HAL
open science

Physical expander in Virtual Tree Overlay

Taisuke Izumi, Maria Potop-Butucaru, Mathieu Valero

To cite this version:

Taisuke Izumi, Maria Potop-Butucaru, Mathieu Valero. Physical expander in Virtual Tree Overlay. DISC 2011 - 25th International Symposium on Distributed Computing, Sep 2011, Rome, Italy. pp.82-96, 10.1007/978-3-642-24100-0_6 . inria-00569098

HAL Id: inria-00569098

https://inria.hal.science/inria-00569098v1

Submitted on 24 Feb 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Physical Expander in Virtual Tree Overlay*

Taisuke Izumi[1], Maria Gradinariu Potop-Butucaru[2], and Mathieu Valero[2]

[1] Graduate School of Engineering, Nagoya Institute of Technology, Japan
t-izumi@nitech.ac.jp
[2] Université Pierre et Marie Curie - Paris 6, LIP6 CNRS 7606, France
maria.gradinariu@lip6.fr, mathieu.valero@gmail.com

Abstract. In this paper, we propose a new construction of constant-degree expanders motivated by their application in P2P overlay networks and in particular in the design of robust trees overlay.
Our key result can be stated as follows. Consider a complete binary tree T and construct a random pairing Π between leaf nodes and internal nodes. We prove that the graph G_Π obtained from T by contracting all pairs (leaf-internal nodes) achieves a constant node expansion with high probability. The use of our result in improving the robustness of tree overlays is straightforward. That is, if each physical node participating to the overlay manages a random pair that couples one virtual internal node and one virtual leaf node then the physical-node layer exhibits a constant expansion with high probability. We encompass the difficulty of obtaining this random tree virtualization by proposing a local, self-organizing and churn resilient uniformly-random pairing algorithm with $O(\log^2 n)$ running time. Our algorithm has the merit to not modify the original tree virtual overlay (we just control the mapping between physical nodes and virtual nodes). Therefore, our scheme is general and can be applied to a large number of tree overlay implementations. We validate its performances in dynamic environments via extensive simulations.

1 Introduction

Background and Motivation P2P networks are appealing for sharing/diffusing/searching resources among heterogeneous groups of users. Efficiently organizing users in order to achieve these goals is the main concern that motivated the study of overlay networks. In particular, *tree overlays* recently becomes an attractive class of overlay networks because efficient implementations of various communication primitives in P2P systems tied to the hierarchical and acyclic properties of trees such as content-based publish/subscribe or multicast ([4, 6, 9, 14]). Many P2P and distributed variants of classical tree structures such as B-trees, R-trees or P-trees have been designed so far [1, 5, 7, 8, 17, 18].

* This work is supported in part by Grand-in-Aid for Young Scientists ((B)22700010) of JSPS. Additional support from ANR projects R-Discover, SHAMAN, and ALADDIN.

Because of the dynamic nature of P2P networks, robustness to faults and churn (e.g., frequent node join and leave) is indispensable for the service on the top of them to function properly. A recent trend in measuring the robustness of overlay networks is the evaluation of *graph expansion*. The (node) expansion $h(G)$ of an undirected graph $G = (V_G, E_G)$ is defined as:

$$h(G) = \min_{S \subseteq V_G || S| \leq n/2} \frac{|\partial S|}{|S|},$$

where ∂S is the set of nodes that are adjacent to a node in S but not contained in S. The implication of node expansion is that the deletion of at least $h(G) \cdot k$ nodes is necessary to disconnect a component of k nodes in G. That is, graphs with good expansion are hard to be partitioned into a number of large connected components. In this sense, the expansion of a graph can be seen as a good evaluation of its resilience to faults and churn. Interestingly, the expansion of tree overlays is trivially $O(1/n)$, which is far from adequate. This weakness to faults is the primary reason why tree overlays are not pervasive in real applications.

Our focus in this paper is to provide the mechanism to make tree overlays robust and suitable to real applications. In particular, we are interested in *generic* schemes applicable to a large class of tree overlays with minimal extra cost: As seen above, there are many variations of tree-based data structures with distinguished characteristics, but their distributed implementations always face the problem how to circumvent the threat of disconnection. Therefore, providing such a generic robustization scheme would offer the substantial benefit for implementing distributed tree-based data structures in a systematic way.

Our contribution Solutions for featuring P2P tree overlays with robustness range from increasing the connectivity of the overlay (in order to eventually mask the network churn and fault) to adding additional mechanism for monitoring and repairing the overlay. However the efficiency of these techniques is shadowed by the extra-cost needed to their implementation in dynamic settings. Moreover, the design of those mechanisms often depends on some specific tree overlay implementation, and thus their generalization is difficult. Therefore, we propose a totally novel approach that exploits the principal of *tree virtualization*. That is, in a tree overlay one physical node may be in charge of several virtual nodes. The core of our approach is to use this mapping between virtual and physical nodes such that the physical-node layer exhibits a good robustness property.

Our primary contribution is the following theorem which is the key in the construction of our random virtualization scheme:

Theorem 1. *Let T be a complete n-node binary tree with duplication of the root node (the duplicated root is seen to be identical to the original root). Then, we can define a bijective function Π from leaf nodes to internal nodes. Let G_Π be the graph obtained from T by contracting pair $(v, \Pi(v))$ for all v* [3]*. Choosing Π uniformly at random G_Π has a constant (node) expansion with high probability.*

[3] The contraction of $(v, \Pi(v))$ means that we contract edge $\{v, \Pi(v)\}$ as if it exists in T.

