensional space where a distance metric is given by the corresponding network communication latency. Therefore, for each node $n^i \in N$, the set of the nodes, it is possible to define a function $\delta : N \times N \to \mathbb{R}$ such that

$$\delta(n^i, n^j) = \delta^{ij}, \quad \forall\, n^i, n^j \in N, \tag{1}$$

where δ^{ij} is the amount of time taken to bring a small amount of data (e.g., as for a ping) from n^i to n^j. By using the given definition of distance, for each node n^i, it is possible to obtain an ordered list Ω^i so that

$$\Omega^i = \left\{ n^i_\alpha \in N \right\}_{\alpha=0}^{|N|} : \delta(n^i, n^i_\alpha) \leq \delta(n^i, n^i_{\alpha+1}). \tag{2}$$

In such a way, the first item of the list will be $n^i_0 = n^i$ and the following items will be ordered according to their network latency as measured by n^i. Using this complete ordering of peers, it is possible to introduce the concept of content permeability and diffusion.

The adopted mathematical model will be defined in a continuous set by means of a variable δ indicating the distance between two points. In order to represent the BitTorrent network, we need to associate one point to one *peer* (or *node*) of the network, and to obtain such a map we implement a discrete interpretation of this mathematical model. Therefore, while the following model will be developed as a continuous model, starting from Section 3.3 we will make use only of several discrete points, each mapping the nodes of the network, and the model will allow us to obtain their distance as δ^{ij}. Therefore, for each node $n_j \in N$, there exists a point j in our discrete set so that it will be possible to define a *discrete distance* $\delta^{ij} \;\; \forall\, n_i, n_j \in N$, while the points of the continuous model lacking a correspondent real node of the network will be ignored.

The motivation for having a continuous model to start with is evident when considering how users share files on a P2P system: each file consists of several fragments, so sharing fragments can be seen as a diffusion phenomenon. For this reason, we model fragment spreading in terms of Fick's diffusion law, which is described in the following.

3.2. Fick's diffusion law and its use for P2P. Fick's second law is commonly used in physics and chemistry to describe the change of concentration per unit time of some element diffusing into another. Using both the first and second Fick laws, the diffusion of a content into a mean is given as the solution of the vector differential equation

$$\frac{\partial \Phi}{\partial t} = \nabla \cdot (D \nabla \Phi), \tag{3}$$

where Φ is the concentration, t the time and D the permeability to the content. Since this is a separable equation and we make use of a 1-dimensional metric based on the distance δ, and assuming D as constant among the nodes, Eqn. (3) can be written as a scalar differential equation,

$$\frac{\partial \Phi}{\partial t} = D \frac{\partial^2 \Phi}{\partial \delta^2}. \tag{4}$$

The partial differential equation (4), given initial and boundary conditions, admits at least a solution known as Green's function, which describes how a single point of a probability density (in this case, initially at $\delta = 0$) evolves in time and space. Thus the evolution of the system from any initial condition can be found simply by adding up the right amount of probability density at the right points in space, given by

$$G(\delta, t) = \frac{1}{2\pi} \int e^{-D\xi^2 t} e^{-i\xi^2 \delta} \mathrm{d}\xi. \tag{5}$$

It suffices to find a particular normalised solution, so that

$$\int G(\delta, t)\, \mathrm{d}\delta = 1. \tag{6}$$

In order to find an appropriate solution for the problem of fragment spreading through the BitTorrent network, it is possible to apply the infinite-source diffusion boundary conditions and initial conditions. The resulting particular solution can then be written as

$$G(\delta, t) = \frac{1}{\sqrt{4\pi D t}}\, e^{-\frac{\delta^2}{4Dt}}. \tag{7}$$

The Green function found permits us to study the diffusion dynamics of a single content and, as a matter of facts, it can be rewritten as a solution of Eqn. (4) in the form

$$\Phi(\delta, t) = \Phi_0 \Gamma\Big(\frac{\delta}{\sqrt{4Dt}}\Big) \;, \quad \Phi_0 = \frac{1}{\sqrt{4\pi D t}}, \tag{8}$$

where Γ is the complementary Gaussian error function,

$$\Gamma(x) = 1 - \frac{2}{\sqrt{\pi}} \int_0^x e^{-\xi^2}\, \mathrm{d}\xi, \quad \forall\, x \in \mathbb{R}^+. \tag{9}$$

Equation (9) can be computed as successive iterations from a Taylor series,

$$\Gamma(x) = 1 - \frac{2}{\sqrt{\pi}} \sum_{j=0}^{\infty} \frac{x}{2j+1} \prod_{k=1}^{j} \frac{-x^2}{k}, \quad \forall\, x \in \mathbb{R}^+. \tag{10}$$

In the work of Chiani *et al.* (2003), a pure exponential approximation for Eqn. (10) has been proposed in which, within an error of the order of 10^{-9}, $\Gamma(x)$ is calculated as

$$\Gamma(x) \approx \frac{1}{6} e^{-x^2} + \frac{1}{2} e^{-\frac{4}{3}x^2}, \quad \forall\, x \in \mathbb{R}^+. \tag{11}$$

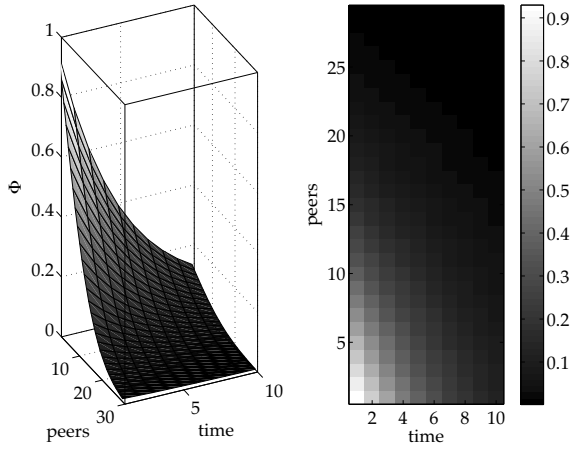


Fig. 2. Approximation of the concentration Φ as in (12), computed on a set of points having distances δ^{ij}.

Using (11) in (8), it eventually follows that

$$\begin{cases} \Phi(\delta,t) \approx \Phi_0 \left[\frac{1}{6} e^{(\Phi_0 \delta)^2} + \frac{1}{2} e^{-\frac{4}{3}(\Phi_0 \delta)^2}\right], \\ \Phi_0 = (4\pi D t)^{-\frac{1}{2}} \end{cases} \tag{12}$$

for every node at a certain distance $\delta \in \mathbb{R}^+$ at a time $t \in \mathbb{R}^+$. Figure 2 shows a representation of concentration Φ, which can be computed for file fragments, as given by (12), while varying the amount of peers and over time.

3.3. From concentration to probability. In Eqn. (12), the scaling factor Φ_0 is a function of the time t. On the other hand, the formalism used was developed mainly to focus on the distance δ, and handling t merely as a parameter. The above mathematical formalism is valid as long as the distances $\delta(n^i, n^j)$ remain time-invariant. The common practice considers the distance between nodes δ as time-invariant; however, the actual network latencies vary (almost) continuously, with time, and a stationary Ω^i ordered set is a very unlikely approximation for the network. In our solution, we make the latency time-dependent. In turn, this makes it possible to choose a different fragment to be shared over time.

