Chapter 9

Dynamic Networks

Many large-scale distributed systems and networks are dynamic. In some networks, e.g., peer-to-peer, nodes participate only for a short period of time, and the topology can change at a high rate. In wireless ad-hoc networks, nodes are mobile and move around. In this chapter, we will study how to solve some basic tasks if the network is dynamic. Under what conditions is it possible to compute an accurate estimate of the size or some other property of the system? How efficiently can information be disseminated reliably in the network? To what extent does stability in the communication graph help to solve these problems?

There are various reasons why networks can change over time and as a consequence, there also is a wide range of possible models for dynamic networks. Nodes might join or leave a distributed system. Some components or communication links may fail in different ways. Especially if the network devices are mobile, the connectivity between them can change. Dynamic changes can occur constantly or they might be infrequent enough so that the system can adapt to each change individually.

9.1 Synchronous Edge-Dynamic Networks

We will look at a *synchronous dynamic network model* in which the graph can change from round to round in a worst-case manner. To simplify things (and to make the problems we study well-defined), we assume that the set of nodes in the network is fixed and does not change. However, we will make almost no assumptions how the set of edges changes over time. Formally, we model a synchronous dynamic network by a dynamic graph $G(r) = (V, E(r))$, where V is a static set of nodes, and $E(r)$ is the set of (undirected) edges in round $r \in \{1, 2, 3, \ldots\}$. We assume that the graph $G(r)$ is *connected* in each round r. Otherwise, we allow the graph to change arbitrarily from round to round.

For simplicity, we restrict our attention to deterministic algorithms. Nodes communicate with each other using *anonymous broadcast*: At the beginning of a round r, each node u decides what message to broadcast based on its internal state; at the same time, an adversary chooses a set $E(r)$ of edges for the round. As in standard synchronous message passing, all nodes v for which $\{u, v\} \in E(r)$ receive the message broadcast by node u in round r and each node can perform arbitrary local computations upon receiving the messages from its neighbors.

We assume that all nodes in the network have a unique $O(\log n)$ bit identifier (ID).

In general, we assume that nodes might not all start a computation at the same time and that nodes might be woken up (activated) spontaneously. However, we assume that each node wakes up at the latest when it receives the first message. We refer to this general case as *asynchronous start* and to the special case where all nodes start a computation at time 0 as *synchronous start*.

9.2 Problem Definitions

We study the following two problems.

Counting. An algorithm is said to solve the counting problem if whenever it is executed in a dynamic graph comprising n nodes, all nodes eventually terminate and output n.

***k*-token dissemination.** In an instance of the *k-token dissemination* problem, there are k tokens (pieces of information) and each token is initially known by at least one node $v \in V$ (in general, we allows a node to hold more than one token initially). An algorithm solves k-token dissemination if all nodes eventually terminate and output the complete set of all k tokens. We usually assume that each token in the nodes' input is represented using $O(\log n)$ bits. Nodes may or may not know k, depending on the context. Of particular interest is *all-to-all token dissemination*, a special case where $k = n$ and each node initially knows exactly one token.

9.3 Basic Information Dissemination

To start, let us study how a single piece of information is propagated through a dynamic network. As described, we assume that we have a dynamic network graph G with n nodes such that $G(r)$ is connected for all rounds $r \geq 1$. Further assume that there is a single piece of information (token), which is initially known by a single node.

Theorem 9.1. *Assume that there is a single token in the network. Further assume that at time* 0 *at least one node (which starts at time* 0*) knows the token and that once any node v knows the token, it broadcasts it in every round. In a dynamic graph G with n nodes, after $r \leq n-1$ rounds, at least $r+1$ nodes know the token. Hence, in particular after $n-1$ rounds, all nodes know the token.*

Proof. We can proof the theorem by induction on r. Let $T(r)$ be the set of nodes that know the token after r rounds. We need to show that for all $r \geq 0$, $|T(r)| \geq \min\{r+1, n\}$. Because we assume that at time 0 at least one node knows the token, clearly, $|T(0)| \geq 1$. For the induction step, assume that after r rounds, $|T(r)| \geq \min\{r+1, n\}$. If $T(r) = V$, we have $|T(r+1)| \geq |T(r)| = n$ and we are done. Otherwise, we have $V \setminus T(r) \neq \emptyset$. Therefore, by the assumption that $G(r+1)$ is connected, there must be two nodes $u \in T(r)$ and $v \in V \setminus T(r)$ such that $\{u, v\} \in E(r+1)$. Hence, in round $r+1$, node v gets the token and therefore $|T(r+1)| \geq |T(r)| + 1 \geq \min\{r+2, n\}$. □