An immediate consequence of this theorem is that the physical-node layer achieves a constant expansion with high probability if a random chosen couple composed of one leaf and one internal node is assigned to each physical node. It should be noted that our random tree virtualization does not modify the original properties of the tree overlay since we only control the mapping to physical nodes. This feature yields a general applicability of our result to a large class of tree overlay implementations.

The above result relies on the uniform random bijection (i.e. random perfect bipartite matching) between internal and leaf nodes in the tree overlay. Therefore, in order to prove the effectiveness of our proposal in a P2P context we also addressed the construction of random perfect bipartite matching over internal and leaf nodes. Interestingly, we can propose a local and self-organizing scheme based on the parallel random permutation algorithm by Czumaj et. al.[11]. Our scheme allows us to increase the graph expansion to a constant within $O(\log^2 n)$ synchronous rounds (n is the number of physical nodes). The quick convergence of our scheme in dynamic settings is validated through extended simulations.

Roadmap In Section 2, we introduce the relate work mainly in the field of distributed computing. Section 3 presents the proof of our main result. The issue about the distributed implementation of our scheme is explained in Section 4 which includes the simulation result. Finally, Section 5 provides the conclusion and future research issues.

2 Related works

Expander graphs have been studied extensively in many areas of theoretical computer science. A good tutorial can be found in [21]. In the following we restrict our attention to distributed constructions with a special emphasize on specific P2P design.

There are several results about expander construction in distributed settings. Most of those results are based on the distributed construction of random regular graphs, which exhibit a good expansion with high probability. To the best of our knowledge one of the first papers that addressed expander constructions in peer-to-peer settings is [19]. The authors compose d Hamiltonian cycles to obtain a $2d$-regular graph. In [20] the authors propose a fault-tolerant expander construction using a pre-constructed tree overlay. It provides the mechanism to maintain an approximate random d-regular graph under the assumption that the system always manages a spanning tree. The distributed construction of random regular graphs based on a stochastic graph transformation is also considered in [10, 15]. They prove that repeating a specific stochastic graph modification (e.g., swapping the two endpoints of a length-three path) eventually returns a uniformly-random sampling of regular graphs. Since all the previously mentioned algorithms are specialized in providing good expansion, the combination with overlays maintenance is out of their scope. Therefore, these works cannot be easily extended to a generic fault tolerant mechanism in order to improve the

resiliency of a distributed overlay. Contrary to the previous mentioned works, our study can be seen as a way of identifying implicit expander properties in a given topological structure. There are several works along this direction. In [12] the authors propose a self-stabilizing constructions of spanders, which are spanning subgraphs including smaller number of edges than the original graph but having the asymptotically same expansion as the original[4]. Abraham et.al. [2] and Aspnes and Wieder [3] respectively give the analysis of the expansion for some specific distributed data structures (skip graphs and skip b-trees). Recently Goyal et.al. [16] prove that given a graph G, the composition of two random spanning trees has the expansion at least $\Omega(h(G)/\log n)$, where n is the number of node in G. We can differentiate our result from the above works by its generality and the novelty of random tree virtualization concept.

3 The expander property of G_Π

3.1 Notations

For an undirected graph G, V_G and E_G respectively denote the sets of all nodes and edges in G. Given a graph G and a subset of nodes $S \subseteq V_G$, we define Ind(S) to be the subgraph of G induced by S. For a set of nodes S, its complement is denoted by $\overline{S}$. The *node boundary* of a set $S \subseteq V_G$ is defined as a set of nodes in $\overline{S}$ that connect to at least one node in S, which is denoted by ∂S.

Let $T = (V_T, E_T)$ be a binary tree. The sets of leaf nodes and internal nodes for T are respectively denoted by $L(V_T)$ and $I(V_T)$. Given a subset $S \subseteq V_T$, we also define $L(S) = L(V_T) \cap S$ and $I(S) = I(V_T) \cap S$. For a (sub)tree X, the root node of X is denoted by $r(X)$, and the parent of $r(X)$ is denoted by $p(X)$.

3.2 Preliminary results

In the following T denotes a complete binary tree without explicit statement. The root of T is denoted by $r(T)$. We prove several auxiliary results that will be further used in our main result.

Lemma 1. *For any nonempty subset $S \subseteq I(V_T)$ such that* Ind(S) *is connected, $|\partial S| \geq |S| + 1$. In particular, if $r(T) \notin S$ holds, $|\partial S| \geq |S| + 2$.*

The above lemma can be generalized for any (possibly disconnected) subset $S \subseteq I(V_T)$.

Lemma 2. *Given any nonempty subset $S \subseteq I(V_T) \setminus \{r(T)\}$ such that* Ind(S) *of T consists of m connected components, $|\partial S| \geq |S| + m + 1$ holds.*

The following corollary is simply deduced from Lemma 2.

Corollary 1. *Let X be a subtree of T. For any subset $S \subseteq I(V_X)$, $|\partial S \cap V_X| \geq |S \cap V_X|$. In particular, if S is nonempty, we have $|\partial S \cap V_X| \geq |S \cap V_X| + 1$.*

[4] A spander is also called a *sparsifier*.

3.3 Main Result

In what follows, $|L(V_T)|$ is denoted by n for short (i.e., n is the number of nodes in G_Π). We also assume Π is a bijective function from leaf nodes to the set of internal nodes (where the root doubly appears), which is chosen from all $n!$ possible functions uniformly at random. For a subset of nodes $S \subseteq L(V_T)$, we define $\Pi(S) = \{\Pi(u) | u \in S\}$ and $Q_\Pi = S \cup \Pi(S)$.