For the P2P system, Eqn. (12) states that a certain file fragment z_k^i in a node n^i at a time t_0 has a probability $P_k^{ij}(t_0,t)$ to be given (or diffused) to node n^j, at a distance $\delta^{ij}(t_0)$ from n^i, within a time t, which is proportional to $\Phi(\delta,t)$ so that

$$P_k^{ij}(t_0,t) = p_k^{ij}\left[\frac{1}{6} e^{\left(p_k^{ij}\ \delta^{ij}\right)^2} + \frac{1}{2} e^{-\frac{4}{3}\left(p_k^{ij}\ \delta^{ij}\right)^2}\right], \tag{13}$$

where $p_k^{ij} = p_k^{ij}(t_0,t)$, i.e., it depends on time t_0 and t and carries both the diffusion factors and the temporal dynamics. Since we are interested in a simple proportion, and not a direct equation, we can also neglect the factor 4π and then write p_k^{ij} in the normalised form,

$$p_k^{ij}(t_0,t) = \frac{1}{\sqrt{4\pi}} \cdot \frac{1}{\sqrt{D_k(t_0)}} \cdot \frac{1}{\sqrt{t}}. \tag{14}$$

It is now important to have a proper redefinition of the coefficient D. Let us say that T_k is the number of users interested in file fragment z_k (whether asking or offering it), S_k is the number of seeds for the file fragment and ρ_k is the mean share ratio of the file fragment among peers (including leeches). Then it is possible to consider the urge to share the resource as an osmotic pressure which, during time, varies the permeability coefficient of the network D. In order to take into account the mutable state in a P2P system, D should vary according to the amount of available nodes and file fragments. We have chosen to define

$$D_k(t_0) \triangleq \frac{T_k(t_0)}{S_k(t_0) + [T_k(t_0) - S_k(t_0)]\,\rho_k(t_0)}. \tag{15}$$

Then, by formally substituting D with D_k in Φ_0 in Eqn. (12), we obtain the analytical form of the term p_k^{ij}.

3.4. Discrete time evolution on each node. Indeed, the physical nature of the adopted law works in the entire variable space; however, for the problem at hand, discrete-time simplifications are needed. Let us suppose that for a given discrete time step $\tau = 0$ node n^i effectively measures the network latencies of a set of nodes $\{n^j\}$; then, an ordered set Ω^i as in Eqn. (2) is computed. Now, for every node n^i, probability P_k^{ij} is computed for each of its own file fragment z_k and for every node n^j. This probability corresponds to a statistical *prevision* of the *possible* file fragments spreading onto other nodes.

Suppose that for a while no more measures for δ have been taken; at a later discrete time step τ, file fragment z_k^i will be copied to the first node to be served, which is chosen according to the minimum probability of diffusion, latencies and time since the last measures were taken (see the following subsection and Eqn. (18)). Then, such a file fragment is reaching other nodes if the latency for such nodes is less than time t_k^i, computed as

$$t_k^i(\tau) = \sum_{\alpha_k=0}^{\tau} \delta(n^i, n_\alpha^i). \tag{16}$$

Index k is used in Eqn. (16) to refer to file fragment z_k^i.

Indeed, it should be highlighted that since nodes need and offer their own file fragments, the ordered set of nodes referred by a given node should depend on resource z_k, i.e., $\Omega_k^i = \{n_{\alpha_k}^i\}$.

It is now possible to have a complete mapping of the probability of diffusion by reducing the time dependence

from (t_0, t) to a single variable dependence from the discrete time-step τ. For each resource z_k, as $P_k^{ij}(\tau)$ stated, it is possible to reduce $D_k(t_0, t)$ to a one-variable function $D_k(\tau)$ by assuming that at t_0 we have $\tau = 0$ and considering only the values of $D_k(t_0, t)$ when t is the execution moment of a computational step τ.

3.5. Assigning priorities and corrections. Once all $P_k^{ij}(\tau)$ have been computed and its values stored into a proper data structure, it is actually simple to determine the most urgent file fragment to share, which is the resource that has the least probability to be spread, i.e., the k for which $P_k^{ij}(\tau)$ is minimum.

Furthermore, we should consider that, over time, an old measured δ differs from the actual value, and hence the measure becomes less reliable. To take into account the staleness of δ values, we gradually consider the choice of a fragment, less bound to δ, and this behaviour is provided by the negative exponential in Eqn. (17). Given enough time, the choice will be based only on the number of available fragments. However, we consider that by that time a new measure for δ would have been taken and incorporated again into the model choosing the fragment.

Generally, for nodes having the highest latencies with respect to a given node n^i, more time will be needed to receive a fragment from the node n^i. We aim at compensating such a delay by incorporating into our model the inescapable latencies of a P2P network. Therefore, the node that will receive a fragment first will be among the furthest. For the model, we have then chosen a decay law. Now it is possible to obtain a complete time-variant analytical form of the spreading of file fragments (see Fig. 3) defined as in the following:

$$\chi_k^{ij}(\tau) = \frac{e^{-c\tau\delta^{ij}}}{P_k^{ij}(\tau)}, \tag{17}$$

where the decay constant c can be chosen heuristically, without harming the formulated law, and tuned according to other parameters. If k indicates a file fragment and k^* the index of the most urgent file fragment to share, this latter is trivially found as the solution of a maximum problem so that

$$k^* : \chi_{k^*}^{ij}(\tau) = \max_k \left\{ \chi_k^{ij}(\tau) \right\}. \tag{18}$$

Figure 3 shows the decay of several computed χ values for different peers requiring a file fragment z_3 (3 is the fragment index). Of course, all the priorities depend on the value of the two-dimensional matrix of values of P_k^{ij} (we mark that the index i does not change within the same node n^i). Among these values, there is no need to compute elements where $j = i$ and for those elements where the node n^j is not in the queue for resource z_k. In both the cases, it is assumed that $P_k^{ij} = 1$. Moreover,

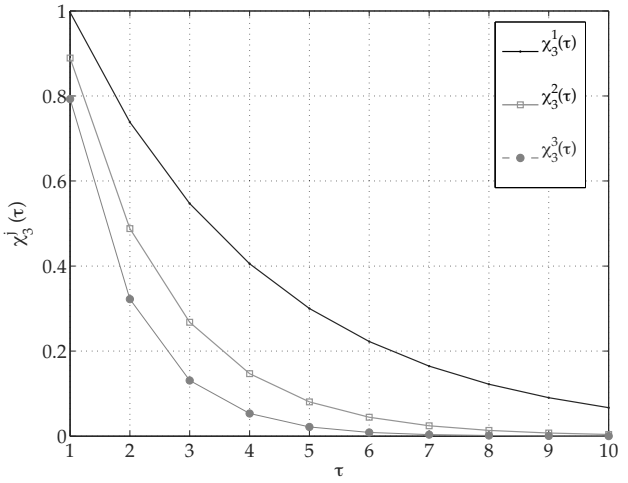


Fig. 3. Time decay of some normalised $\chi_k^j(\tau)$ for increasing time steps τ.

after n^i having completed to transfer z_k to the node n^j, the element of indices (j, k) is set to 1. In a similar fashion, each peer is able to identify a possible resource to ask for in order to maximise the diffusion of rare ones instead of common ones.

4. Multiscale neural predictor to devise availability trends for file fragments

Although the model proposed in Sections 2 and 3 evolves in time as in Eqn. (17), such a model is based on initial conditions, essentially fragment availability, measured at a certain time. On the other hand, it is only when new data are received (e.g., when the tracker of the BitTorrent network sends new information on the state of the network, the number of peers and seeds for the file fragments) that an updated result can be obtained by changing the initial conditions in our mathematical model as well. Therefore, while integrating certain dynamics, the mathematical model alone can neither predict, nor anticipate future network conditions by itself. In order to predict the future state of the BitTorrent network, and then suggest the appropriate priority actions as a consequence, we developed an appropriate predictor which takes advantage of several analysis methods as well as machine learning techniques in order to tamper with the timeline.

