Bittella: A Novel Content Distribution Overlay Based on Bittorrent and Social Groups

Rubén Cuevas, Carmen Guerrero, Isaías Martinez-Yelmo , Ángel Cuevas, and Carlos Navarro

Departamento Ingeniería Telemática. Universidad Carlos III de Madrid, 28911 Leganés (Spain)*
{rcuevas,guerrero,imyelmo,acrumin,cnavarro}@it.uc3m.es

Abstract. This paper presents *Bittella*: a new social network for content distribution based on Peer-to-Peer technologies. It exploits the common interests of the users in order to create social groups based on an algorithm called *Ranking Algorithm*. On the other hand, *Bittella* is deployed over a semantic-search based and unstructured p2p network, in spite of this it uses Bittorrent-like download techniques in order to improve the download time. For this purpose, a new Bittorrent trackerless scheme is proposed.

1 Introduction

Peer-to-Peer (p2p) systems have become one of the most successful technologies in the Internet during the last years supported by file-sharing applications like Gnutella [6], Bittorrent [1] or Kademlia [8]. Two main categories of p2p systems have been defined so far [7]: unstructured p2p systems (e.g. Gnutella [6]) and structured p2p systems (e.g. Kademlia[8]). The main problem of unstructured p2p systems is the generation of massive traffic in the search procedure which is usually based on flooding. On the other hand, there is not control on the data placement, thus, these systems are resilient in dynamic environments. The structured p2p systems are based on Distributed Hash Table (DHT). Therefore, there is control on data placement and the search procedure generates less traffic than in the p2p unstructured systems. However, this control on data placement produces a very high cost to maintain a consistent distributed structure in the typical dynamic environment of file-sharing applications. Moreover, there is a third type of p2p network model which provides a better download rate: Bittorrent [1]. It is based on a web servers infrastructure in order to perform the searching procedure of the file. The web servers store the *.torrent* file which indicates the IP address of the *Tracker*. This is the central entity responsible of the management of the sharing process within the Bittorrent swarm.

* This work was supported by the European Commission through the NoE CONTENT FP6-CONTENT-038423, the Spanish government through the Project IMPROVISA TSI2005-07384-C03-027 and the Madrid regional government through the Project BIOGRIDNET CAM-S-0505/TIC-0101.

On the other hand, the Social Networking is a novel phenomenon which is growing exponentially in the Internet during the last years. This concept was firstly studied on the Social Sciences [10], and latter it was adopted by the Internet community in different applications (e.g. Skype or MSN). However, if we consider the content distribution applications based on p2p systems, and more concretely the file-sharing applications, the concept of social relationship is not used. Hence, the advantages of the social relationships (defined by the common interests among the peers in the content distribution applications) are not exploited on the current p2p systems.

This paper proposes *Bittella*, a novel *Social Network for Content Distribution*. *Bittella* creates the social groups with common interests transparently to the final user. For this purpose, it is based on a semantic-search based p2p network. This type of search permits a better exploitation of the common interests and thus, the formation of a more robust and reliable Social Network. Due to the semantic search is much easier to be performed on the unstructured p2p networks, we decided to implement *Bittella* over this type of p2p networks. In order to solve the problem of the overloaded traffic generated in the searching process in unstructured p2p, *Bittella* exploits the social relationships and the common interests among peers. Thus, a peer queries firstly the peers sharing its interests. These peers have the highest probability of storing the desired content. By doing so, the flooding is drastically reduced.

On the other hand, since it has been demonstrated that Bittorrent is the most effective p2p technology in terms of download rate and fairness, the decision was to use it for the file-sharing in our scheme. This implies to utilize Bittorrent over an unstructured and semantic-search based p2p network. For this purpose, a novel trackerless Bittorrent model for *Bittella* is defined in this paper.

The *Bittella* architectural framework is composed by three different layers: the lowest layer is the unstructured p2p network (e.g. Gnutella); the medium layer is formed by the Bittorrent-like swarms for file-sharing; the upper layer is the social layer where we can find the social groups with common interests.

Finally, the paper introduces the *Ranking Algorithm*, a novel procedure which leads to the creation of groups of peers with common interests. It ranks the known peers considering those in the first positions of the ranking as partners on the social group.