Provided a subset $S \subseteq L(V_T)$ satisfying $|S| < n/2$, we say a subtree X is *S-occupied* if all of its leaf nodes belong to S. A S-occupied subtree X is *maximal* if there is no S-occupied subtree X containing X as a subtree. Note that two S-occupied maximal subtrees X_1 and X_2 in a common tree T are mutually disjoint and $p(X_1) \neq p(X_2)$ holds because of their maximality. We first show two lemmas used in the main proof.

Lemma 3. *Let X be a maximal S-occupied subtree for a nonempty subset $S \subseteq L(V_T)$. Then, $|(\partial Q_\Pi) \cap V_X| \geq |\overline{Q_\Pi} \cap V_X|/2$ holds.*

Lemma 4. *Given a subset $S \subseteq L(V_T)$ such that $|S| \leq n/2$, let $X_0, X_1, \cdots X_k$ be all maximal S-occupied subtrees and $V_X = \cup_{i=1}^{k} V_{X_i}$. For any $\alpha < 1$, $\Pr(|\Pi(S) \cap I(V_X)| \geq \alpha |I(V_X)|) \leq \binom{|S|}{\alpha(|S|-k)} \left(\frac{(|S|-k)}{n}\right)^{\alpha(|S|-k)}$.*

The implication of the above two lemmas is stated as follows: We are focusing on a subset of boundary nodes ∂Q_Π that are associated with some "hole" (that is, the set of nodes not contained in Q_Π) in S-occupied subtrees. Lemma 3 implies that at least half of the nodes organizing the hole belongs to ∂Q_Π. Lemma 4 bounds the probability that S-occupied subtrees has the hole with size larger or equal to $(1-\alpha)|I(V_X)|$. We also use the following inequality:

Fact 1 (Jensen's inequality) Let f be the convex function, $p_1, p_2, \cdots$ be a series of real values satisfying $\sum_{i=1}^{\infty} p_i = 1$, and $x_1, x_2, \cdots$ be a series of real values. Then, the following inequality holds:

$$\sum_{i=1}^{\infty} p_i f(x_i) \geq f(\sum_{i=1}^{\infty} p_i x_i)$$

We give the proof of the main theorem (the statement is refined).

Theorem 1. *The node expansion of G_Π is at least $\frac{1}{480}$ with probability $1 - o(1)$.*

Proof. To prove the lemma, we show that with high probability, $|\partial S| \geq |S|/480$ holds for any subset $S \subseteq V_{G_\Pi}$ such that $|S| \leq n/2$. In the proof, we identify $L(T)$ and V_{G_Π} as the same set, and thus we often refer S as a subset of $L(V_T)$ without explicit statement. Given a set S, let k be the number of maximal S-occupied subtrees, $\mathcal{X} = \{X_1, X_2, \cdots, X_k\}$ be all maximal S-occupied trees, and $V_X = \cup_{i=1}^{k} V_{X_i}$. We also define $P = \{p(X_i) | X_i \in \mathcal{X}\}$. Throughout this proof, we omit the subscript Π of Q_Π. For a subset $Y \subseteq V_T$, let $Q(Y) = Q \cap Y$ and $q(Y) = |Q(Y)|$ for short.

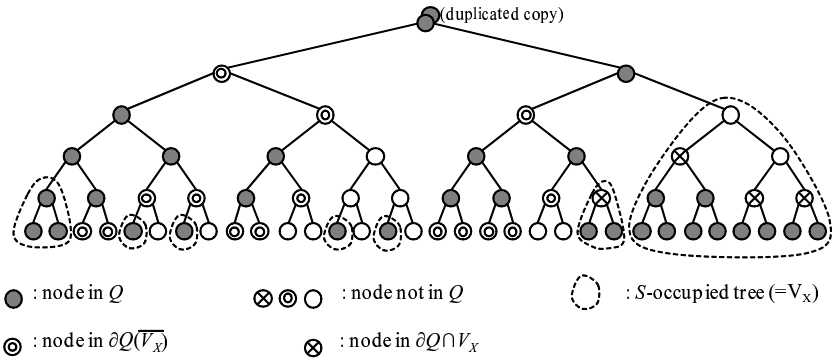

Fig. 1: Illustration of the set $(\partial S) \cap V_X$, $\partial Q(\overline{V_X})$, and their boundaries. in the proof of Theorem 1.

The goal of this proof is to show that $|\partial Q| \geq |S|/240$ holds with high probability in T for any given S. It follows that $|\partial S| \geq |S|/480$ holds in G_Π. To bound $|\partial Q|$, we first consider the cardinality of two subsets $(\partial Q) \cap V_X$ and $\partial(Q \cap \overline{V_X})$ $(= \partial Q(\overline{V_X}))$. See Figure 1 for an example. While these subsets are not mutually disjoint, only the roots of S-occupied subtrees can be contained in $\partial Q(\overline{V_X})$. It implies

$$|(\partial Q \cap V_X) \cap \partial Q(\overline{V_X})| \leq q(P). \tag{1}$$

Lemma 2 and 3 lead to the following inequalities:

$$\begin{aligned} |(\partial Q) \cap V_X| &\geq \frac{1}{2}\left(\sum_{i=1}^{k} |\overline{Q} \cap V_{X_i}|\right) \\ &\geq (|S| - k - q(V_X))/2. \end{aligned} \tag{2}$$

$$\begin{aligned} |\partial(Q(\overline{V_X})| &\geq |Q(\overline{V_X})| + 1 \\ &\geq (|S| - 1) - q(V_X) + 1 = |S| - q(V_X). \end{aligned} \tag{3}$$

By inequalities 1, 2, and 3, we can bound the size of ∂Q as follows:

$$\begin{aligned} |\partial Q| &\geq |(\partial Q) \cap V_X| + |\partial Q(\overline{V_X})| - q(P) \\ &\geq (|S| - k - q(V_X))/2 + |S| - q(V_X) - q(P) \qquad (4) \\ &\geq 3(|S| - k - q(V_X))/2 + k - q(P). \qquad (5) \end{aligned}$$

