DANTE: A Self-Adapting Peer-to-Peer System

Luis Rodero Merino[1], Luis López[1], Antonio Fernández[1], and Vicent Cholvi[2]

[1] LADyR, Universidad Rey Juan Carlos,
28933, Móstoles, Spain
{lrodero,llopez,anto}@gsyc.escet.urjc.es
[2] Universitat Jaume I,
12071, Castellón, Spain
vcholvi@lsi.uji.es

Abstract. In this paper we introduce DANTE, an unstructured P2P system in which the topology of the underlying overlay network can be dynamically adapted to the system conditions. Such an adaption is performed by the peers in an autonomous manner. DANTE uses a simple search mechanism based on *random walks* that, combined with the topology adaptation, allows it to work in a very efficient way. We have evaluated how DANTE behaves in practice, showing that it adapts very well to varying system conditions.

1 Introduction

Peer-to-peer (P2P) systems [1] are one of the most important revolutions happening in the Internet today, offering new and richer communication opportunities for Internet users. P2P is a new communication paradigm in which *resources*, such as media files, services, data, etc., are *both* provided *and* consumed by all participants (also called *peers* or *network nodes*). This contrasts with the traditional client-server model, in which the role of each participant is restricted and well defined. In P2P systems, instead, each participant is at the same time a server, because it offers resources, and a client, because it demands them. Clear advantages of P2P systems, compared to classical systems, are their flexibility, scalability, and fault tolerance. These properties are mainly due to the lack of any central entity that coordinates or controls the peers. Nonetheless, the lack of a central coordinator has brought many new technical challenges to be solved.

One of the key issues that any P2P system has to face is how to efficiently locate resources. In most systems, to do so, peers that demand resources issue *queries* or *searches* that cause *search messages* to travel through the *overlay network*, looking for peers where those resources are offered. The *search mechanism* implemented by the P2P system dictates how search messages are routed through the overlay network. Roughly speaking, P2P systems and their search mechanisms can be classified as either *structured* or *unstructured*. Structured P2P networks (see [2] for examples) use specialized placement algorithms to assign responsibility for each resource to specific peers, as well as a "directed" search mechanism to efficiently locate resources. In contrast, unstructured P2P

networks (e.g., Gnutella [3]) have no precise control over the resource placement and generally use search mechanisms based on "flooding" or random walks. Directed search mechanisms are particularly efficient, because they efficiently route queries toward the peers responsible for a given resource. Additionally, they usually require few communication steps, generate little traffic, and do not produce false negatives (i.e., the search fails only if there is no matching file in the system).

Search mechanisms based on flooding or random walks are usually less efficient than directed search mechanisms (queries are broadcast in a whole neighborhood or sent in a random walk) and may yield false negatives. They have, however, very little management overhead, adapt well to the transient activity of P2P clients, take advantage of the spontaneous replication of popular content, and allow users to perform more elaborate queries than with directed search protocols. These properties seem to make unstructured P2P systems very suitable for mass-market distributed resource sharing. Flooding, nonetheless, presents the problem of scalability, as the network bandwidth consumed by search messages grows exponentially with the number of nodes and the scope of those messages. Thus, search mechanisms based on random walks have gained growing attention from the research community, which is looking for new ways to improve their efficiency. A new and promising technique to do so is the use of *dynamic topologies*.

In the next section we present DANTE[3], an unstructured P2P system in which the topology of the underlying overlay network is dynamically adapted to the system conditions. Furthermore, DANTE also uses a simple search mechanism based on *random walks*. In Section 3 we show how those features allow DANTE to work in an autonomous and efficient manner. Finally, Section 4 concludes the paper.

2 DANTE

In this section, we describe DANTE, a P2P system that, as it has been said previously, uses a mechanism to form topologies that self-adapt depending on the network load conditions. Such an adaptation is performed by the peers without the need of global information, nor any central system to control their actions. To achieve this, each node runs a *reconnection mechanism* (described in detail in Section 2.2), that periodically decides to which other peers it must connect to.

2.1 Resource Searches in DANTE

In DANTE, each node holds a set of resources and maintains an index of the resources held by its neighbors. Using this information, a node can explore its neighborhood at no communication cost. Clearly, in general, this also increases the success rate and reduces the network traffic, at a moderate storage cost.

[3] From Dynamic self- Adapting Network TopologiEs.