2 Basic Functionality of *Bittella* Protocol

This section explains the functionality of *Bittella*. Firstly, we present the procedure to create the Bittorrent-like swarm to share a given content. Indeed, this is the description of our trackerless Bittorrent. After that, the section presents the functionality of the *Bittella* searching protocol. Then, the section describes the *Ranking Algorithm* and the creation of the social groups. Finally, we introduce the concept of the *Secure Permanent Peer ID* which leads to obtain a reliable and robust social structure along the time.

2.1 *Bittella* Swarm Creation

This section assumes that there is an unstructured p2p network already deployed. Indeed, Gnutella will be considered during the remainder of the section, but it could be any other unstructured p2p. In this scenario, when a peer has a new content to share, it operates similarly as a file provider in Bittorrent: it divides the content into chunks, computes the hashes of each chunk and the complete file, and creates a *.bittella* file with the number of each chunk and the hash associated to it and the hash of the complete file. At this point, there is a seed (named *Bittella Seed*) for the given content and this is available on the p2p network. Eventually, a peer (e.g. *Peer A*) solicits the content as will be explained in section 2.2. When the query from *Peer A* reaches the Bittella Seed, this one answers including in the response the *.bittella* file. In the instant when the *Peer A* receives the response, it can starts to download the content by soliciting chunks to the seed. Afterwards, another peer, e.g. *Peer B*, could send a query looking for the same content. This query can reach to: only the seed, only the *Peer A* or both the seed and the *Peer A*. If we suppose that *Peer A* is the one which answers, it delivers to *Peer B* the following data: **(i)** the *.bittella* file; **(ii)** the list of chunks of the content owned by itself; **(iii)** a list of seeds and peers known in the swarm[1] (in this example this list is only composed by the seed). Therefore, the swarm will be growing while new peers solicit the content.

In the traditional Bittorrent, the *.torrent* file is obtained from a Web Server and the management of the sharing process is performed by the Tracker. Some trackerless schemes based on structured p2p networks have been proposed so far. The main objective of these proposal is to remove the single point of failure represented by the Tracker. However, the mechanism of swarm creation introduced in this section can be intended as a novel trackerless Bittorrent scheme. For the best of our knowledge, this is the first proposal where the search of the *.bittorrent* file is done by using an unstructured p2p network.

2.2 *Bittella* Searching Protocol

In this section, we suppose a stable *Bittella* network with several swarms enough populated. In this scenario, if a peer, e.g. *Peer C*, wants to obtain a content, e.g. *Content X*: Firstly, the peer checks if it has stored local information related to *Content X* (The procedure to create the local information is explained in Section 2.3). If it has, then, it sends the query directly to the peer(s) which have high probability to answer the query. This(ese) peer(s) answers to *Peer C* including in the response the following information: **(i)** the *.bittella* file; **(ii)** the list of *Content X* chunks owned by itself (themselves); **(iii)** a list with the IP addresses of seeds and other peers within the swarm. Then, *Peer C* selects other peers from the obtained list and asks them about their list of chunks. At this point, *Peer C* starts the downloading procedure by soliciting chunks to other peers. In addition, if it is necessary the peers can perform a gossiping protocol

[1] It can include all the peers and seeds known by *Peer A* or maybe only a random selection of them.

in order to identify more members within the swarm. This mechanism is named *Peer Exchange* [2].

On the other hand, if *Peer C* has not local information about the *Content X*, it queries those peers present in its same social group. These are the peers sharing its interests and the ones with a higher probability of know the location of the content. However, in some cases the peer can ask for content that differs from its interests, then if it only queries within its social group these unexpected queries could be unsuccessful in many occasion. In order to avoid this, the queries will be also sent to some of the underlying neighbors[2] in order to increase the probability of success of the unexpected queries.

It must be noted that the described procedure must be applied for both type of queries, those generated by the peer and those to be routed by the peer.

Finally, as it occurs in Bittorrent, when a peer finishes the download of a given content it becomes a seed for this content.