Our approach is built on wavelets and neural networks to model the future trends of file fragments availability in a near future.

4.1. Basis of wavelet decomposition. Wavelet decomposition is a powerful analysis tool for physical and dynamic phenomena that reduces the data redundancies

and yields a compact representation expressing the intrinsic structure of a phenomenon. In fact, the main advantage when using wavelet decomposition is the ability to pack the energy signature of a signal or a time series, and then to express relevant data as a few non-zero coefficients. This characteristic has been proven very useful to optimise the performances of neural networks (Gupta *et al.*, 2004).

Like sine and cosine for Fourier transforms, wavelet decomposition uses functions, i.e., wavelets, to express a function as a particular expansion of coefficients in the wavelet domain. Once a mother wavelet has been chosen, it is possible, as explained in the following, to create new wavelets by dilates and shifts of the mother wavelet. Such newly generated wavelets, if chosen with certain criteria, eventually form a Riesz basis of the Hilbert space $L^2(\mathbb{R})$ of square integrable functions. Such criteria are at the basis of *wavelet theory* and come from the concept of multiresolution analysis of a signal, also called multiscale approximation. When a dynamic model can be expressed as a time-dependent signal, i.e., described by a function in $L^2(\mathbb{R})$, it is possible to obtain a multiresolution analysis of such a signal. For the space $L^2(\mathbb{R})$, such an approximation consists of an increasing sequence of closed subspaces which approximate, with a greater amount of details, the space $L^2(\mathbb{R})$, eventually reaching a complete representation of $L^2(\mathbb{R})$ itself. A complete description of multiresolution analysis and the relation with wavelet theory can be found in the work of Mallat (2009).

One-dimensional decomposition wavelets of order n for a signal $s(t)$ give a new representation of the signal itself in an n-dimensional multiresolution domain of coefficients plus a certain residual coarse representation of the signal in time. For any discrete time step τ, then, the corresponding M order wavelet decomposition $\hat{\mathbf{W}}s(\tau)$ of the signal $s(\tau)$ will be given by the vector

$$\hat{\mathbf{W}}s(\tau) = [d_1(\tau), d_2(\tau), \ldots, d_M(\tau), a_M(\tau)], \quad (19)$$

where d_1 is the most detailed multiresolution approximation of the series and d_M the least detailed, and a_M is the residual signal.

These coefficients are computed by means of successive iterations and by recursively applying a bank of wavelet filters to the signal and its residuals (the nature of such filters will be clear in the following). The resulting coefficients are able to express intrinsic time-energy features of the signal, i.e., features of a time series, while removing redundancies and offering a well-suited representation, we give as inputs for a neural network.

It is now possible to give a more rigorous definition of a wavelet. Let us take into account a multiresolution decomposition of $L^2(\mathbb{R})$,

$$\emptyset \subset V_0 \subset \ldots \subset V_j \subset V_{j+1} \subset \ldots \subset L^2(\mathbb{R}).$$

If we call W_j the orthogonal complement V_j, then it is possible to define a wavelet as a function $\psi(x)$ if the set of $\{\psi(x-l)|l \in \mathbb{Z}\}$ is a Riesz basis of W_0 and also meets the following two constraints:

$$\int_{-\infty}^{+\infty} \psi(x)\,\mathrm{d}x = 0 \quad (20)$$

and

$$||\psi(x)||^2 = \int_{-\infty}^{+\infty} \psi(x)\psi^*(x)\,\mathrm{d}x = 1.$$

If the wavelet is also an element of V_0, then there exists a sequence $\{g_k\}$ such that

$$\psi(x) = 2\sum_{k\in\mathbb{Z}} g_k \psi(2x-l).$$

Then the set of functions $\{\psi_{j,l}|j,l \in \mathbb{Z}\}$ is a Riesz basis of $L^2(\mathbb{R})$. It follows that a wavelet function can be used to define a Hilbert basis, which is a complete system, for the Hilbert space $L^2(\mathbb{R})$. In this case, the Hilbert basis is constructed as the family of functions $\{\psi_{j,l}|j,l \in \mathbb{Z}\}$ by means of dilation and translation of a mother wavelet function ψ so that $\psi_{j,l} = \sqrt{2^j}\psi(2^j x - l)$. Hence, given a function $f \in L^2(\mathbb{R})$, it is possible to obtain the following decomposition:

$$f(x) = \sum_{j,l\in\mathbb{Z}} \langle f|\psi_{j,l}\rangle = \sum_{j,l\in\mathbb{Z}} d_{j,l}\psi_{j,l}(x), \quad (21)$$

where $d_{j,l}$ are called wavelet coefficients of the given function f in the wavelet basis given by the inner product of $\psi_{j,l}$. Likewise, a projection on the space V_j is given by

$$\mathbb{P}_j f(x) = \sum_i \langle f|\varphi_{i,j}\rangle\varphi_{i,j}(x),$$

where $\varphi_{i,j}$ are called dual scaling functions. When the basis wavelet functions coincide with their duals, the basis is orthogonal. Choosing a wavelet basis for the multiresolution analysis corresponds to selecting the dilation and shift coefficients. In this way, by performing the decomposition, we obtain the $\{d_i|a_M\}$ coefficients sets of (19).

For the present work, we adopted biorthogonal wavelet decomposition (this wavelet family is described by Mallat (2009)), for which symmetrical decomposition and exact reconstruction are possible with finite impulse response (FIR) filters (Rabiner and Gold, 1975). Figure 4 shows the implemented biorthogonal wavelet functions and the related filter coefficients.

An accurate study has shown that biorthogonal wavelet decomposition optimally approximates and

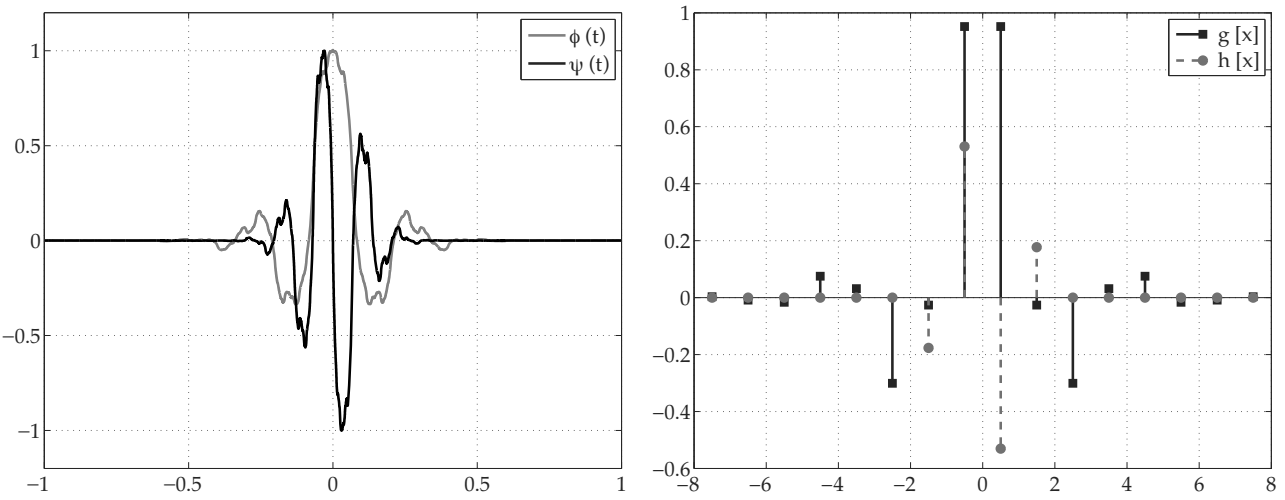


Fig. 4. Implemented biorthogonal wavelets (left) and the related wavelet filter (right).