All peers can issue queries, which are performed by using *random walks*. Then, when a peer issues a query, it first locally checks if the searched resource is held by itself or by one of its neighbors. If this check succeeds, then the search finishes successfully. Otherwise, the node issues a TTL[4]-limited *Look For Resource* (*LFR*) message that is sent to a neighbor chosen uniformly at random. On the reception of that message, the receiving node operates in the same way as the first requesting node. The process ends when the resource is found, thus replying to the issuing peer with a *Resource Found* message (*RF*), or when the TTL expires, replying with a *Resource Not Found* (*RNF*) message.

2.2 DANTE Self-Adaptation Mechanism

The self–adaptation mechanism used in DANTE is inspired on the results of Guimerà et al. [4], regarding the relationship between topological properties of networks and the specific dynamic behavior when faced with local search with congestion, and on the protocol proposed by Cholvi et al. [5] regarding how to achieve good topologies, but solves the problems these works presented.

Briefly, Guimerà et al., by means of using a combination of analytical and simulation techniques, were able to characterize the topologies that, given a search mechanism based on random walks and assuming that each node has information about the resources held by its first-order neighbors, minimize the average time needed to perform a search. Clearly, those topologies should be the topologies of choice in practical overlay networks. They found that when the system is not congested, the topology that provides the best results is a star-like structure formed by a small number of central nodes with the rest of nodes connected to them. Furthermore, they also found that when the system is congested, the topology that provides the best results is a random-like one. But, perhaps more importantly, they reported that there is a sharp transition between these two topologies. These results are very interesting and could be used in almost any system where its entities organize as a network. However, the approach followed by Guimerà et al. assumes a global knowledge of the network, which is usually not available in a real P2P system. A practical topology adaptation mechanism that fits P2P systems should be run locally at the nodes, and should not need global knowledge.

In order to put these results to work, in [5], the authors proposed a mechanism that, depending on the current system load, makes nodes to locally change their connections so that the obtained topologies are random-like for high loads and star-like for low loads. To achieve this, they used a reconnection mechanism that assigns a value Π_i to each node i of the network. Such a value tries to capture the "willingness" of a node to accept new connections. Then, the new end in a reconnection is chosen using probabilities proportional to these values. Unfortunately, this mechanism cannot be directly applied to P2P systems, since although the value Π_i can be locally computed at node i, to choose the new neighbors of a node all values Π_i have to be known at the node.

[4] Time-To-Live, the maximum number of links the message will traverse.

In the rest of the section, we explain how the above mentioned problems are solved in DANTE. As said above, in DANTE each peer knows its own resources as well as the resources held by its neighbors. Based on this, it is easy to understand that nodes will be more interested on being connected to peers with many neighbors. Therefore, DANTE encourages peers to establish connections with high degree nodes. However, this holds only as long as these highly connected nodes can handle all the incoming traffic. If the number of queries is high, a well- connected peer may receive more search messages that it can manage, thus becoming *congested*. To face this, the mechanism used in DANTE considers all congested nodes as the worst possible candidates, regardless of their degree.

Taking into account this reasoning, DANTE uses an algorithm that, when the network traffic is low, drives the network to a star-like overlay topology. Thus, searches could be answered in only one hop, since the central nodes will know all the resources in the system. In turn, when the number of searches increases, well-connected nodes will become congested and their neighbors will start to disconnect from them. Hence, this will drive the network to a random-like topology that although makes search messages to traverse longer paths to find some resource, will perform better than using a highly congested central node.

More specifically, in DANTE each node can establish connections to other nodes. We say that a connection is *native* for the establishing node and *foreign* for the accepting node. Nodes can change their native connections, but not their foreign ones. Furthermore, each node periodically runs a reconnection mechanism with which native connections are changed. This mechanism firstly obtains a list of potential candidates to which it can connect (this is described in Section 2.3). Then, it assigns a probability to each candidate, and chooses candidates at random using their respective probabilities. Finally, the peer reconnects its native connections to the chosen candidates.