2.3 *Bittella* Ranking Algorithm

Firstly, it must be highlighted that we consider a social group as a group of peers with common interests within the p2p network. The *Ranking Algorithm* is the defined procedure in order to find out which peers have common interests, and therefore, belong to the same social group. The Ranking Algorithm runs on each individual peer and is a passive procedure. That is, it does not create any kind of message and only uses the messages of the protocol to evaluate which peers have common interests. This is a great advantage because the *Ranking Algorithm* does not produce any kind of overhead. In addition, the *Ranking Algorithm* allows the creation of social groups in a transparent manner to the user and maintaining the anonymity. These are two of the most important features that the users require to the content distribution applications: *transparency* means simplicity from the user point of view; that is, the application optimizes the searches of contents without any kind of configuration or waste of time from the user, apart from selecting the content to be downloaded. On the other hand, *anonymity* is another required feature since nobody wishes that others could identify what kind of contents he/she is downloading. Therefore, with these three features in mind (no overhead, transparency and anonymity) the *Ranking Algorithm* is defined as follows.

Each peer generates a ranking of the other peers in the *Bittella* network. The top one is the peer with most similarity. In order to rank the other peers, *Bittella* uses two different mechanisms. The first one is the number of swarms where the peer has been met (e.g. a peer receives one point per each swarm where it is found out). It is an intuitive mechanism, if *Peer A* and *Peer B* have common interests they will meet each other in many swarms and they will rank each other in a high position. The second mechanism considers the queries to be routed. That is, if a peer has to route a query which matches with some of its last queries (e.g. the last 20 queries), it gives some points to the peer which

[2] Neighbors in the underlay p2p network.

generated this query. Due to Bittella uses semantic queries, the query can fully match or partially match any of my previous queries, then the points assigned to that peer may vary: for instance, from 0 when there is not match to 1 when the query fully match[3], assigning intermediate values between 0 and 1 when there is a partial match. Here, we can observe the importance of the semantic search in the case of dealing with social groups. It leads to an accurate evaluation of the common interests among the peers. This is the main reason to reject the use of DHT-p2p systems in our purpose, these systems do not permit the use of semantic searches reducing the power of the *Ranking Algorithm* in the creation of the social groups.

Equation 1 includes the formula in order to obtain the Rank of a Peer. The $SwarmMatch_i$ is equal to 1 when the peer has been met on the swarm of the content i and 0 if it has not been met. $QueryMatch_k$ indicates the matching between the k^{th} routed query and the previous generate queries, thus it varies from 0 to 1. The factor β adjust the query matching depending on if the query matched is an old or a recent one, it varies from 0 (oldest) to 1 (the most recent). Finally, α is the factor which adjusts the importance of each one of the mechanism (swarm matching or query matching) and varies from 0 (query matching more important) to 1 (swarm matching more important).

$$PeerRank = \alpha * \sum_i SwarmMatch_i + (1 - \alpha) * \sum_k \beta * QueryMatch_k \qquad (1)$$

When a peer has elaborated the ranking it learns the information about the peers placed in the highest positions (e.g., the top 100; this number depends on the host capacity). This procedure, named *Bittella Learning Procedure*, is performed by directly requiring to the top peers the following information: **(i)** the contents that they have already downloaded; **(ii)** the contents that they are currently downloading; **(iii)** the list of the top peers on their rankings. This is performed every time that a top peer is found in a swarm. That is, when a peer find out one of its top peers in a swarm the former requires from the latter, apart from the list of chunks, the information described above. By doing so, the result of the *Bittella Learning Procedure* is that each peer has the updated information about the available contents in its social group. These are the most likely contents to be solicited by the peer.

Another important feature of the *Ranking Algorithm* is that it is self-adaptive. If the interests of a given node change, the algorithm brings this peer to its new social group due to the peers on this new social group will start to receive points and occupy the first positions in the ranking. In order to make the algorithm more adaptive, peers which are not found in any swarm or whose queries have not been received during a period of time decrease their rank.

2.4 Secure Permanent Peer ID

If we analyze the behaviour of p2p users, they are available on the network intermittently. In the study developed by Pouwelse et al. [9] we find that only

[3] The complete matching is equivalent to find the peer on a swarm.