Remarks:

- Note that Theorem 9.1 only shows that after $n-1$ rounds all nodes know the token. If the nodes do not know n or an upper bound on n, they do not know if all nodes know the token.
- We can apply the above techniques also if there is more than one token in the network, provided that tokens form a totally-ordered set and nodes forward the smallest (or largest) token they know. It is then guaranteed (in the asynchronous start case) that some node with the smallest (resp. largest) token starts participating after at most $n-1$ rounds and thus, the smallest (resp. largest) token in the network will be known by all nodes after at most $2n-2$ rounds. In the case of synchronous start, all nodes know the smallest (or largest) token after $n-1$ rounds. Note, however, that also in this case nodes do not *know* when they know the smallest or largest token.

The next theorem shows that if the nodes know n, the idea of last remark can be used to solve k-token dissemination for arbitrary k.

Theorem 9.2. *If all nodes know n, they can solve k-token dissemination in $O(kn)$ rounds.*

Proof. There is always a total order on the set of all possible tokens (e.g., by using their bit representation). We can thus use the idea from the last remark above. After $2n-2$ rounds, all nodes know the smallest token and because all nodes know n, all nodes also know when these $2n-2$ rounds are over. In the following, in blocks of $n-1$ rounds, all nodes can learn the smallest of the remaining tokens until all nodes know all the tokens after at most $(k+1)(n-1)$ rounds. □

Remarks:

- The above algorithms seems to use a brute-force approach to solve the token dissemination problem and the resulting running time might seem very large. If we have stronger connectivity requirements about the dynamic graph, the problem can be solved in a faster way. A dynamic graph is called T-interval connected if for any T consecutive round r, $r+1$, ..., $r+T-1$, there is a connected subgraph which is present throughout these T rounds. It can then be shown that k-token dissemination can be solved in $O(n+nk/T)$ rounds.

9.4 Counting the Number of Nodes

The next theorem shows that for the general asynchronous start case, assuming that the graph is connected in every round does not suffice to obtain anything better than what is stated by the above theorems. If nodes do not know n or an upper bound on n initially, they cannot find n or an upper bound on n.

Theorem 9.3. *Counting is impossible in a dynamic graph $G(r) = (V, E(r))$ as defined above with asynchronous start.*

Proof. Suppose by way of contradiction that $\mathcal{A}$ is a protocol for solving the counting problem and assume that $\mathcal{A}$ requires at most $t(n)$ rounds in a dynamic graph of size n. Let $n' = \max\{t(n)+1, n+1\}$. We will show that the protocol cannot distinguish a static line of length n from a dynamically changing line of length n'.

Given a sequence $A = a_1 \circ \ldots \circ a_m$, let $\text{shift}(A, r)$ denote the cyclic left-shift of A in which the first r symbols ($r \geq 0$) are removed from the beginning of the sequence and appended to the end. Consider an execution in a dynamic line of length n', where the line in round r is composed of two adjacent sections $A \circ B_r$, where $A = 0 \circ \ldots \circ (n-1)$ remains static throughout the execution, and $B(r) = \text{shift}(n \circ \ldots \circ (n'-1), r)$ is left-shifted by one in every round. The computation is initiated by node 0 and all other nodes are initially asleep. We claim that the execution of the protocol in the dynamic graph $G = A \circ B(r)$ is indistinguishable in the eyes of nodes $0, \ldots, n-1$ from an execution of the protocol in the static line of length n (that is, the network comprising section A alone). This is proven by induction on the round number, using the fact that throughout rounds $0, \ldots, t(n)-1$ none of the nodes in section A ever receives a message from a node in section B: although one node in section B is awakened in every round, this node is immediately removed and attached at the end of section B, where it cannot communicate with the nodes in section A. Thus, the protocol cannot distinguish the dynamic graph A from the dynamic graph $A \circ B(r)$, and it produces the wrong output in one of the two graphs. □

Remark:

- Theorem 9.3 in particular implies that if no upper bound on the number of nodes is known also k-token dissemination cannot be solved in the asynchronous start case. The nodes could otherwise use their n IDs as n tokens and as soon as all nodes know all n tokens, the nodes also know n. Recall that in the problem definition, we require nodes to terminate.