The probability assigned to a candidate i is based on its "attractiveness", denoted as Π_i and defined as

$$\Pi_i \; = \; k_i^{\gamma_i}, \tag{1}$$

where k_i is the degree (number of neighbors) of peer i, and γ_i is computed as

$$\gamma_i \; = \begin{cases} 2 \text{ if node } i \text{ is not congested} \\ 0 \text{ otherwise.} \end{cases} \tag{2}$$

So, Π_i is set to 1 if peer i is congested, and to k_i^2 otherwise (note that the congestion of a node is a value that can be measured directly from the node's local state). Based on the values Π_j for each candidate j in the set C of candidates, we assign to peer i the probability p_i of being chosen as

$$p_i = \frac{\Pi_i}{\sum_{j \in C} \Pi_j}. \tag{3}$$

As it can be readily seen, nodes with higher attractiveness will be chosen with higher probability. Therefore, in DANTE there is a tendency to connect to nodes with high degree, unless those peers become congested.

The rationale behind the assignment of probabilities is as follows. First, note that it is known [6] that by assigning the same probability to each node, one obtains a random-like topology. This is achieved when all nodes are congested and so Π_i is set to 1 for all nodes. In turn, if no node is congested and the value of Π_i is set to k_i^2 for each node i, one obtains a star-like topology [6]. Consequently, the network will evolve towards a random-like topology when many nodes get congested and towards a star-like topology otherwise. Remember that this will provide us with the topologies of choice, both at low and high network loads.

2.3 Peer Sampling

In order to provide peers with sets of candidates (and their congestion level) to apply the heuristic presented in the previous section, DANTE uses a special message *Look For Nodes* (*LFN*) that is used for collecting information about the state of the network. This message traverses the network following a TTL-limited random walk, storing information about the nodes it visits. When the message TTL expires, a *Nodes Found* (*NF*) message is sent to the message's source node, carrying the information about the peers the *LFN* message visited. Then, the decision is taken considering only this information.

Clearly, if reconnections are not very frequent, this technique of sampling the peers has very small incidence on the network load. Furthermore, it has been shown [7, 8] that the sample obtained with this mechanism is a good sample of the overall network. Indeed, when the network has highly connected nodes or *hubs* (possibly due to low or medium loads), since the collecting message follows a random walk, these hubs will be reached with higher probability than poorly connected nodes. This is good since peers are mainly interested on hubs for reconnections. On the other hand, when the network is random (possibly due to high load), all nodes will have roughly the same degree, and then the chosen nodes will be representative of the whole network.

Other mechanisms than using random-walk messages could be considered for the purpose of collecting information. An interesting option is to use a gossip-based technique to spread data about nodes state. Jelasity et al. [9] use the term *peer sampling service* to generically name a service based on gossiping that provides nodes with a random sample of peers in the network. The peers provided by this service would then be the set of candidates used in DANTE. An example of peer sampling service is the one provided by the *T–Man* protocol of Jelasity et al. [10]. In *T–Man* each node keeps a *view* of the network, consisting on a list of peers of a bounded size c. To spread information about peers, nodes periodically exchange and combine their views. Peers listed in a view are ranked, and when the list grows to a size greater than c, the peers with the lowest ranks are discarded. Therefore, nodes could use the peers in their views as the set of candidates for reconnections.

2.4 DANTE Robustness

Another interesting feature of DANTE is its robustness against node failures. This comes from the fact that when a peer enters or leaves the network, only its neighborhood is affected. Therefore, if the current network topology is random-like, a very small number of peers will notice the change. The same will happen if the current network topology is star-like and the node that leaves is not a central one. Even in the worst case, when a central node disappears, the system still will be able to keep working, since there are several central nodes (as many as the number of native connections of each peer). Furthermore, the DANTE adaptation mechanism also guarantees that some other node will quickly become central and replace the peer that disappeared.