We consider the following two cases according to the value of k:
(Case 1) $k > |S|/16$: We show $|\partial Q| > |S|/240$ holds for any Π. If $q(P) \leq 12k/13$ holds, we have $|\partial Q| \geq k/13 \geq |S|/240$ from inequality 5 because of $q(V_X) \leq |S| - k$. Furthermore, if $|S| - q(V_X) - q(P) \geq |S|/240$, we have $|\partial Q| \geq |S|/240$ from inequality 4. Thus, in the following argument, we assume $q(P) > 12k/13$ and

$|S| - q(V_X) - q(P) < |S|/240$. Consider the subgraph H of T induced by $Q(P)$. We estimate the number of connected components in H to get a bound of $|\partial V_H|$ in T. Letting $\mathcal{C} = \{C_1, C_2, C_3, \cdots C_m\}$ be the set of connected components of H, we associate each leaf node u in a S-occupied subtree X_i with the component in $\mathcal{C}$ containing $p(X_i)$ (the node is associated with no component if $p(X_i)$ is not in $Q(P)$). Since each node $v \in H$ has one child belonging to V_X, each component in H forms a line graph monotonically going up to the root. Thus, if a component C_i has j nodes $u_1, u_2, \cdots u_j$, which are numbered from the leaf side, each node u_h $(1 \leq h \leq j)$ has a child as the root of S-occupied trees having at least 2^{h-1} leaf nodes (recall that T is a complete binary tree). It follows that the number of nodes in S associated with C_i is $\sum_{h=1}^{j} 2^{h-1} \geq 2^j - 1$. Letting l_i be the number of components in $\mathcal{C}$ consisting of i nodes, we have:

$$\begin{aligned}
|S|/m &\geq \sum_{i=0}^{n} \frac{l_i}{m} \left(2^i - 1\right) \\
&\geq 2^{\sum_{i=0}^{n} i \cdot (l_i/m)} - 1 \\
&\geq 2^{\frac{12k}{13m}} - 1 \\
&\geq 2^{\frac{3|S|}{52m}} - 1,
\end{aligned}$$

where the second line is obtained by applying Jensen's inequality. To make the above inequality hold, the condition $|S|/m \leq 120 \Leftrightarrow m \geq |S|/120$ is necessary. Next, we calculate how many nodes in $\partial V_H \cap \overline{V_X}$ is occupied by Q. From the definition of H, $Q(\partial V_H)$ does not contain any node in P (if a node $v \in P$ is contained, it will be a member of V_H and thus not in ∂V_H). Thus, any node in ∂V_H is a member of V_X, $\overline{Q} \cap P$, or $\overline{V_X \cup P}$. Let $Y = Q(\overline{V_X \cup P} \cap \partial V_H)$ and $y = |Y|$ for short. Since any node in $\overline{Q} \cap P$ is not contained in Q and the cardinality of $\partial V_H \cup V_X$ can be bounded by $q(P)$ (as the roots of S-occupied trees), we have the following bound from Lemma 2:

$$\begin{aligned}
|\partial Q| &\geq |\partial V_H \setminus Q| \\
&\geq |(\partial V_H) \cap (\overline{Q} \cap P)| \\
&\geq |\partial V_H| - |(\partial V_H) \cap V_X| - |(\partial V_H) \cap (\overline{V_X \cup P})| \\
&\geq |\partial V_H| - q(P) - y \\
&\geq q(P) + m + 1 - q(P) - y \\
&\geq m - y.
\end{aligned}$$

The illustration explaining this inequality is shown in Figure 2. Since Y, $Q(P)$ and $Q(V_X)$ are mutually disjoint, we obtain $y + q(V_X) + q(P) \leq |S| \Leftrightarrow y \leq |S| - q(V_X) - q(P) < |S|/240$. Consequently, we obtain $|\partial Q| \geq |S|/240$.

(Case 2) $k \leq \frac{|S|}{16}$: In the following argument, given a set S satisfying $k < |S|/16$, we bound the probability $\Pr(|\partial Q| < |S|/32)$. From the inequality 5 and the fact of $k - q(P) \geq 0$, we have $|\partial Q| \geq 3(15|S|/16 - q(V_X))/2$. To be $|\partial Q| \leq |S|/32$, we need $3(15|S|/16 - q(V_X))/2 \leq |S|/32 \Leftrightarrow q(V_X) \geq 11|S|/12$. Thus, from Lemma

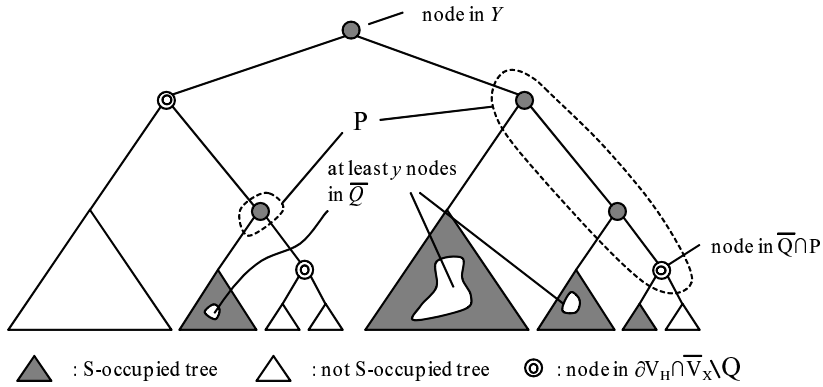

Fig. 2: Illustration of the boundary $(\overline{V_H} \cap \overline{V_X})\backslash$ in the proof of Theorem 1.