There are two ways to avoid the impossibility of Theorem 9.3. One can strengthen the connectivity condition and require that for any two consecutive rounds r and $r+1$, the graph $(V, E(r) \cap E(r+1))$ is connected. That is, for any two consecutive round, there is some connected subgraph which is present in both rounds. Intuitively, this condition always allows at least some of the nodes which are woken up in a given round r to report back to some node in round $r+1$. As this assumption makes things much more technical, we instead consider the synchronous start case where all nodes start the protocol at time 0 (with round 1). We will see that this is sufficient to compute the number of nodes and thus by Theorem 9.2 to also solve the k-token dissemination problem.

We study the following simple protocol, where all nodes try to collect the IDs of all the nodes. All nodes maintain a set A containing all the IDs they have collected so far. In every round, each node broadcasts its current set A and it adds all IDs it receives to A. Nodes terminate when they first reach a round r in which $|A| \leq r$.

```
A ← {self};
for r = 1, 2, ... do
    broadcast A;
    receive B_1, ..., B_s from neighbors;
    A ← A ∪ B_1 ∪ ... ∪ B_s;
    if |A| ≤ r then terminate and output |A|;
    ;
end
```

Algorithm 1: Counting in linear time.

Before analyzing Algorithm 1, we fix some notation that will help to argue about the algorithm. Each node $u \in V$ keeps track of a set A of the IDs known to u. We use $A_u(r)$ to denote the value of A at node u at the end of round r (and thus also immediately before round $r+1$ starts).

Lemma 9.4. *Assume that we are given a dynamic graph $G = (V, E)$ which is connected in every round r. If all nodes in V execute Algorithm 1 and if all nodes together start at time 0, for all $u \in V$ and all $r \geq 0$, we have $|A_u(r)| \geq \min\{r+1, n\}$.*

Proof. For a node $v \in V$ and any times $t_1 \geq 0$ and $t_2 \geq t_1$, we define $I_v(t_1, t_2)$ to be the set of nodes u which influence v if only considering the time interval $[t_1, t_2]$ (i.e., including the rounds $t_1+1, \ldots, t_2$). More formally, $I_v(t_1, t_2)$ contains all nodes $u \in V$ such that a message started at time t_1 (i.e., right before round $t_1 + 1$) at node u can reach node v by time t_2. For all $t \geq 1$, we clearly have $I_v(t, t) = \{v\}$. Further, for $0 \leq r < t$, $I_v\big(t-(r+1), t\big)$ can be defined as $I_v\big(t-(r+1), t\big) := I_v(t-r, t) \cup \{u \in V : \exists w \in I_v(t-r, t) \text{ s.t. } \{u, w\} \in E(t-r)\}$, i.e., all the nodes in $I_v(t-r, t)$ and all the nodes for which u and $I_v(t-r, t)$ are connected by an edge in round $t-r$.

Note that the set of nodes which a node v knows at time t is exactly the nodes in $I_v(0, t)$ and to prove the lemma, it therefore suffices to show that for all v and all $t \geq 0$ and all $0 \leq r \leq t$, we have $|I_v(t-r, t)| \geq \min\{r+1, n\}$. As $I_v(t, t) = \{v\}$, the statement is clearly true for $r = 0$ (for any t). For $r > 0$, the recursive definition of I_v yields that $I_v(t-r, t) = I_v\big(t-(r-1), t\big) \cup \big\{u \in V : \exists w \in I_v\big(t-(r-1), t\big) \text{ s.t. } \{u, w\} \in E(t-(r-1))\big\}$. if $I_v\big(t-(r-1), t\big) = V$, we also have $I_v(t-r, t) = V$ and therefore $|I_v(t-r, t)| = n$. Otherwise, the connectivity requirement guarantees that in round $t-(r-1)$, there is at least one edge connecting the nodes in $I_v\big(t-(r-1), t\big)$ with the remaining nodes and therefore $I_v(t-r, t)$ contains at least one node which is not contained in $I_v\big(t-(r-1), t\big)$. □

Theorem 9.5. *In an dynamic graph G which is connected in every round, using Algorithm 1 and assuming synchronous start, all nodes can compute the number of nodes (and terminate) in n rounds.*

Proof. Follows directly from Lemma 9.4. For all nodes u, $|A_u(r)| \geq r+1 > r$ for all $r < n$ and $|A_u(n)| = |A_u(n-1)| = n$. □

Remarks:

- If for any two consecutive rounds r and $r+1$, the graph $G(r) \cap G(r+1) := (V, E(r) \cap E(r+1))$ is connected (i.e., the dynamic graph is 2-interval connected), a similar algorithm also solves the counting problem for the more general asynchronous start assumption.