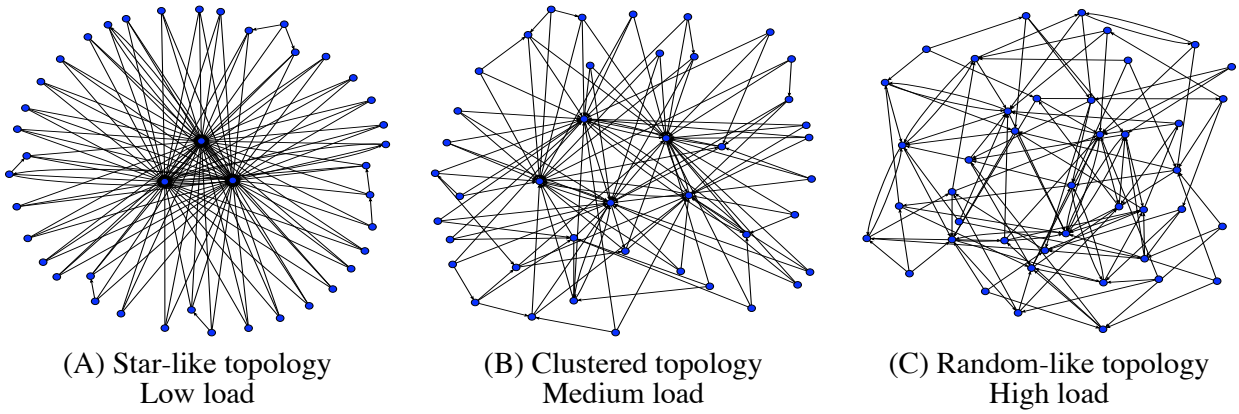

Fig. 1. Overlay topologies obtained in DANTE's prototype with (A) low, (B) medium, and (C) high load.

3 DANTE at Work

A prototype of DANTE, as described above, has been implemented and used to evaluate the properties of the system. The prototype has been developed in Java and works over UDP[5]. Experiments performed with this implementation on a real network have confirmed that, as expected, the overlay network topology evolves as the load on the system changes, ranging from star-like topologies under low load to random-like topologies under high load. Moreover, our experiments also show that these topologies present the best performance for these load levels.

[5] The source code of the prototype is available at http://gsyc.escet.urjc.es/ladyr/dante/.

3.1 Experimental Setup

We start by describing the configuration we have used to run the experiments of DANTE's prototype. Our experiments have been executed with 42 peers, each with three native connections. (Initially, the network had a random topology.) Peers hosted disjoint sets of resources, all containing 5000 resources. (Note that there was no replication.) Every peer periodically issued a new query, in which the resource to search was chosen uniformly at random from the set of all resources in the system. The load in the system was controlled by the *query generation rate*, which was the number of queries per minute issued by every peer in the system. This rate was fixed for each experiment. We have run experiments with values of the query generation rate from 2 to 12 in steps of 2.

In the experiments conducted each query was issued with a TTL of 30. This value was empirically chosen in order to maintain a high success rate in searches (few false negatives). With this TTL value the rate of successful searches has been above 99% for all experiments, except when the topology was fully random, that had a 96% success rate.

In our experiments all peers had the same *real* processing power, since it was the same software running on similar hardware[6]. In each experiment, there was a global parameter named *capacity threshold* (or just *threshold*). This parameter intended to summarize the level of load every peer can take before being congested. In our prototype the threshold represents the maximum number of queries per minute a peer can handle: if a peer receives a number of queries per minute greater than this threshold, the node is considered to be *congested*. We have run experiments with 5 different threshold values, namely, 0, 10, 50, 100, and 1,000,000.

Each experiment has been run for 120 minutes, out of which we have analyzed only the queries started between minutes 16 and 75, both included (to avoid initial transient states and unfinished searches). During the experiments, DANTE's adaptation mechanism has been triggered periodically at each peer every 30 seconds. Each time this happened, the peer changed its three native connections simultaneously.

3.2 Topology Adaptation

The first fact that can be observed from the experiments conducted is that the network topology actually adapts itself to the load in the system. This fact can be readily observed in Figure 1. This figure shows the network topologies obtained with the same threshold (of 10) under three different load levels. In Figure 1.(A) the system is lightly loaded. As expected, the network has evolved to a star-like

[6] Although typical P2P systems have peers with different capacities, the resulting topologies in these heterogeneous systems under very low and very high loads would be similar to the ones obtained here. Furthermore, heterogeneity could improve the network performance, as the peers with higher capacity would become hubs, and hence the number of hops needed to find resources would be decreased.

topology. In Figure 1.(C) we see the overlay network obtained under high load, which forms a random-like topology.

It is interesting to observe the network topology obtained under medium load, shown in Figure 1.(B). As it can be observed, the topology is somewhere in between a random-like and a star-like. In these networks obtained under medium load there are hubs that know many other peers. This, in general, will allow queries to finish in fewer hops than with a fully random topology.

Regarding topology adaptation, we have two especial sets of experiments in which no topology change is observed. The first set is the one done with a threshold value of 0. With this threshold all peers permanently consider themselves to be congested, and hence the resulting topology is always random-like, independently of the load. The second is the set of experiments done with a threshold value of 1,000,000. Since no peer ever receives that many queries per minute, then no peer ever considers itself congested. This makes the network to form a star-like topology regardless of the load on the system. These two sets of experiments have been run to have a reference on the performance of systems with pure random-like and star-like topologies.