4, we can bound the probability as follows:

$$\begin{aligned}\Pr(|\partial Q| < |S|/32) &\leq \Pr(|\Pi(S)| \geq 11|I(V_X)|/12) \\ &\leq \binom{|S|}{11(|S|-k)/12}\left(\frac{|S|-k}{n}\right)^{11|S|/12} \\ &\leq \binom{|S|}{(|S|+k)/12}\left(\frac{|S|}{n}\right)^{11|S|/12}.\end{aligned}$$

Fixing k and $|S|$, we look at the number of possible choices of S. Since we can determine a S-occupied subtree X_i by choosing one node in T as its root, the set S is determined by choosing k nodes from all nodes in T. Thus, the total number of subset S organizing at most $|S|/16$ S-occupied subtrees are bounded by $\sum_{i=1}^{|S|/16}\binom{2n}{i} \leq \frac{|S|}{16}\binom{2n}{|S|/16}$. Summing up this bound for any $|S| < n/2$. The total number is bounded by $\sum_{|S|=1}^{n/2}\frac{|S|}{16}\binom{2n}{|S|/16}$. Using the union bound and a well-known bound $\binom{n}{m} \leq (ne/m)^m$, we have:

$$\begin{aligned}&\Pr\left(\bigcup_{S\subseteq L(V_T)||S|\leq n/2} |\partial Q| < \frac{|S|}{32}\right) \\ &\leq \sum_{|S|=1}^{n/2}\frac{|S|}{16}\binom{2n}{|S|/16}\binom{|S|}{(|S|+k)/12}\left(\frac{|S|}{n}\right)^{11|S|/12}. \\ &= o(1).\end{aligned}$$

All the details of the previous calculation are provided in the Appendix. The theorem is proved. □

4 Distributed Construction of G_Π

To prove the impact of Theorem 1 in P2P settings we have to construct scalably a random bijection (i.e., random perfect bipartite matching) between internal and leaf nodes in tree overlays. In this section, we show that this distributed construction is possible with nice self-$*$ properties. That is, our scheme is totally-distributed, uses only local information, and is self-healing in the event of nodes joins and leaves. In the following we state the computational model and our network assumptions.

4.1 Computational Model

We consider a set of virtual nodes (peers) distributed over a connected physical network. Virtual nodes are structured in a binary tree overlay. Each physical node managing a virtual node v can communicate with any physical node managing v's neighbors in the overlay. The communication is synchronous and round-based. That is, the execution is divided into a sequence of consecutive rounds. All messages sent in some round are guaranteed to be received within the same round.

We assume that each physical node manages exactly one internal node and one leaf node in the virtual overlay. Moreover, we also assume that the tree is balanced. Note that these assumptions are not far from practice. Most of distributed tree overlay implementations embed balancing schemes. The preservation of matching structure is easily guaranteed by employing the strategy that one physical node always join as two new nodes. We can refer as an example the join/leave algorithm in [13], which is based on the above strategy and generally applicable to most of binary-tree overlays.

4.2 Uniformly-Random Matching Construction

The way leaf and internal nodes are matched via a physical node is generally dependent on the application requirements and is rarely chosen uniformly at random. That is, the implicit matching offered by the overlay may be extremely biased and the expansion factor computed in the previous section may not hold. Fortunately, the initial matching can be quickly "mixed" to obtain a uniformly-random matching. To this end we will extend the technique proposed by Czumaj et. al.[11] for fast random permutation construction to distributed scalable matchings in tree overlays. The following stochastic process rapidly mixes the sample space of all bipartite matchings between leafs and internal nodes:

1. Each leaf node first tosses a fair coin and decides whether it is active or passive.
2. Each active node randomly proves a leaf node and sends a matching-exchange request.
3. The passive node receiving exactly one matching-exchange request accepts it, and establishes the agreement to the sender of the request.

4. The internal nodes managed by the agreed pair are swapped.

Note that the above process is performed by all leaf nodes concurrently in a infinite loop. Following the analysis by Czumaj et. al.[11], the mixing time of the above process is $O(\log n)$.

The only point that may create problems in distributed P2P settings is the second step. For that point, we can propose a simple solution to implement the random sampling mechanism with $O(\log n)$ time and message complexity. The algorithm is as follows: First, the prober sends a token to the root. From the root, the token goes down along the tree edges by selecting with equal probability one of its children. When the token reaches a leaf node, the destination is returned as the probe result.

Overall, the distributed scalable algorithm for constructing a random bipartite matching takes $O(\log^2 n)$ time. In the following subsection, we evaluate the performances of the above scheme face to churn.

4.3 Experimental Evaluation in Dynamic Environment

In this section, we experimentally validate the performance of our approach by simulation. In the simulation scenario the following four phases are repeated.

Node join We assume that a newly-joining node knows the physical address of some leaf node u in the network. Let v and v' be the leaf and internal nodes that will be managed by the newly-joining physical node. The node u is replaced by a newly internal node v'. Then v and u becomes children of v'.

Node leave The adversary chooses a number of nodes to make them leave. Since it is hard to simulate worst-case adversarial behavior, we adopt a heuristic strategy: given a physical node v with leaf v_L and internal node v_I, let $h(v)$ be the height of the smallest subtree containing both v_L and V_I. Intuitively, the physical node v with higher $h(v)$ has much contributition for avoiding the node boundary to be contained in a small subtree containing v_L. Following this intuition, the adversary always makes the node v with highest $h(v)$ leave.

Balancing Most of tree-based overlay algorithms have some balancing mechanism. While the balancing mechanism has a number of variations, we simply assume a standard rotation mechanism. After a number of node joins and leaves, the tree is balanced by standard rotation operation.