the 17% of the peers stay more than 1 hour on the Bittorrent network after they finished the download. Then, the peers leave and come back continuously to the network. Furthermore, the most of the users have a dynamic IP address. Hence, when an user leaves the network and rejoins it the next time, the connection will be established (with high probability) with a different IP address. If we consider these factors, a peer only belongs to the social group during the time that it is connected to the *Bittella* network. If it leaves and rejoins again it is considered as a new user and therefore the procedure to discover its social group must be done again. It is not a desirable feature. Thus, in order to avoid this, but keeping the anonymity and transparency initially defined, the following procedure was defined. The peer obtains a Permanent ID when firstly connects to the network. This ID is the public key of a *Public/Private Key Pair.* Then, it discovers its social group. At a certain moment, finally, it leaves the *Bittella* network. Then, all the peers which have this one on their top ranking discover that the peer left and freeze its entry, put it out of the ranking and assigns a timeout to this entry. In order to discover top peers which have left each peer sends periodically keep alive messages to the top nodes in its ranking. If the peer does not return to the *Bittella* network before the time out expires, the entry is removed. Otherwise, if the peer rejoins the network before the timeout expiration (in the worst case with a new IP address) it informs the peers in its top ranking about its presence in the network and its new IP address. Besides, it can be done in a secure way since the others can challenge it with a nonce (encoding something with its ID, that is the *Public Key*) and it can answer the challenge, demonstrating that it is the right node. The peers which receive the message storing the new IP address, recover the entry and put the peer again on the ranking. Therefore, by applying this mechanism the peers are able to leave and join the networks without losing the information about its social group and using a secure process. Moreover, in order to improve the anonymity the peers can change their Permanent Peer-ID. For this purpose, the node generates a new *Public/Private Key Pair* and sends the new Permanent Peer-ID (that is the Public Key) signed with the previous private key to its top ranking nodes. The top ranking nodes can decipher the new Permanent ID and store it.

3 *Bittella* Three Layer Architecture

This section describes *Bittella* from an architectural point of view. *Bittella* is a three layer architecture where the lower layer is called *Underlay Layer* and it is basically the underlying p2p network; the medium layer is the *Swarm Layer* and is formed by the Bittorrent-like swarms; finally, the higher layer is the *Social Layer* which is formed by the social groups based on common interests. Figure 1 represents this architecture.

- **Underlay Layer:** This is the fully distributed and unstructured p2p network (e.g. Gnutella). In this layer each node has some neighbors which are called *Underlay Neighbors.* Basically, when *Bittella* uses this layer, it uses the search mechanism defined on it. Usually, it is flooding.

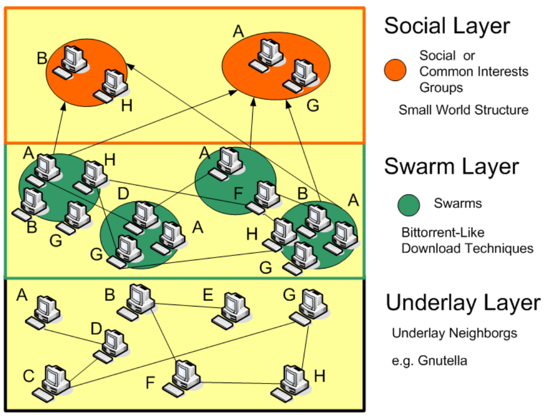

Fig. 1. Three Layer *Bittella* Architecture

– **Swarm Layer:** This layer is formed by different Bittorrent-like swarms. In this paper we focus on the file-sharing, but as future work we will add the live streaming and the VoD distribution. Therefore, there would be different type of swarms. In addition, the swarm layer is basic for the *Ranking Algorithm*.
– **Social Layer:** This layer is formed by different *Social Groups* based on common interests. From the topological point of view, this layer has a *Small-World* structure, that is, a loosely connected graph of highly connected subgraphs [11]. In *Bittella*, the highly connected subgraphs are the *Social Groups* and they are loosely connected by the links among the *Underlay Neighbors*. The *Small-World* structure on p2p networks has been previously analyzed [5] [3] [4].Therefore, all the demonstrated advantages of the Small World structure can be considered *Bittella* advantages as well.

4 An Initial Simulation Analysis

Bittella was simulated in an small scenario with 1000 nodes. Therefore, each node is going to include in its ranking all the other known nodes due to the reduced size of the network. The performed experiments analyzes if the *Bittella Learning Procedure* proposed in this paper is useful. If the results are positive, we can consider the *Ranking Algorithm* as an extension which should be applied in larger networks with a higher number of nodes.