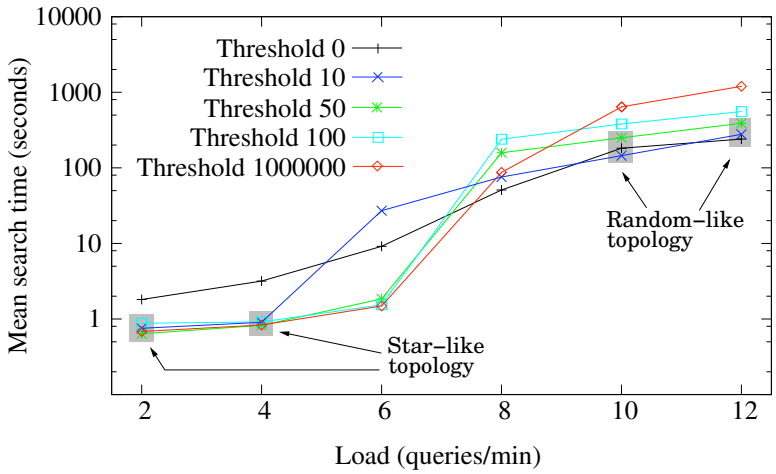

Fig. 2. Results of DANTE's prototype execution.

3.3 Performance

We study now the search performance observed in our experiments under different topologies and loads. To measure the search performance we use the *mean search time*, which is computed as the average time taken to complete a query. A query is completed when either the issuing peer finds out the peer holding the resource or it receives a message indicating that the query failed. The values of the mean search time for all the experiments conducted are presented in

Figure 2. The values that correspond to executions with the same threshold are connected.

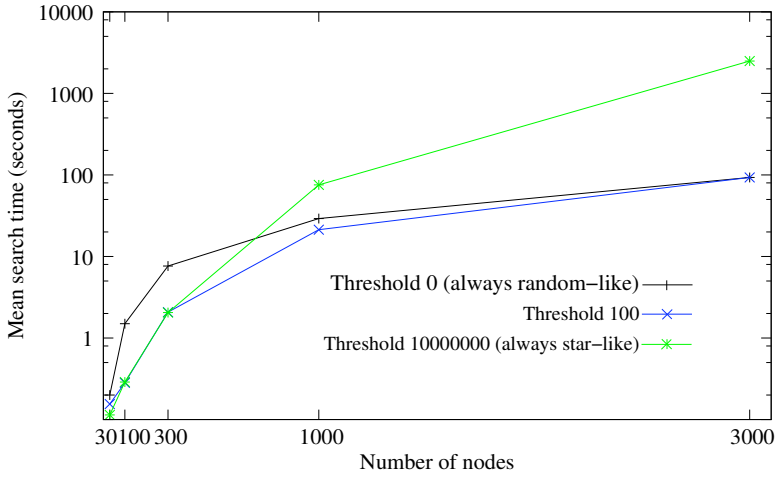

Fig. 3. Results of DANTE scalability simulations.

A first look at Figure 2 allows to confirm the analytical results of Guimerà [4]. On the one hand, among those considered, the star-like topology (threshold 1,000,000) has the best performance under low load. The random-like topology (threshold 0), on the other hand, has the best performance under high load. Interestingly, the random-like topology has the worst performance under low load while the star-like topology has the worst performance under high load, justifying the interest on topology adaptation.

A second conclusion that can be extracted from Figure 2 is that, when using a proper threshold, DANTE makes the network evolve to a topology with good performance given the system load. That is the objective of DANTE's adaptation mechanism: the network is able to self-adapt to the load conditions, trying to keep the topology close to optimal. Interestingly, the overall performance depends on tuning the threshold value properly. As it can be observed, while the three "reasonable" thresholds considered (10, 50, and 100) guarantee close to optimal performance under extreme load conditions, their performance at medium loads is not the same. For a query generation rate of 6 the experiments with threshold 10 show bad performance, because the threshold is too small and prevents the network to evolve to a star-like topology (which seems to be the optimal for this load).