9.5 Token Dissemination Lower Bound

We have seen that the k-token dissemination problem can be solved in n rounds in the synchronous start case even if n is not known to the algorithm. However, this algorithm requires nodes to send very large messages. As nodes always broadcast all IDs and tokens they know, a message can consist of n node IDs and k tokens. If n is known, we have seen a simple algorithm in which every message consists of at most one token. Hence, if tokens consist of $O(\log n)$ bits, all messages have size $O(\log n)$. However, the running time of this anelgorithm is $O(nk)$. In the following, we will see that for deterministic so-called *token-forwarding algorithms*, the trivial $O(nk)$ upper bound can at best be beaten by a logarithmic factor. A token-forwarding algorithm is an algorithm in which in each round, each node can only broadcast a subset of the tokens it knows. It is for example not allowed to broadcast a combination (e.g., the bitwise XOR) of several tokens. For our lower bound, we assume that in every round, each node is only allowed to broadcast one of the tokens it knows.

Analogously to Algorithm 1, let $A_v(r)$ be the set of tokens known by node v after r rounds. Consider round $r+1$. Before round $r+1$, node v knows the tokens in $A_v(r)$ and therefore in round $r+1$, v can only broadcast some token $x_v(r+1)$ in $A_v(r)$. Because we only consider deterministic algorithms, for each node v, $A_v(r)$ and the token $x_v(r+1)$ sent by v in round $r+1$ only depend on the initial states of all nodes and the graph sequence of the first r rounds. The adversary which constructs the sequence of graphs therefore knows $A_v(r)$ and $x_v(r+1)$ of all nodes v when constructing the graph for round $r+1$. We will make use of this and show how to construct a graph on which only a few new tokens can be learned.

As long as all (or most) nodes only know a small number of tokens, it is not possible to show that in any given round, the total progress is small and only a few tokens can be learnt in the whole network network. Consider for example the case where there are n tokens and in the beginning, every node has exactly one token. Then, already in the first round, every node learns at least one addition token. In order to only consider states where all nodes already know many tokens, we use an additional trick. For each node $v \in V$, we define an additional token set R_v. For any $r \geq 0$, we say that node v learns a token τ in round $r+1$ iff $r \notin A_v(r) \cup R_v$, i.e., we essentially pretend that node v knows all tokens in R_v from the beginning.

Consider a round $r+1$ and two nodes u and v. Recall that $x_u(r+1)$ and $x_v(r+1)$ are the tokens broadcast by u and v in round $r+1$, respectively. If $x_u \in A_v(r) \cup R_v$ and if $x_v \in A_u(r) \cup R_u$, even if u and v are connected by an edge in $G(r+1)$, they cannot learn a token from each other in round $r+1$. In this case, we say that the edge $\{u, v\}$ is a *free edge* in round $r+1$. As an adversary, we construct the graph $G(r+1)$ of round $r+1$ as follows. We first add all free edges. Note that even after adding all free edges, no node learns

any new token (according to our definition of learning a token). After adding all free edges, we add any minimal set of additional edges to make $G(r+1)$ connected. The following lemma shows that there are reasonably small sets R_v for all nodes $v \in V$ such that we need to add at most $O(\log(nk))$ non-free edges.

Lemma 9.6. *There exist sets R_v for $v \in V$ such that for all $v \in V$, $|R_v| \leq 2k/3$ and for all rounds r and all $\{x_v(r) : v \in V\}$, after adding all free edges, the number of remaining components is at most $O(\log(nk))$.*

Proof. We prove the lemma by using the probabilistic method. We choose the sets R_v at random and we then show that with probability larger than 0 (we can also prove high probability), the sets R_v are good (in the sense of the lemma statement) for all possible combinations of transmitted tokens $x_v(r)$. This implies that there exists a combination of sets R_v which works for all combinations of $x_v(r)$.