denoises the time series under analysis. Such a wavelet family is in good agreement with previous optimal results obtained by the authors for the decomposition of other phenomena. In fact, such a decomposition splits a phenomenon in a superposition of mutual and concurrent predominant processes with a characteristic time-energy signature. For stochastically-driven processes, such as stellar phenomena (Capizzi *et al.*, 2012; Napoli *et al.*, 2010) or renewable energy and system load (Bonanno *et al.*, 2012b; 2012a), and for a large category of complex and distributed systems, wavelet decomposition gives a unique and compact representation of the leading features for a time-variant phenomenon.

Then, the datasets regarding the time series of the number of peers and seeds were decomposed by using wavelet biorthogonal decomposition identified by the couple of numbers 3.7, i.e., implemented by using FIR filters with the 7th order polynomials degree for the decomposition and the 3rd order for the reconstruction (the filter coefficients are depicted in Fig. 4) .

4.2. Wavelets and neural networks. A neural network can be built to perform such a construction, i.e., a neural network would act as an inverse second generation wavelet transform. In the work of Bonanno *et al.* (2014), a neural network with a rich representation of past outputs like a fully connected recurrent neural network (RNN), known as the Williams–Zipser network or the nonlinear autoregressive network with exogenous inputs (NARX) (Williams, 1989), has been proven able to generalise as well as structure itself to behave as an optimal discrete wavelet filter. Moreover, for such a kind of RNNs, when applied to the prediction and modelling of stochastic phenomena, like the analysed behaviour of users, which lead to a variable number of access requests in time, real time recurrent learning (RTRL) has been proven to be very effective. A complete description of the RTRL algorithm, NARX and RNNs can be found in the work of Williams and Zipser (1989) or Haykin (2009).

RTRL has been used to train the RNN, and such a trained RNN achieves the ability to perform lifting stages, hence the matching of the time series dynamics at the corresponding wavelet scale. This construction brings the possibility to match non-polynomial and nonlinear signal structures in an optimised straightforward N-dimensional mean square problem (Mandic and Chambers, 2001). NARX networks have been proven able to use the intrinsic features of time series in order to predict the following values of the series (Capizzi *et al.*, 2012). One class of transfer functions for the RNN has to be chosen to approximate the input-output behaviour in the most appropriate manner. For phenomena having deterministic dynamic behaviour, the relative time series at a given time point can be modelled as a functional of a certain amount of previous time steps. In such cases, the model used should have some internal memory to store and update context information (Lapedes and Farber, 1986). This is achieved by feeding the RNN with a delayed version of past data, commonly referred to as time delayed inputs (Connor *et al.*, 1994).

4.3. Proposed multiscale neural predictor. As stated in Section 4.2, it would be desirable to have a neural network able to predict the future evolution of the availability of file fragments while also performing the wavelet inverse transform. The first property is a common characteristic of neural networks, since such solutions are universal approximators, as demonstrated by Cybenko (1989). For the latter property, we could use a mother wavelet as the transfer function; however, mother wavelets

lack some elementary properties needed by a proper transfer function such as, e.g., the absence of local minima and a sufficiently graded and scaled response (Gupta *et al.*, 2004). This leads us to look for a close enough substitute to approximate the properties of a mother wavelet without affecting the functionalities of the network itself. The function classes that more closely approximate a mother waveform have to be found among radial basis functions (RBFs), which are good enough as transfer functions and partially approximate half of a mother waveform. It is indeed possible to properly scale and shift a couple of RBFs to obtain a mother wavelet. If we define an RBF function as $f : [-1, 1] \to \mathbb{R}$, then we could dilate and scale it to obtain a new function,

$$\tilde{f}(x+2l) = \begin{cases} +f(2x+1), & x \in [-1, 0), \\ -f(2x-1), & x \in (0, +1], \end{cases} \tag{22}$$

$\forall\ l \in \mathbb{Z}$. With such a definition, starting from the properties of the RBF, it is then possible to verify the following:

$$\int_{2h+1}^{2k+1} \tilde{f}(x)\,\mathrm{d}x = 0, \quad \forall\,(h,k) \in \mathbb{Z}^2 : h < k. \tag{23}$$

Starting from (22) and (23), it is possible to verify Eqns. (20) and (21) for the chosen $\tilde{f}$, which we can now call a mother wavelet. The chosen mother wavelet is a composition of two RBF transfer functions that are realised by the proposed neural network to obtain the properties of a wavelet transform. The proposed RNN has two hidden layers with an RBF transfer function.

For this work, the initial dataset was a time series representing the past values of χ_k^{ij} in Eqn. (17). For more practical notation, we indicate such a time series as $x(\tau)$, where τ is the discrete time step of the data, sampled with a fixed ratio. A biorthogonal wavelet decomposition of the time series has been computed to obtain the correct input set for the RNN as required by the devised architecture. This decomposition has been achieved by applying the wavelet transform as a recursive couple of conjugate filters (see Fig. 4) in such a way that the i-esime recursion $\hat{W}_i$ produces, for any time step of the series, a set of coefficients d_i and residuals a_i, so that

$$\hat{W}_i[a_{i-1}(\tau)] = [d_i(\tau), a_i(\tau)], \quad \forall\, i \in [1, M] \cap \mathbb{N}, \tag{24}$$

where we intend $a_0(\tau) = x(\tau)$. The input set can then be represented as a $T \times (M+1)$ matrix of T time steps of an M level wavelet decomposition, where the τ-esime row represents the τ-esime time step as the decomposition

$$\mathbf{u}(\tau) = [d_1(\tau), d_2(\tau), \ldots, d_M(\tau), a_M(\tau)]\,. \tag{25}$$

Each row of this dataset is given as the input value to the M input neurons of the proposed RNN. The properties of this network (Napoli *et al.*, 2013) make it possible, starting from an input at a time step τ_n, to predict how rare fragments will be at a time step τ_{n+r}. In this way, the RNN acts like a functional

$$\hat{N}[\mathbf{u}(\tau_n)] = x(\tau_{n+r}), \tag{26}$$

where r is the number of time steps of forecast in the future. Figure 5 depicts a model of the RNN architecture developed in this work.

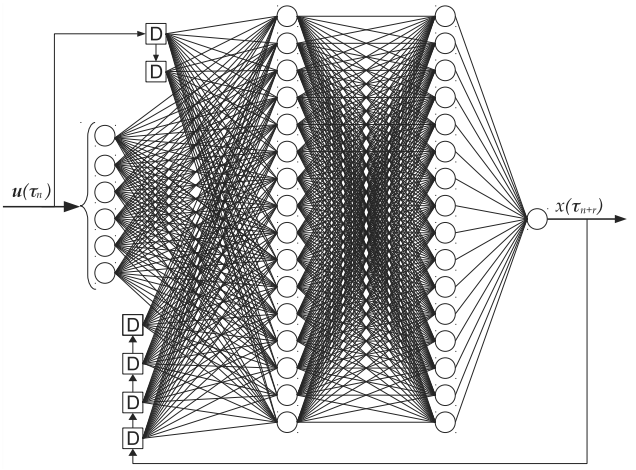


Fig. 5. Selected recurrent neural network architecture.