Matching Reconstruction We run once the matching-mixing process described in the previous section.

Since exact computation of node expansion is coNP-complete, we monitor the second smallest eigenvalue λ of the graph's Laplacian matrix, which has a strong corelation to the node expansion: a graph with the second smalles eigenvalue λ is a $\lambda/2$-expander. In the following we propose our simulations results first in a churn free setting then in environments with different churn levels. Due to the space limitation the churn-free simulations are defered to the Anexe section.

Without churn We ran 100 simulations of 100 rounds with 512 nodes and no churn. The value of λ is calculated at the begin of each round. Those simulations tends to emphasize what is "expectable" from the mixing protocol and some of its dynamic properties.

λ varies from 0.263 to 0.524 with an average value of 0.502 and a standard deviation of 0.017. The low standard deviation and the closeness of average and maximum reached values of λ indicates that the minimum is rarely reached. Basically it is obtained when most nodes become responsible of internal nodes that are "close" from their leaves. Intuitively if each node is responsible of an ancestor of its leaf, there is no additionnal links between the left and the right subtrees of the root. In that case we do not take benefit of mixing and get bad expansion properties inherited from tree structures.

Churn prone environments. We ran 100 simulations of 100 rounds with 512 nodes and a given rate of churn. Time is divided in seven rounds groups. During the first round a given percentage of new nodes join the system. During the second a given percentage of nodes leave the system. During the third round the tree is balanced and the mixing protocol is run. During other rounds the mixing protocol is run. λ is measured at the end of each round. Nodes gracefully leave the system. Those simulations investigate the impact of churn on λ and how fast our mixing protocol restores a stable configuration.

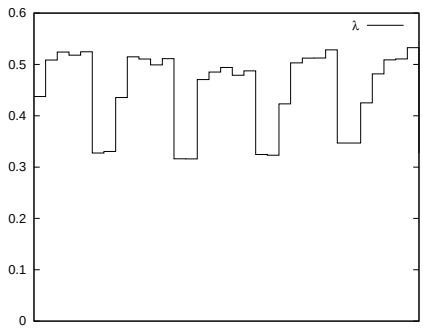

Fig. 3: λ over time with 10% of churn

Figure 3 shows the evolution of λ over time in presence of 10% of churn (10% is relative to the initial number of nodes). Each step stands for a round. From a stable configuration where λ oscillates between 0.52 and 0.48, it drops down to 0.4 every seven rounds due to arrivals and departures. The structure is sensitive to churn in the sense that it significantly decreases the value of λ. But on the other hand, proposed mixing protocol converges fast. It needs two rounds to reach the average "expectable" value of λ.

Figure 4 shows the evolution of λ over time in presence of 30% of churn (30% is relative to the initial number of nodes). Each step stands for a round.

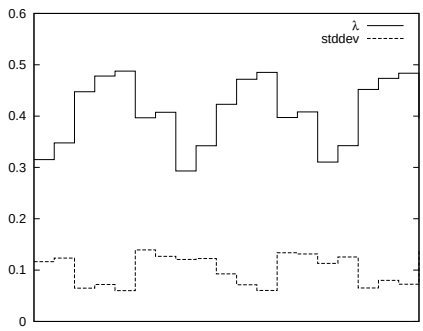

Fig. 4: λ over time with 30% of churn

Basically churn increases stretches the curve; down part are downer around 0.3, high values are quite stable around 0.45 and fluctuations are widespread. While the previous one 3 gives a good overview of the global behaviour of the protocol facing churn, this figure emphases some interesting details. First it emphasizes that mixing protocol is not monotonic; it might decrease λ. Second, the impact of arrivals and departures are distinct. Moreover the magnitude of their impact is not predictable because the selection of bootstrap node is random. Starting from a stable situation arrivals will always decrease λ. With the proposed join mechanism, a new comer is weakly connected to the rest of the system. Starting from a stable situation gracefull departures will almost always decrease λ. But with the proposed leave mechanism their impact is more subtle because; they can largely modify the tree balance which also implies links exchanges. In some rare cases those exchanges (which could be thought as side effect shuffles of links) or the departure of weakly connected nodes could increase λ.

5 Concluding Remarks

We proposed for the first time in the context of overlay networks a generic scheme that transforms any tree overlay in an expander with constant node expansion with high probability. More precisely, we prove that a uniform random tree virtualization yields a node expansion at least $\frac{1}{480}$ with probability $1-o(1)$. Second, in order to demonstrate the effectiveness of our result in the context of real P2P networks we further propose and evaluate in different churn scenario a simple scheme for uniformly random tree virtualization in $O(log^2 n)$ running time. Our scheme is totally distributed and uses only local information. Moreover, in the event of nodes join/leave or crash our scheme is self-healing.

The virtualization scheme itself is a promising approach and provides several interesting questions. We enumerate the open problems related to our result:

- **Better analysis of matching convegence**: As the simulation result exhibits, the convergent value of expansion computed from λ is considerably larger than the theoretical bound. In addition, the convegence time is also

faster than the theoretical bound. Finding improved bounds for both of the expansion and convegence time is an open problem.

- **Effective utilization of expander propoerty**: In addition to fault resiliency, expander graphs also offer the rapidly-mixing property of random walks on the graph. That is, MCMC-like sampling method effectively runs on our scheme. It is an interesting research direction that we utilize expansion property for implementing some statistical operation over distributed datas or query load balancing.
- **Application of virtualization scheme to other overlays**: The virtualization scheme is simple and generic, and thus we can apply it to other well-known overlay algorithms such as Chord or Pastry. Clarifying the class of overlay networks where the virtualizaton scheme efficiently works is a challenging problem.