3.4 Scalability

We now study how DANTE's performance changes as the number of peers increases. To do so, we fix a query generation rate and run experiments with

systems of different sizes. Since the query generation rate is fixed, peers issue queries at the same rate, independently of the size. However, since most queries cannot be completed locally, the average load per peer (number of queries processed by the peer) will grow with the size (even in a star-like topology). This means that we cannot expect to observe that the mean search time remains constant as the network size grows (which is a classical definition of scalability).

In order to evaluate systems with thousands of peers we have developed a simulator of DANTE[7], which captures the essence of both DANTE and its prototype. Then, we have run simulations under similar conditions as the experiments done with the prototype. We performed simulations with five different network sizes, namely, 30, 100, 300, 1000, and 3000, and three different threshold values, namely, 0, 100, and 1,000,000. In all cases the query generation rate was fixed to one query every 100 seconds. In order to guarantee a high success rate[8] we set the TTL for query messages to $n \log n$ for a system with n peers (this estimation is based on results in [11]), and fixed the TTL of *LFN* messages to 25 (which empirically provided a good peer sample, even for 3,000 peers).

The results obtained from the simulations are presented in Figure 3. There it can be seen that the star-like topology (threshold value of 1,000,000) shows very good performance for networks with few peers. However, as the number of peers increases, the central nodes get congested and the performance degrades quickly. On the other hand, the random-like topology (threshold value of 0) shows a comparatively bad performance for low number of peers, but its relative performance improves as the number of peers increases. As expected, the experiments with threshold 100 present the very desirable feature that for small networks show a performance close to that of the star-like topology, while a performance close to that of a random-like topology for large networks. Interestingly, for medium size networks (1000 peers) this threshold shows better performance than both the star-like and the random-like topologies.

4 Conclusions

P2P systems are a promising new paradigm, specially suited to situations where there is not a hierarchy among system participants. However, the lack of central entities in the system demands innovative solutions to new problems. For example, users do not have a central repository to ask for the location of resources. To face this problem, new search techniques must be devised.

Recent research has shown the key importance of the network topology on search efficiency. With DANTE we propose a self-adapting mechanism that makes the network change its topology aiming always to an optimal configuration that depends on the system load. This mechanism envisions the P2P system as a community where, from the individual work of participants, a global behavior emerges making the system able to adapt to changing conditions.

[7] We could not run those experiments in a real network since this requires having an infrastructure formed with thousands on peers.

[8] All completed searches were successful.

The first results obtained with this approach seem promising. However, much work remains to be done in order to improve the efficiency of these techniques. For example new heuristics can be developed that make the network topology to evolve more smoothly depending on the peer congestion, avoiding sharp changes.

References

1. Androutsellis-Theotokis, S., Spinellis, D.: A survey of peer-to-peer content distribution technologies. ACM Comput. Surv. **36** (2004) 335–371
2. Balakrishnan, H., Kaashoek, M.F., Karger, D.R., Morris, R., Stoica, I.: Looking up data in P2P systems. Communications of the ACM **46** (2003) 43–48
3. Gnutella.com: (The gnutella website) http://www.gnutella.com.
4. Guimerà, R., Díaz-Guilera, A., Vega-Redondo, F., Cabrales, A., Arenas, A.: Optimal network topologies for local search with congestion. Physical Review Letters **89** (2002)
5. Cholvi, V., Laderas, V., López, L., Fernández, A.: Self-adapting network topologies in congested scenarios. Physical Review E **71** (2005) 035103
6. Krapivsky, P.L., Redner, S., Leyvraz, F.: Connectivity of growing random networks. Physical Review Letters **85** (2000) 4629–4632
7. Gkantsidis, C., Mihail, M., Saberi, A.: Random walks in peer-to-peer networks. In: INFOCOM. (2004)
8. Newman, M.E.J.: A measure of betweenness centrality based on random walks. Social Networks **27** (2005) 39–54
9. Jelasity, M., Guerraoui, R., Kermarrec, A.M., van Steen, M.: The peer sampling service: Experimental evaluation of unstructured gossip-based implementations. In: Lecture Notes in Computer Science (Proceedings of Middleware 2004). Volume 3231., Springer-Verlag (2004) 79–98
10. Jelasity, M., Babaoglu, O.: T-Man: Gossip-based overlay topology management. In: Proocedings of the 3rd international workshop on engineering self-organising applications (ESOA05). (2005)
11. Cooper, C., Frieze, A.: The cover time of sparse random graphs. In: Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms, Society for Industrial and Applied Mathematics (2003) 140–147