5. Setup of the neural predictor

For the problem at hand, a five-level wavelet decomposition has been selected that properly characterises the data under analysis. Therefore, the devised RNN (see Fig. 5) uses a six-neuron input layer (one for each level detail coefficient d_i and one for the residual a_5). This RNN architecture presents two hidden layers with sixteen neurons each and realises an RBF (as explained in Section 4.3).

Inputs are given to the RNN in the following form:

- the wavelet decomposition of the time series $\mathbf{u}(\tau_n)$ for time step τ_n,
- the previous delayed decompositions $\mathbf{u}(\tau_{n-1})$ and $\mathbf{u}(\tau_{n-2})$,
- the last four delayed outputs $x(\tau_{n+r})$ predicted by the RNN.

Delays and feedback are obtained by using the relative delay lines and operators (D). These feedback lines provide the RNN with internal memory, hence the modelling abilities for dynamic phenomena. For the case study proposed in this paper, we have used several different time series containing raw data coming from the BitTorrent network, specifically for each shared file: (i)

the number of peers, (ii) the number of seeds, and (iii) the sharing ratio.

We consider the time series complete (with no missing information or data gaps) since the delivery of the series is the responsibility of the tracker and since the BitTorrent protocol requires to periodically negotiate with the tracker. On the other hand, in the BitTorrent network, such values are given on a file-related basis; in fact, we have that a file is a set of fragments. Therefore, for the l-esime shared file, represented as K_l, at a time t_0 the raw values given by the BitTorrent tracker correspond to vector ξ_1,

$$\xi_1 = \begin{pmatrix} T_k(t_0) \\ S_k(t_0) \\ \rho_k(t_0) \end{pmatrix}, \quad \forall\, k : z_k \in K_l, \tag{27}$$

where $T_k(t_0)$, $S_k(t_0)$ and $\rho_k(t_0)$ are the ones used in Eqn. (15). This means that at time t_0 we can compute $D_k(t_0)$ and, consequently, Φ from Eqn. (12), as well as $\chi_k^{ij}(t_0)$ from Eqn. (17). That is, for each time step τ we indirectly obtain $\chi_k^{ij}(\tau)$ from the data given by the tracker and then using our mathematical model.

As in Eqn. (17), we note that i and j are the indices of the nodes in the BitTorrent network, and k represents a file fragment. Once the time series of values χ_k^{ij} has been obtained, we want to predict the future availability of each k-esime file fragment. Therefore, the above developed RNN predictor has been trained for each shared file (not just for a file fragment, since the time series to be fed to the RNN are the same for each fragment belonging to the same file). Moreover, given the definition of a file as a set of fragments, it follows that

$$K_{l_1} \cap K_{l_2} = \emptyset, \quad \forall\, l_1 \neq l_2. \tag{28}$$

Therefore, for L shared files we would have L neural networks (each one associated to a file) to obtain L predictions of the parameter vectors in Eqn. (27). Then, since files are shared among nodes, the results of the predictions referring a file are spread to the corresponding nodes. The L trained networks have all the same topology, hence we need to store the trained weight matrices only, in case of a restart. Then, each node n^i uses a subset of all predictions, i.e., the ones related to the files the node has got (see Fig. 6).

The employed RNNs were trained by using a gradient descent back-propagation algorithm with a momentum led adaptive learning rate as presented by Haykin (2009). For a prediction of 2 hours in advance of the time series, the relative error was less than 6%. The output of the RNNs is the selected file fragment ids that have to be sent first.

5.1. Predicted file availability. By considering both the predicted $\tilde{x}_k(\tau_{n+r})$ and the modeled $\chi_k(\tau_{n+r})$, it is possible, at a time step τ_n, to take counteracting actions and improve the availability estimated for a future time τ_{n+r}, hence increasing the diffusion of rare file fragments. This is achieved, in practice, by using altered values for $D_k(\tau_{n+r})$, which account for the forecast of future time steps. Such modified values are computed by our RNNs, and then predicted future values for $T_k(\tau_{n+r})$, $S_k(\tau_{n+r})$ and $\rho_k(\tau_{n+r})$ are sent to each node active as a peer.

Each time a new file becomes shared on the P2P BitTorrent network, a new RNN is created and trained on a server (e.g., requested from a cloud system (Borowik *et al.*, 2015; Napoli *et al.*, 2016)), in order to provide predictions related to peer availability of the novel set of shared fragments. Values indicating the prediction are sent to the peers periodically, and allow peers to update their values of $D_k(\tau)$. The update frequency can be tuned in order to correctly match the dynamic of peers.

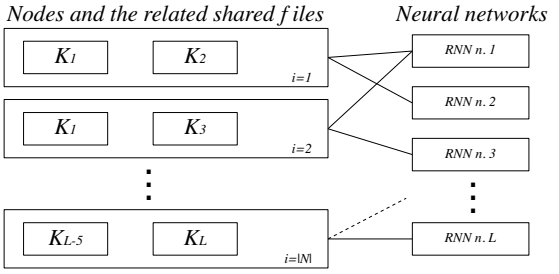


Fig. 6. Employed associative topology among neural networks and files.

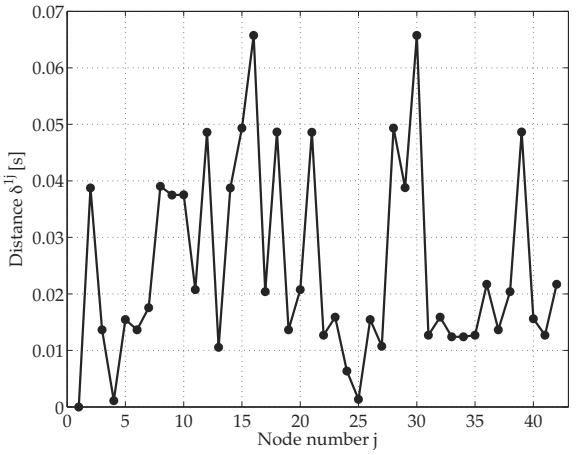


Fig. 7. Distances measured by node n^1 with respect to all the 42 nodes available in the experiment.