The sets R_v are chosen as follows. For different nodes u, v, R_u and R_v are chosen indepedently. Further, for each node $v \in V$, R_v contains each of the k tokens independently with probability $1/2$. Now consider a round r and assume that we add all the free edges. If the resulting graph consists of ℓ connected components, we need to add $\ell - 1$ additional edges to make the graph connected. Let us therefore consider the connected components of the resulting graph. Assume that the node sets of the resulting connected components are $V_1, \ldots, V_\ell$. We consider an arbitrary set $\{v_1, \ldots, v_\ell\}$ of nodes such that for each $i \in \{1, \ldots, \ell\}$, $v_i \in V_i$. Note that by definition of the sets $V_1, \ldots, V_\ell$, for any $i \neq j$, the edge $\{v_i, v_j\}$ is not free. In other words, in the graph induced by all non-free edges, the set $\{v_1, \ldots, v_\ell\}$ induces a clique of size ℓ. We next show that the occurrence of such a clique consisting of only non-free edges has very low probability.

First, note that for every node v, the token $x_v(r)$ broadcast by v in round r has to be from the set $A_v(r-1)$. Hence, if for any two nodes u and v, $x_u(r) = x_v(r)$, the edge $\{u, v\}$ is definitely free in round r (if two nodes broadcast the same token, they cannot learn a new token from each other). Consequently, for the nodes $v_1, \ldots, v_\ell$, we have $x_{v_i}(r) \neq x_{v_j}(r)$ for all $i \neq j$. The fact that the edge $\{v_i, v_j\}$ is non-free in particular implies that for $x_{v_i}(r) \notin R_{v_j}$ or $x_{v_j}(r) \notin R_{v_i}$. The probability for this to happen is $1 - \Pr(x_{v_i}(r) \in R_{v_j}) \cdot \Pr(x_{v_j}(r) \in R_{v_i}) = 3/4$. Hence, each edge is non-free with probability at most $3/4$. Also note that because $x_{v_i}(r) \neq x_{v_j}(r)$ for $i \neq j$ and because each token is independently added with probability $1/2$ to each R_v, the events $\left\{x_{v_i}(r) \notin R_{v_j} \vee x_{v_j}(r) \notin R_{v_i}\right\}$ are independent for different pairs (v_i, v_j). Hence, for a specific set of nodes $\{v_1, \ldots, v_\ell\}$, the probability that all the $\binom{\ell}{2}$ edges between these nodes are non-free is at most $(3/4)^{\binom{\ell}{2}}$. To upper bound the probability that there can be some set of ℓ nodes with only non-free edges in some round, we can apply a union bound over all possible sets S of ℓ nodes and all possible $x_v(r)$ for $v \in S$. We

therefore get

$$\begin{aligned}\Pr(\text{set of } \ell \text{ nodes with only non-free edges is possible}) &\leq \binom{n}{\ell} \cdot k^\ell \cdot \left(\frac{3}{4}\right)^{\binom{\ell}{2}} \\ &\leq n^\ell \cdot k^\ell \cdot \left(\frac{3}{4}\right)^{(\ell-1)^2/2} \\ &= e^{\ell \ln(nk) - \frac{(\ell-1)^2}{2}\ln(4/3)}.\end{aligned}$$

If we choose $\ell \geq C \cdot \ln(nk)$ for a sufficiently large constant C, the above probability is smaller than $1/n^2$. Hence, with probability at least $1 - 1/n^2$, it is not possible to find a round r and a set of ℓ nodes $\{v_1, \ldots, v_\ell\}$ such that all edges $\{v_i, v_j\}$ for $i \neq j$ are non-free in round r.