The *Bittella Simulator* was implemented in Java, and it is a discrete time simulator: the time is divided into cycles and each node finishes all the pending events on each cycle. The simulations were deployed with the following parameters: N (the number of nodes forming the Gnutella network) equal to 1000; n (the number of neighbors that each node has in the *Underlay Network*) equal to 5; R (the number of contents offered in the p2p system) equal to 62; C (the number of chunks per content) equal to 100; Q (the number of queries -i.e. content solicitation- generated during the simulation) equal to 3000; S (the number

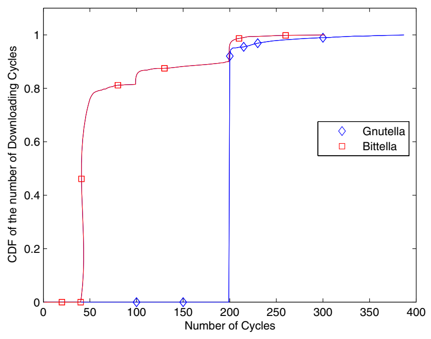

Fig. 2. CDF of the download time: *Bittella* vs. Gnutella

of initial seeds per content) equal to 2; TTL (the radius of the flooding queries) equal to 5; BW (the number of chunks which can be simultaneously downloaded or uploaded) equal to 4. The experiments were deployed in a static situation, where nodes do not join or leave the network and they compare the performance of Gnutella and *Bittella* in terms of *Bandwidth Consumption* and *Download Time*.

The first metric analyzed is the CDF (Cumulative Distribution Function) of the number of simulation cycles spent in the download process. The results are presented in Figure 2. The Figure shows that in a time equal to 50 cycles the 80% of the files have been downloaded using *Bittella* whereas Gnutella downloads never takes less than 200 cycles. Therefore, in the 80% of the download processes *Bittella* reduces the time spent in a factor 4. And in the 100% of the cases it improves the download time of Gnutella. Basically, despite of Gnutella and *Bittella* nodes are configured with the same bandwidth, *Bittella* uses it more efficiently. This behaviour is due to the use of Bittorrent-like download techniques.

The second experiment focuses on evaluating the reduction of bandwidth generation during the searching procedure obtained by *Bittella* in front of Gnutella. In this experiment, we measure the total number of queries generated every 20 simulation cycles. That is, the original queries but also the replication of they forwarded due to the flooding algorithm. In addition, the local hit rate offered by *Bittella* is measured. That is, the ratio of queries which do not need to be flooded (because the peer has local information to solve the query) in front of the total number of the generated queries. Again, it is measured in periods of 20 simulation cycles. Figures 3 and 4 show the results of the experiment. The upper graphic in Figure 3 shows the *Bandwidth Consumption* of Gnutella and *Bittella* in terms of relative bandwidth units. We can see that at the beginning of the simulation, *Bittella* and Gnutella present the same *Bandwidth Consumption*, but according with the simulation advance, Gnutella keeps the same *Bandwidth Consumption*, around $2.5 * 10^5$ bandwidth units, whereas *Bittella* reduces it. At the middle of the simulation (around the cycle 500) *Bittella* offers a *Bandwidth Consumption* around $1.75 * 10^5$ that represents the 30% of reduction compared