6. Experiments

Figure 7 shows the measured distances of the available nodes measured by the first node ($i = 1$). For our experiments we used a mixture of hosts connected by the Italian research and education fast network (GARR). The

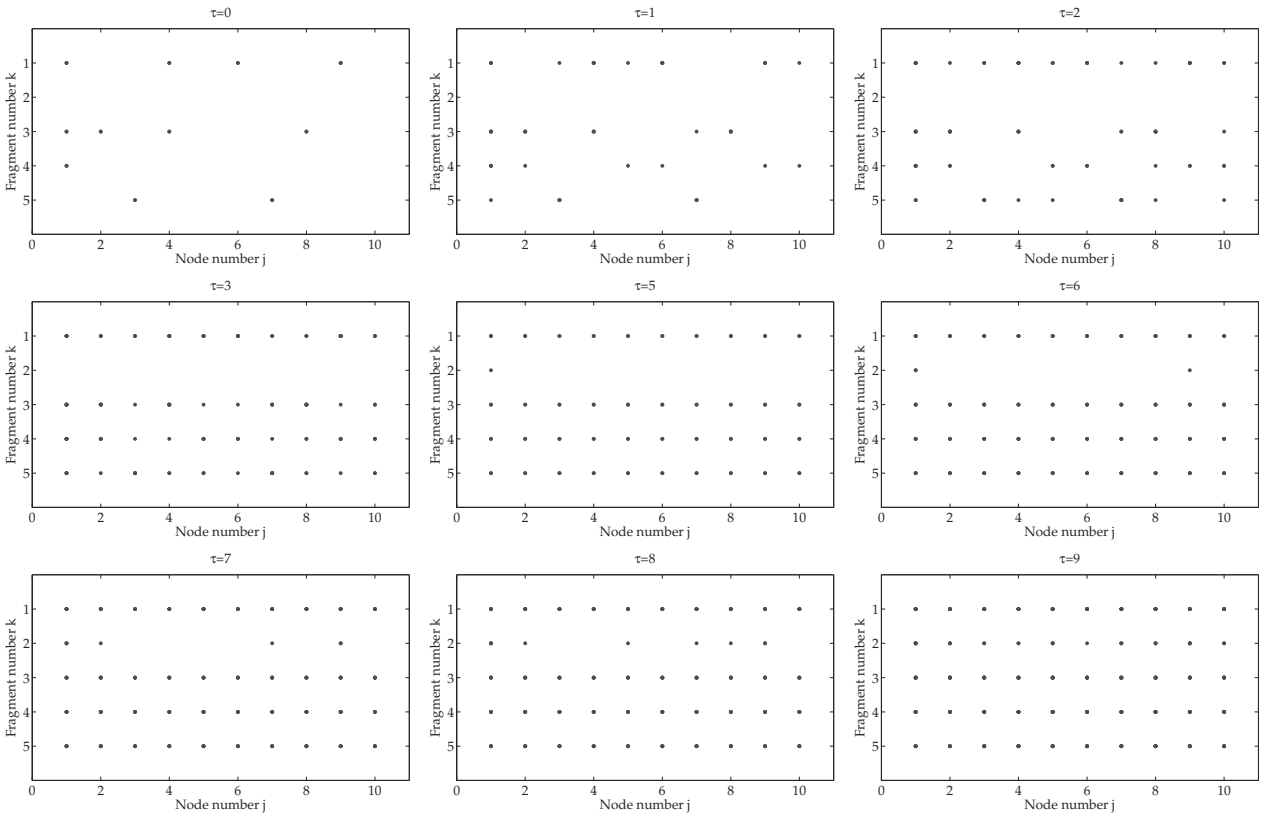


Fig. 8. Evolution of a subnet composed of 10 nodes sharing 4 different file fragments (since z_2 is missing). At a time step $\tau = 4$ the fifth file fragment (z_2) is injected on node n^i and then spread all over.

simulated BitTorrent network comprised 42 nodes sharing 5 files.

For the sake of clarity, we also simulated a subnetwork of 10 nodes sharing 5 file fragments (see Fig. 8). In the latter example, at the initial condition of the system, four of the file fragments happen to be heterogeneously spread among peers of the P2P network, while a fifth fragment (namely, z_2) is not present within the connected nodes. In the order, step after step, each node selected a file fragment to require and a file fragment to send, e.g., at the time step $\tau = 1$ the node n^1 tried to send file fragment z_4 to as many nodes as possible because of its urgency (since it is the rarest fragment) starting from n^2 (since it is the farthest node from n^0). Simultaneously, the nodes n^2, n^3, n^6, n^7, n^8 and n^9 sent the only fragment they had at $\tau = 0$. Since both z_1 and z_3 are equally rare, the node n^4 at $\tau = 0$ sent these two fragments on a node distance-basis (the furthest the first).

At a successive time step ($\tau = 1$), the situation seems to change radically because of the fragments that have been just transferred among nodes. In this simulation, all fragments, except z_2 because it is actually unavailable on any node, have been shared among nodes, in a very low number of time steps. It should be pointed out that from $\tau = 1$ to $\tau = 3$ some previously rare fragments have been rapidly spread and that only later on the most common fragments will be transferred. At $\tau = 3$ the system of peers seems to reach a steady situation: all fragments have been shared, except fragment z_2, since it is unavailable, hence all the nodes are waiting for it.

Let us now suppose that, during the time step $\tau = 4$, an eleventh node (additional to the previous network of peers) transfers z_2 to n^1; the result is then depicted in the scenario at $\tau = 5$. In this second part of the experiment, while the rarity of z_2 is not important, then only the distance of the nodes leads to the order of distribution. For example, when $5 \leq \tau \leq 6$, node n^1 sends the file to n^9, which is the most distant node with respect to n^1. The same strategy is then adopted by other nodes receiving it until the fragment has been shared with all nodes ($\tau = 9$). The described behaviour has been determined by the model in Eqn. (18).

Moreover, the evolution shown does not consider the file fragments that could have been passed among the nodes in between two different updates, and so that for each step the value of χ for n^{10} would drop to zero (the highest values of χ are an indication of the urgency of receiving a fragment).

The described model and formula allow subsequent sharing activities, after the initial time steps, to be

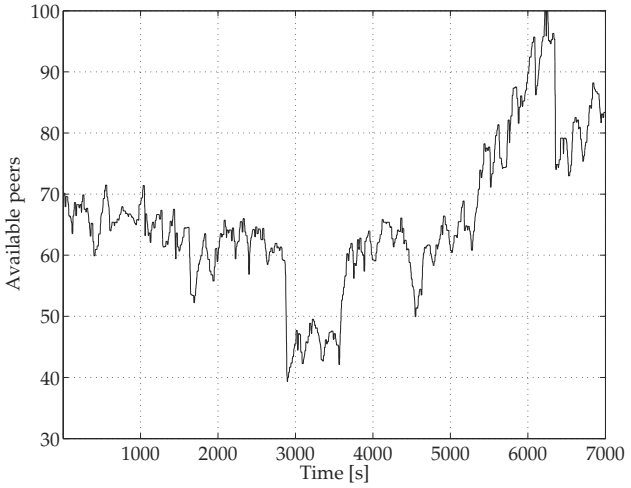


Fig. 9. Measured node availability.

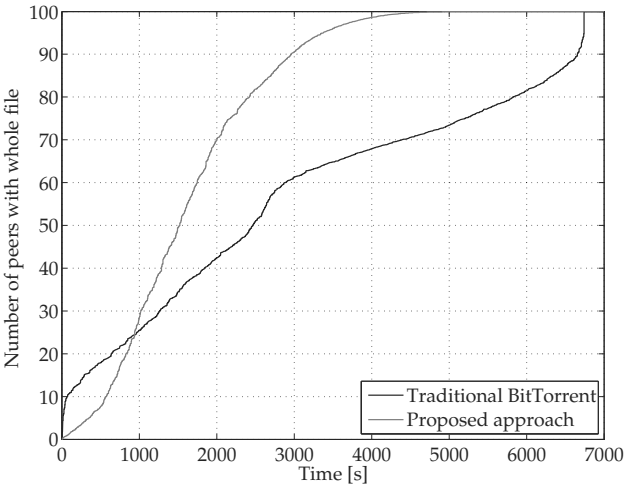


Fig. 10. Performances of the proposed approach compared with a traditional BitTorrent network for a 1 GB file shared among 100 nodes.

determined in terms of which fragments should be sent. In the long run, this law will privilege the near nodes, while in the short term, distant nodes are often the ones having higher priority.