References

1. K. Aberer, P. Cudre-Mauroux, A. Datta, Z. Despotovic, M. Hauswith, M. Punceva, and R. Schmidt. P-Grid: a self-organizing access structure for p2p information. In *CoopIS*, 2001.
2. I. Abraham, J. Aspnes, and J. Yuan. Skip b-trees. In *In Ninth International Conference on Principles of Distributed Systems (pre-proceedings*, pages 284–295, 2005.
3. J. Aspnes and U. Wieder. The expansion and mixing time of skip graphs with applications. *Distributed Computing*, 21(6):385–393, Oct. 2008.
4. S. Baehni, P. T. Eugster, and R. Guerraoui. Data-aware multicast. In *DSN*, 2004.
5. S. Bianchi, A. K. Datta, P. Felber, and M. Gradinariu. Stabilizing peer-to-peer spatial filters. In *ICDCS*, 2007.
6. P. C. and V. K. A topologically-aware overlay tree for efficient and low-latency media streaming. In *QSHINE*, 2009.
7. R. Y. W. C. Zhang, A. Krishnamurthy. Brushwood: Distributed trees in peer-to-peer systems. 2005.
8. E. Caron, F. Desprez, C. Fourdrignier, F. Petit, and C. Tedeschi. A repair mechanism for fault-tolerance for tree-structured peer-to-peer systems. In *HiPC*, 2006.
9. M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. I. T. Rowstron, and A. Singh. Splitstream: High-bandwidth content distribution in cooperative environments. In *IPTPS*, 2003.
10. C. Cooper, M. Dyer, and A. Handley. The flip markov chain and a randomising p2p protocol. In *Proceedings of the 28th ACM symposium on Principles of distributed computing*, pages 141–150. ACM, 2009.
11. A. Czumaj and M. Kutylowski. Delayed path coupling and generating random permutations. *Random Struct. Algorithms*, 17(3-4):238–259, 2000.
12. S. Dolev and N. Tzachar. Spanders: distributed spanning expanders. In *SAC*, pages 1309–1314, 2010.
13. C. du Mouza, W. Litwin, and P. Rigaux. Sd-rtree: A scalable distributed rtree. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 296–305. IEEE, 2007.
14. P. T. Eugster, R. Guerraoui, S. B. Handurukande, P. Kouznetsov, and A.-M. Kermarrec. Lightweight probabilistic broadcast. *ACM Trans. Comput. Syst.*, 21(4), 2003.

15. T. Feder, A. Guetz, M. Mihail, and A. Saberi. A local switch Markov chain on given degree graphs with application in connectivity of peer-to-peer networks. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 69–76. IEEE, 2006.
16. N. Goyal, L. Rademacher, and S. Vempala. Expanders via random spanning trees. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 576–585. Society for Industrial and Applied Mathematics, 2009.
17. H. V. Jagadish, B. C. Ooi, and Q. H. Vu. Baton: a balanced tree structure for peer-to-peer networks. In *VLDB*, 2005.
18. H. V. Jagadish, B. C. Ooi, Q. H. Vu, R. Zhang, and A. Zhou. Vbi-tree: A peer-to-peer framework for supporting multi-dimensional indexing schemes. In *ICDE*, 2006.
19. C. Law and K.-Y. Siu. Distributed construction of random expander networks. In *In IEEE Infocom*, pages 2133–2143, 2003.
20. M. K. Reiter, A. Samar, and C. Wang. Distributed construction of a fault-tolerant network from a tree. In *SRDS*, pages 155–165, 2005.
21. H. S., L. N., and W. A. Expendar graphs and their applications. *American Mathematical Society*, (43), 2006.

6 Omitted proofs

We explain the proof details omitted in the main body.

6.1 Proof of Lemma 1

Proof. We prove the case of $r(T) \notin S$ by induction on the cardinality of S. (**Basis**) If $|S| = 1$, the lemma clearly holds because every internal node except for the root of T has degree three. (**Inductive step**) Suppose as the induction hypothesis that $|\partial S| \geq |S| + 2$ holds for any connected set S of size k. we prove that for any $v \in I(V_T)$ that is connected to a node in S, $|\partial(S \cup \{v\})| \geq |S \cup \{v\}| + 2$ holds. For short, let $S' = S \cup \{v\}$. Since S is connected, v is connected to exactly one node in S. In other words, node v has two neighbors of v in neither S nor ∂S, which are elements of $\partial S'$. In contrast, v is an element of ∂S but not in $\partial S'$. It follows that $\partial S' \geq \partial S - 1 + 2$ holds. From the induction hypothesis, we obtain $\partial S' \geq \partial S + 1 \geq |S| + 1 + 2 = |S'| + 2$. The case $r(T) \in S$ is obviously deduced from the case of $r(T) \notin S$. The lemma is proved. □

6.2 Proof of Lemma 2

Proof. Let $C_1, C_2, \cdots C_j, \cdots C_m$ be the set of connected components in $\mathrm{Ind}(S)$. In the case of $r(T) \in S$, we assume $r(T) \in C_1$ without loss of generality. We prove the lemma by induction on j. (**Basis**) It holds from Lemma 1 (**Inductive step**) Suppose $|\partial S| \geq |S| + j + 1$ as the induction hypothesis. Consider adding a new component C_{j+1} into S. Let c be the number of nodes in C_{j+1}. Since $r(T) \notin C_{j+1}$, from Lemma 1, $|\partial V_{C_{m+1}}| \geq c+2$ holds. At most one node is shared by ∂S and $\partial V_{C_{m+1}}$, we have $|\partial S \cup V_{C_{j+1}}| \geq |S|+j+1+c+2-1 \geq (|S|+c)+(j+1)$. The lemma is proved. □