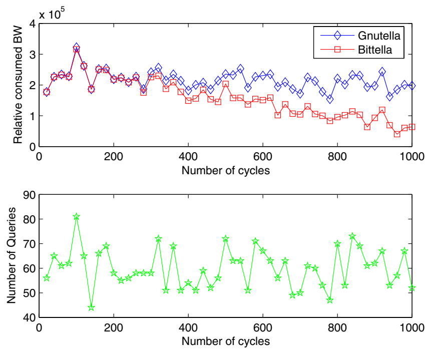

Fig. 3. BW consumption & Generated Queries : Gnutella vs. *Bittella*

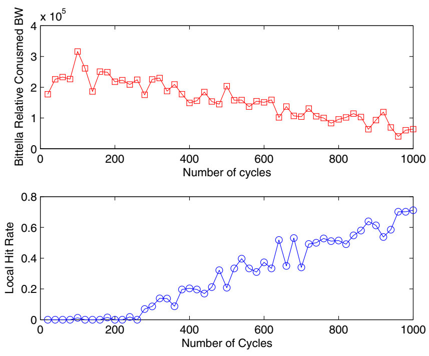

Fig. 4. *Bittella*: BW Consumption & Local Hit Rate

with Gnutella. Even more, at the end of the simulation, *Bittella* shows a *Bandwidth Consumption* below $1 * 10^5$ bandwidth units which means a reduction of the 150% (1.5 times) compared with Gnutella. This reduction in the *Bandwidth Consumption* is produced by the learning procedure described in this paper. The lower graphic on Figure 3 represents the number of queries generated each 20 cycles. If we compare it with the graphic above, it is easy to check that those periods which present a higher number of queries result in more bandwidth consumption, because more queries mean more flooding. However, we can check that the *Bittella Learning Procedure* mitigates this effect. When the nodes have a high level of knowledge (at the end of the simulation) the variation on the number of the queries generated affects in a minor degree because the flooding is drastically reduced.

Finally, Figure 4 shows in the upper graphic the BW consumption of *Bittella* and in the lower graphic the *Local Hit Rate*. This figure demonstrate the behaviour of the *Bittella Learning Procedure*. Along the advance of the simulation, the nodes learn more and more, thus, the local hit rate increases up to reach values near to the 80% at the end of the simulation. The high local hit rate produces the reduction of the number of queries to be flooded and therefore, the drastic reduction of the *Bandwidth Consumption*. Hence, this experiment proves that with the use of *Bittella* the total traffic generated on unstructured p2p networks is reduced.

5 Conclusion and Further Work

This paper presents a novel architecture of social network for content distribution, *Bittella*. It is based on an unstructured and semantic-search based p2p network and exploits the common interests of the users in order to create the *Social Groups*. The paper introduces a new mechanism, the *Ranking Algorithm* which allows the creation of the *Social Groups* in an easy way and without generating any kind of extra traffic. In addition, the paper shows a novel trackerless Bittorrent architecture which permits the use of Bittorrent downloading techniques on unstructured p2p networks. For the best of our knowledge, this is the first trackerless Bittorrent system over a fully distributed unstructured p2p systems proposed so far. Finally, *Bittella* presents the concept of the *Secure Permanent ID* which permits to maintain the social structure in spite of the churn behaviour of the nodes in p2p environments.

Some preliminary results have been presented in the paper showing that *Bittella* increases the download rate compared with other fully distributed unstructured p2p systems as Gnutella. Besides, the experiments have demonstrated that the *Bittella Learning Algorithm* proposed reduces drastically the traffic needed for the search procedure compared to the flooding algorithm used by Gnutella. This is due to the high local hit rate obtained by the application of the learning algorithm.

Further work will focus on the deep analysis of *Bittella* protocol and the *Ranking Algorithm* by means of simulation. In addition, the distribution of VoD and Live Streaming on *Bittella* will be intensively studied.

References

1. Bittorrent, `http://www.bittorrent.org`
2. Peer exchange, `http://en.wikipedia.org/wiki/Peer_exchange`
3. Cohen, E., Fiat, A., Kaplan, H.: Associative search in peer to peer networks: harnessing latent semantics. In: Proceeedings of INFOCOM 2003 (2003)
4. Hui, K., Lui, J., Yau, D.: Small world overlay P2P networks. In: Proceedings of IWQOS 2004 (2004)
5. Iamnitchi, A., Foster, I.: On Exploiting Small-World Usage Patterns in File-Sharing Communities
6. Klingberg, T., Manfredi, R.: Gnutella 0.6. Network Working Group (June 2002)
7. Lua, K., Crowcroft, J., Pias, M., Sharma, R., Lim, S.: A survey and comparison of peer-to-peer overlay network schemes. Communications Surveys & Tutorials, IEEE (2005)
8. Maymounkov, P., Mazieres, D.: Kademlia: A peer-to-peer information system based on the XOR metric. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, Springer, Heidelberg (2002)
9. Pouwelse, J., Garbacki, P., Epema, D., Sips, H.: The Bittorrent P2P File-sharing System: Measurements and Analysis. In: Castro, M., van Renesse, R. (eds.) IPTPS 2005. LNCS, vol. 3640, Springer, Heidelberg (2005)
10. Wasserman, S., Faust, K.: Social network analysis. Cambridge Univ. Press, Cambridge (1995)
11. Watts, D., Strogatz, S.: Collective dynamics of'small-world'networks. Nature (1998)