A more extensive comparison was performed by simulating both our approach and the standard BitTorrent protocol. We wanted to share a file of size 1 GB among 100 peers, therefore sharing 65536 file fragments, each of size 16 KB. In our initial conditions there was only one seed (i.e., a node with all the fragments), while each of the other peers was provided with one file fragment (a different fragment for each peer, therefore multiple replicas of the fragments were on the network). We decided to start with this setup in order to simplify the comparisons of the results excluding the transient phase (i.e., when only one seed begins to share a file with peers that are not yet able to share the file). Finally, we supposed that each peer could send one fragment and receive five fragments at the same time. For the simulations we used network latencies and nodes availability from real data: we measured the latencies in our network (a partial amount of data is given in Fig. 7), while we applied a scaled profile of real peers availability on the traditional BitTorrent network (see Fig. 9).

The resulting comparison is shown in Fig. 10: while our approach has a slow start (since it prefers to diffuse replicas to remote peers instead of giving them to the nearest peers), it definitively prevails over the standard BitTorrent protocol due to the said ability to quickly adapt to the number of replicas and peers available.

7. Related works

Several studies have analysed the behaviour of BitTorrent systems from the point of view of fairness, i.e., how to have users contribute with contents that can be uploaded by other users, levelling the amount of downloads with that of uploads. Fewer works have studied the problem of unavailability of contents in P2P BitTorrent networks, while the main focus has often been on the appropriate ranking systems that give priorities to peers or moderate the interactions between them (Visan *et al.*, 2011). For networks consisting of a large number of nodes, some priority management systems are based on scalable algorithms that ensure rapid convergence, such as Epidemic-style or gossip-based algorithms as in the work of Ghit *et al.* (2010).

Another approach is that of Qiu and Srikant (2004), who propose to rank peers according to their upload bandwidth; hence, when having to provide some contents, the selection of peers is performed accordingly. One of the mechanisms proposed to increase file availability has been to use a multi-torrent, i.e., for ensuring fairness, instead of forcing users to stay longer, contribution is provided to uploaders for fragments belonging to different files (Guo *et al.*, 2005). Similarly, Kaune *et al.* (2010) show that, by using the multi-torrent, availability can be easily increased, and confirm that fast replication of rare fragments is essential. Furthermore, bundling, i.e., the dissemination of a number of related files together, has been proposed to increase availability (Menasche *et al.*, 2009).

The above proposed mechanisms differ from our proposal, since we take into account several novel factors: the dynamic of data exchange between distant peers, a decay for the availability of peers, and the forecast of contents availability. Such factors have been related to a proposed model, which manages to select the rarest

content to be spread, taking into account the future availability and the peers that should provide and take such a content.

8. Conclusions

This paper has proposed a solution that improves the availability of fragments on a P2P BitTorrent system by adopting a mathematical model and a neural network, each properly devised for the problem at hand. The model is able to precisely describe diffusion of fragments and the urgency to share fragments, thanks to the mapping that we have proposed of mass diffusion through a porous means and the derived equations. The neural network approximates the availability of peers, and hence fragments, at later time points, by retaining the characteristics of the behaviour of users. This has been achieved firstly by wavelet-transforming of the time series of peer availability, and secondly by feeding such results to a nonlinear autoregressive neural network, which is able to both perform predictions and apply an anti-wavelet transform. By using the estimates of future fragments availability provided by our neural network into the fragment diffusion model, we can then select the fragments that have to be quickly spread to counteract their disappearance due to some user disconnection.

The proposed approach can be easily embedded on a P2P BitTorrent system, while preserving modularity and separation of concerns (Bannò *et al.*, 2010; Giunta *et al.*, 2011; Calvagna and Tramontana, 2013; Tramontana, 2013), since the computational cost due to prediction and modelling is essentially up to the tracker itself, hence freeing peers of the burden. This choice would tap into a resource, the tracker, which is an existing component that peers have to connect to. For the computational cost, an instance of our ensemble (predicting the neural network and the fragment diffusion model) suffices to give accurate suggestions for a file and all its fragments, and updates to peers are given at widely spaced time intervals.

References

Bannò, F., Marletta, D., Pappalardo, G. and Tramontana, E. (2010). Tackling consistency issues for runtime updating distributed systems, *Proceedings of the IEEE International Symposium on Parallel & Distributed Processing, Workshops and PhD Forum (IPDPSW), Atlanta, GA, USA*, pp. 1–8.

Bonanno, F., Capizzi, G., Coco, S., Napoli, C., Laudani, A. and Lo Sciuto, G. (2014). Optimal thicknesses determination in a multilayer structure to improve the SPP efficiency for photovoltaic devices by an hybrid FEM—cascade neural network based approach, *Proceedings of the IEEE International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), Ischia, Italy*, pp. 355–362.

Bonanno, F., Capizzi, G., Gagliano, A. and Napoli, C. (2012a). Optimal management of various renewable energy sources by a new forecasting method, *Proceedings of the IEEE International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), Sorrento, Italy*, pp. 934–940.

Bonanno, F., Capizzi, G. and Napoli, C. (2012b). Some remarks on the application of RNN and PRNN for the charge-discharge simulation of advanced lithium-ions battery energy storage, *Proceedings of the IEEE International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), Sorrento, Italy*, pp. 941–945.

Borowik, G., Woźniak, M., Fornaia, A., Giunta, R., Napoli, C., Pappalardo, G. and Tramontana, E. (2015). A software architecture assisting workflow executions on cloud resources, *International Journal of Electronics and Telecommunications* **61**(1): 17–23.

Calvagna, A. and Tramontana, E. (2013). Delivering dependable reusable components by expressing and enforcing design decisions, *Proceedings of the IEEE Computer Software and Applications Conference (COMPSAC) Workshop (QUORS), Kyoto, Japan*, pp. 493–498.

Capizzi, G., Napoli, C. and Paternò, L. (2012). An innovative hybrid neuro-wavelet method for reconstruction of missing data in astronomical photometric surveys, *Proceedings of the International Conference on Artificial Intelligence and Soft Computing (ICAISC), Zakopane, Poland*, pp. 21–29.

Chiani, M., Dardari, D. and Simon, M.K. (2003). New exponential bounds and approximations for the computation of error probability in fading channels, *IEEE Transactions on Wireless Communications* **2**(4): 840–845.

Cohen, B. (2003). Incentives build robustness in BitTorrent, *Workshop on Economics of Peer-to-Peer Systems, Berkeley, CA, USA*, Vol. 6, pp. 68–72.

Cohen, B. (2008). The BitTorrent protocol specification, `http://jonas.nitro.dk/bittorrent/bittorrent-rfc.html`.

Connor, J.T., Martin, R.D. and Atlas, L. (1994). Recurrent neural networks and robust time series prediction, *Transactions on Neural Networks* **5**(2): 240–254.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function, *Mathematics of Control, Signals and Systems* **2**(4): 303–314.

Fornaia, A., Napoli, C., Pappalardo, G. and Tramontana, E. (2015). Using AOP neural networks to infer user behaviours and interests, *XVI Workshop "From Object to Agents" (WOA), Napoli, Italy*, pp. 46–52.

Ghit, B., Pop, F. and Cristea, V. (2010). Epidemic-style global load monitoring in large-scale overlay networks, *Proceedings of the International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), Fukuoka, Japan*, pp. 393–398.