It remains to show that this probability is still large enough if we also condition on the fact that all the sets R_v are of size at most $|R_v| \leq 2k/3$. The number of tokens in R_v is binomially distributed with parameters k and $1/2$ and we can therefore use the Chernoff bound introduced in Chapter 10 to bound the probability that some set R_v becomes too large. For any $\delta > 0$, we have

$$\Pr(|R_v| \geq (1+\delta) \cdot \mathbb{E}[|R_v|]) = \Pr\left(|R_v| \geq (1+\delta) \cdot \frac{k}{2}\right) \leq e^{-\frac{\min\{\delta,\delta^2\}}{3} \cdot \frac{k}{2}}.$$

Setting $\delta = 1/3$, we get that for all $v \in V$, $\Pr(|R_v| \geq 2k/3) \leq e^{-k/54}$. After adding all free edges, the number of resulting components is definitely at most k (nodes broadcasting the same token are connected by a free edge). Since, we already assumed that $\ell \geq C \cdot \ln(nk)$ for a sufficiently large constant C, we can also assume that $k \geq C \cdot \ln(n)$ for a sufficiently large constant C. For C large enough, the probability for having an R_v of size $|R_v| > 2k/3$ is therefore also at most $1/n^2$. A union bound then gives that with probability at most $2/n^2$, we either have a too large R_v or we can have an ℓ-clique of only non-free edges in some round. Hence, with probability at least $1 - 2/n^2$, neither of these happens and the claim of the lemma holds. As $1 - 2/n^2 > 0$ (for any $n > 1$), there are sets R_v of size at most $2k/3$ which guarantee that in every round, we need to add at most $O(\log(nk))$ non-free edges. □

Theorem 9.7. *Even in the synchronous start case, every deterministic token-forwarding algorithm which forwards at most token per node and round needs at least $\Omega(nk/\log(nk))$ rounds to solve the k-token dissemination problem.*

Proof. The theorem follows almost immediately from Lemma 9.6. In a round r, nodes can only learn new tokens over non-free edges (assuming that if node v learns a token in R_v it does not count as learning a new token). By Lemma 9.6, there are sets R_v of size at most $2k/3$ such that the total number of new tokens learnt in a round ist at most $O(\log(nk))$. Note that if all sets R_v are of size at most $2k/3$, if each token is initially only known by one node, on average all nodes still need to learn essentially a third of all tokens. Hence, together, all nodes need to learn $\Omega(nk)$ tokens which are not in the respective R_v sets. As only $O(\log(nk))$ tokens can be learnt in each round, the lower bound follows. □

Remarks:

- The above lower bound can be generalized to algorithms where each node is allowed to forward s tokens in each round. The lower bound then becomes $\Omega\big(\frac{nk}{s\log(nk)}\big)$.
- Note that while the lower bound only holds for deterministic algorithms and token-forwarding algorithms, it holds even in the presence of a central scheduler which is allowed to see the states of all nodes at the beginning of a round and to decide what each node has to forward in that round.

Chapter Notes

The dynamic network model and the problem definitions that we used in this chapter have been introduced by Kuhn, Oshman, and Lynch in [KLO10]. However, similar dynamic network models have been used before, in particular by O'Dell and Wattenhofer in [OW05] and by Avin et al. in [AKL08]. For a summary on related dynamic network model, we refer to the survey [KO11]. The $\Omega(nk/\log(nk))$ token dissemination lower bound was proven by Dutta et al. in [DPRS13]. The lower bound has been generalized to settings where more than one token can be sent in one message or where the dynamic graph is T-interval connected by Haeuper and Kuhn in [HK12].

Bibliography

[AKL08] Chen Avin, Michal Koucký, and Zvi Lotker. How to explore a fast-changing world (cover time of a simple random walk on evolving graphs). In *Proc. of 35th Coll. on Automata, Languages and Programming (ICALP)*, pages 121–132, 2008.

[DPRS13] Chinmoy Dutta, Gopal Pandurangan, Rajmohan Rajaraman, and Zhifeng Sun. On the complexity of information spreading in dynamic networks. In *Proc. 24th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2013.

[HK12] B. Haeupler and F. Kuhn. Lower bounds on information dissemination in dynamic networks. In *Proc. 26th Symp. on Distributed Computing (DISC)*, pages 166–180, 2012.

[KLO10] Fabian Kuhn, Nancy Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In *Proc. 42nd ACM Symp. on Theory of Computing (STOC)*, pages 513–522, 2010.

[KO11] F. Kuhn and R. Oshman. Dynamic networks: Models and algorithms. *SIGACT News*, 42(1):82–96, 2011.

[OW05] R. O'Dell and R. Wattenhofer. Information dissemination in highly dynamic graphs. In *Proc. of 9th Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, pages 104–110, 2005.