6.3 Proof of Lemma 3

Proof. We omit the subscript Π of Q_Π for short. We divides $\overline{Q} \cap V_X$ into three mutually-disjoint subset $\overline{S_1}$, $\overline{S_2}$, and $\overline{S_3}$: Let $\overline{S_1} \subseteq V_X \cap \overline{Q}$ be the set of nodes that have no neighbor belonging to $Q \cap V_X$, $\overline{S_2} = \partial \overline{S_1} \cap \overline{Q}$, and $\overline{S_3} = (\overline{Q} \cap V_X) \setminus (\overline{S_1} \cup \overline{S_2})$ (see Figure 5). Since X is S-occupied, $\overline{S_2}$ consists only of internal nodes. Thus, from Corollary 1, $|\overline{S_2}| = |\partial \overline{S_1}| \geq |\overline{S_1}|$ holds. By the definition of $\overline{S_2}$ and $\overline{S_3}$, $\overline{S_2} \subseteq (\partial Q) \cap V_X$ and $\overline{S_3} \subseteq (\partial Q) \cap V_X$ hold. Consequently, we have

$$\begin{aligned} 2|\partial Q \cap V_X| &\geq 2(|\overline{S_2}| + |\overline{S_3}|) \\ &\geq |\overline{S_2}| + 2|\overline{S_3}| + |\overline{S_1}| \\ &\geq |\overline{Q} \cap V_X|. \end{aligned}$$

The lemma is proved. □

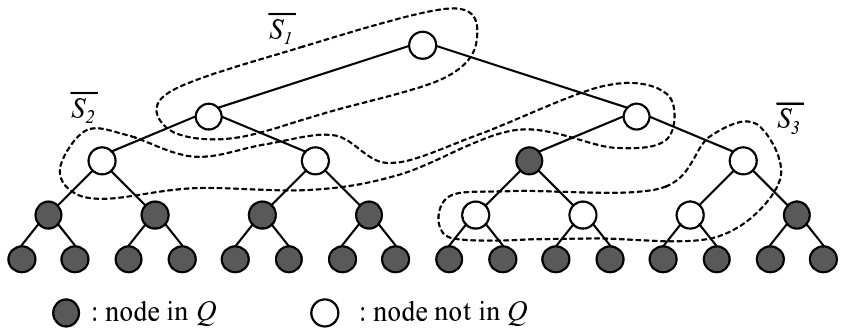

Fig. 5: Illustration of $\overline{S_1}$, $\overline{S_2}$, and $\overline{S_3}$ in the proof of Lemma 3.

6.4 Proof of Lemma 4

Proof. Let $Z = I(V_X)$ for short. First, we fix a subset $S' \subseteq$ of S with cardinality $\alpha|Z|$, and consider the probability that $\Pi(S') \subseteq Z$ holds. Without loss of generality, we can regard Π as a permutation on Z whose first $|S|$ elements are mapped to Z. Thus, to compute the probability, it is sufficient to count the number of permutations where any element in S' appears at the first $|Z|$ elements of Π. Dividing the counted number by $n!$, the probability can be calculated as follows:

$$\begin{aligned}\Pr(S' \subseteq X) &= \frac{1}{n!}\binom{n-|S'|}{|Z|-|S'|}|Z|!(n-|Z|)! \\ &= \frac{|Z|(|Z|-1)(|Z|-2)\cdots(|Z|-\alpha|Z|+1)}{n(n-1)(n-2)\cdots(n-\alpha|Z|+1)} \leq \left(\frac{|Z|}{n}\right)^{\alpha|Z|}.\end{aligned}$$

Using the union bound, we can obtain the following bound:

$$\begin{aligned}\Pr(\Pi(S) \geq \alpha|Z|) &\leq \Pr(\bigcup_{S' \subset S | S' = \alpha|Z|} S' \subseteq Z) \\ &\leq \binom{|S|}{\alpha|Z|}\left(\frac{|Z|}{n}\right)^{\alpha|Z|}.\end{aligned}$$

Since $|Z| = |S| - k$ holds, the lemma is proved. □

6.5 The Calculation Details in the Proof of Theorem 1

We describe the detailed calculation to lead the last inequality in the proof of Theorem 1.

$$\Pr\left(\bigcup_{S\subseteq L(V_T)||S|\le n/2} |\partial Q| < \frac{|S|}{32}\right)$$
$$\le \sum_{|S|=1}^{n/2} \frac{|S|}{16}\binom{2n}{|S|/16}\binom{|S|}{(|S|+k)/12}\left(\frac{|S|}{n}\right)^{11|S|/12}.$$

Using the bound $\binom{n}{m} \le (ne/m)^m$ and the condition $k < |S|/16$,

$$\le \sum_{|S|=1}^{n/2} \frac{|S|}{16}\left(\frac{32en}{|S|}\right)^{|S|/16}\left(\frac{192e}{17}\right)^{17|S|/192}\left(\frac{|S|}{n}\right)^{11|S|/12}$$
$$\le \sum_{|S|=1}^{n/2} \frac{|S|}{16}\left((32e)^{1/16}(192e/17)^{17/192}\right)^{|S|}\left(\frac{|S|}{n}\right)^{(11/12-1/16)|S|}$$

By numeric calculation, we have $\log((32e)^{1/16}(192e/17)^{17/192}) \le 0.841$ and $(11/12 - 1/16) \ge 0.854$. Thus,

$$\le \sum_{|S|=1}^{n/2} \frac{|S|}{16}2^{0.841|S|}\left(\frac{|S|}{n}\right)^{0.854|S|}$$
$$= o(1).$$