Giunta, R., Pappalardo, G. and Tramontana, E. (2011). Aspects and annotations for controlling the roles application classes

play for design patterns, *Proceedings of the IEEE Asia Pacific Software Engineering Conference (APSEC), Ho Chi Minh, Vietnam*, pp. 306–314.

Guo, L., Chen, S., Xiao, Z., Tan, E., Ding, X. and Zhang, X. (2005). Measurements, analysis, and modeling of BitTorrent-like systems, *Proceedings of the ACM SIGCOMM Conference on Internet Measurement, Berkeley, CA, USA*, pp. 35-48.

Gupta, M.M., Jin, L. and Homma, N. (2004). *Static and Dynamic Neural Networks: From Fundamentals to Advanced Theory*, Wiley-IEEE Press, New York, NY.

Haykin, S. (2009). *Neural Networks and Learning Machines*, Vol. 3, Prentice Hall, New York, NY.

Kaune, S., Rumin, R.C., Tyson, G., Mauthe, A., Guerrero, C. and Steinmetz, R. (2010). Unraveling BitTorrent's file unavailability: Measurements and analysis, *IEEE International Conference on Peer to Peer Computing (IEEE P2P), Delft, The Netherlands*, pp. 1–9.

Lapedes, A. and Farber, R. (1986). A self-optimizing, nonsymmetrical neural net for content addressable memory and pattern recognition, *Physica D: Nonlinear Phenomena* **22**(1): 247–259.

Mallat, S. (2009). *A Wavelet Tour of Signal Processing: The Sparse Way*, Academic Press, Cambridge.

Mandic, D.P. and Chambers, J. (2001). *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability*, John Wiley & Sons, Inc., New York, NY.

Menasche, D.S., Rocha, A.A., Li, B., Towsley, D. and Venkataramani, A. (2009). Content availability and bundling in swarming systems, *Proceedings of the ACM Conference Co-NEXT, Rome, Italy*, pp. 121–132.

Napoli, C., Bonanno, F. and Capizzi, G. (2010). Exploiting solar wind time series correlation with magnetospheric response by using an hybrid neuro-wavelet approach, *Advances in Plasma Astrophysics*: *Proceedings of the International Astronomical Union, Giardini Naxos, Italy*, pp. 156–158.

Napoli, C., Pappalardo, G. and Tramontana, E. (2013). A hybrid neuro-wavelet predictor for QoS control and stability, *in* M. Baldoni *et al.* (Eds.), *Proceedings of Artificial Intelligence (AIxIA)*, Lecture Notes in Computer Science, Vol. 8249, Springer, Berlin pp. 527–538.

Napoli, C., Pappalardo, G. and Tramontana, E. (2014a). An agent-driven semantical identifier using radial basis neural networks and reinforcement learning, *XV Workshop "From Objects to Agents" (WOA), Catania, Italy*, Vol. 1260.

Napoli, C., Pappalardo, G. and Tramontana, E. (2014b). Improving files availability for BitTorrent using a diffusion model, *Proceedings of the IEEE International WETICE Conference, Parma, Italy*, pp. 191–196.

Napoli, C., Pappalardo, G., Tramontana, E., Nowicki, R., Starczewski, J. and Woźniak, M. (2015). Toward work groups classification based on probabilistic neural network approach, *in* L. Rutkowski *et al.* (Eds.), *Proceedings of the International Conference on Artificial Intelligence and Soft Computing (ICAISC)*, Lecture Notes in Computer Science, Vol. 9119, Springer, Berlin, pp. 79–89.

Napoli, C., Pappalardo, G., Tramontana, E. and Zappalà, G. (2016). A cloud-distributed GPU architecture for pattern identification in segmented detectors big-data surveys, *Computer Journal* **59**(3): 338–352, DOI: 10.1093/comjnl/bxu147.

Nowak, B., Nowicki, R., Woźniak, M. and Napoli, C. (2015). Multi-class nearest neighbour classifier for incomplete data handling, *in* L. Rutkowski *et al.* (Eds.), *Proceedings of the International Conference on Artificial Intelligence and Soft Computing (ICAISC)*, Lecture Notes in Computer Science, Vol. 9119, Springer, Berlin, pp. 469–480.

Qiu, D. and Srikant, R. (2004). Modeling and performance analysis of BitTorrent-like peer-to-peer networks, *SIGCOMM Computer Communication Review* **34**(4): 367–378.

Rabiner, L.R. and Gold, B. (1975). *Theory and Application of Digital Signal Processing*, Prentice-Hall, Inc., Englewood Cliffs, NJ.

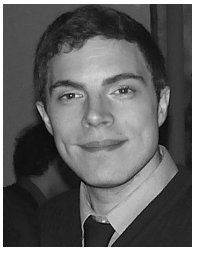
Tramontana, E. (2013). Automatically characterising components with concerns and reducing tangling, *Proceedings of the IEEE Computer Software and Applications Conference (COMPSAC), Workshop QUORS, Kyoto, Japan*, pp. 499–504.

Visan, A., Pop, F. and Cristea, V. (2011). Decentralized trust management in peer-to-peer systems, *Proceedings of the International Symposium on Parallel and Distributed Computing (ISPDC), Cluj-Napoca, Romania*, pp. 232–239.

Williams, R.J. (1989). A learning algorithm for continually running fully recurrent neural networks, *Neural Computation* **1**: 270–280.

Williams, R.J. and Zipser, D. (1989). Experimental analysis of the real-time recurrent learning algorithm, *Connection Science* **1**(1): 87–111.

Woźniak, M., Połap, D., Gabryel, M., Nowicki, R., Napoli, C. and Tramontana, E. (2015). Can we process 2D images using artificial bee colony?, *in* L. Rutkowski *et al.* (Eds.), *Proceedings of the International Conference on Artificial Intelligence and Soft Computing (ICAISC)*, Lecture Notes in Computer Science, Vol. 9119, Springer, Berlin, pp. 660–671.



Christian Napoli received the B.Sc. degree in physics from the Department of Physics and Astronomy, University of Catania, in 2010, and the M.Sc. degree in astrophysics in 2012. Since 2009 he has been a student research fellow at the Department of Electrical, Electronics, and Informatics Engineering, University of Catania, and a collaborator of the Astrophysical Observatory of Catania as well as the National Institute for Nuclear Physics. He is currently pursuing a Ph.D. in informatics with the Department of Mathematics and Informatics, where he is also a research fellow in several projects. His current research interests include neural networks, artificial intelligence, distributed systems and computational models, with emphasis on hybrid neural network modelling techniques for massively parallel architectures.

Giuseppe Pappalardo has been a full professor of computing science at Catania University since 2002. His research is mainly in the area of distributed computing. In particular, he has focused on several topics, including formal description of distributed systems (with realistic applications and contributions to semantical foundations and verification techniques), fault tolerance issues in distributed replicated systems and remote procedure calls, the generalisation of the notion of implementation based on the new concept of interface diversity, the definition of reflective software architectures for the transparent evolution and adaptation of distributed systems.

Emiliano Tramontana has been an assistant professor of computing science at Catania University since 2007. The main areas of his research concern software engineering and distributed systems. More specifically, his research activities have included innovative solutions for consistent update of distributed applications at runtime; more modular and reusable versions of some of the most commonly used design patterns; architectural and algorithmic solutions to negotiate QoS parameters and for the management of resources in order to meet the thresholds provided on QoS parameters; concepts and metrics for analysing the modularity of software systems and to suggest improvements on the quality of the code through refactoring techniques.

Received: 20 January 2015
Revised: 14 July 2015