Austrian Journal of Statistics
July 2019, Volume 48, 17–33.
http://www.ajs.or.at/
doi:10.17713/ajs.v48i5.880

A Metropolis-Hastings Sampling of Subtrees in Graphs

Abdelrahman Eid
Université de Lille

Hiroshi Mamitsuka
Kyoto University

Nicolas Wicker
Université de Lille

Abstract

This article presents two methods to sample uniform subtrees from graphs using Metropolis-Hastings algorithms. One method is an independent Metropolis-Hastings and the other one is a type of add-and-delete MCMC.

In addition to the theoretical contributions, we present simulation studies which confirm the theoretical convergence results on our methods by monitoring the convergence of our Markov chains to the equilibrium distribution.

Keywords: graph sampling, markov chain convergence, markov chain monte carlo, metropolis-hastings, sampling method.

1. Definitions

A graph is a heavily used data structure in the world of algorithms and there are numerous applications of it in computer science like networks of communication, data organization, computational devices and the flow of computation. Graphs have proven particularly useful in linguistics in addition to the use of them in chemistry, physics, sociology and biology.
Let $G(V, M)$ be a graph consists of V as its finite set of vertices and M is a finite set of ordered pairs of the form (u, v) called edges. The graph is directed if the pair (u, v) is not the same as (v, u), where (u, v) indicates that there is an edge from vertex u to vertex v, while it is undirected if both pairs represent the same edge. Also, a graph is called regular if each vertex has the same number of neighbours; i.e. every vertex has the same degree. A special case of regular graphs is called complete graphs where any vertex in the graph is connected with all other vertices. A graph is connected if for every pair (x, y) of distinct vertices there is a path from x to y. A graph without any cycle is a forest; a tree $T(V_T, M_T)$ is a connected forest where the order of a tree is its number of vertices $|V_T|$ and the tree size is its number of edges $|M_T|$. For a rigorous presentation of graph properties we recommend (Bollobás 1979) and (Bollobás 2001). Here we consider the uniform distribution over trees $T(V_T, M_T)$ verifying $\mathrm{card}(\mathrm{V_T}) = k + 1$.

2. Literature review

Many applications are generating very huge graphs with thousands and millions of vertices. These large graphs are challenging to study because they need to run expensive algorithms such as simulations in addition to that it is hard to get an impression of the graph topology from visualization. Unfortunately, such activities on large graphs usually cost a lot of time that is computed at least polynomially in the number of vertices. One way to reduce the runtime is to reduce the graph size by sampling a structure from the large graph which approximates the original graph well.

Graph mining is a branch of data mining which is used for mining graph structures (Rehman, Khan, and Fong 2012). It has gained much attention during the last years and finds its applications in many domains like: social and computer networks, bioinformatics and chemistry. In the literature, various approaches for graph mining have been proposed for classification, clustering and sampling. In general, the graph sampling methods, which are our concern, are divided into three categories (Ahmed, Neville, and Kompella 2011): node sampling, edge sampling and topology-based sampling.

Node sampling algorithm simply creates a representative structure by sampling the vertices uniformly where the edges between the sampled vertices in the large graph are considered as edges also in the sampled structure. A known related method is called random node-edge sampling (Hu and Lau 2013) where vertices are uniformly sampled and edges that are incident to these vertices are also uniformly sampled in the sample graph. Additionally, some node sampling methods also consider the neighbors of the sampled vertices like the random node-neighboursampling method (Leskovec and Faloutsos 2006) where all the edges that are linked to the sampled vertices in the graph are sampled into the required structure. Moreover, many of those methods were developed to use the graph topology information by integrating with topology-based sampling methods like the random walk sampling (Yoon, Lee, Yook, and Kim 2007) and the Metropolis algorithm (Hübler, Kriegel, Borgwardt, and Ghahramani 2008), which replaces some sampled vertices with other vertices, sample structure with properties consistent with the graph.

Similarly, edge sampling builds a subgraph by sampling edges randomly. For instance, in random edge sampling (Ebbes, Huang, Rangaswamy, and Thadakamalla 2008), the subgraph is built from edges sampled randomly and uniformly. Another modified edge sampling method is the induced edge sampling (Ahmed, Neville, and Kompella 2012) with both of its extensions, the totally induced edge sampling and the partially induced edge sampling. The first one applies the random edge sampling and obtains adjacent vertices from these edges, then all edges attached to those vertices are chosen to the sampled graph. In contrast, partially induced edge sampling applies the edge sampling where edges are sampled according to a probability.

Various approaches that leverage tree mining algorithms were developed as well since trees are one of the most well studied probabilistic structures in graphs. Over the past decade, trees have found a surprising number of applications in Internet and computer science like, for example, XML which is a markup language designed to store and transport data. For instance, XML data is very popular because of the nature of its tree structure and as a result it is necessary to develop methods that can treat and extract patterns from this type of data like, for example, tracking down the common trees existing among a set of such data. Additionally, it has been heavily researched in biology and bioinformatic where trees are widely used to represent various biological structures like glycans, RNAs, and phylogenies. For example, Shapiro and Zhang (1990) studied the function of the RNA where the RNA structures were collected in trees in order to compare any newly sequenced RNA to compare the similarities in the topological patterns. Takigawa, Hashimoto, Shiga, Kanehisa, and Mamitsuka (2010) represented glycans as directed trees in which nodes are monosaccharides and edges are linkages and proposed an efficient method for mining frequent and statistically significant subtrees from glycan trees. For a good survey of trees mining algorithms in biology, see (Parthasarathy, Tatikonda, and Ucar 2010)

The authors in a previous work (Hancock, Wicker, Takigawa, and Mamitsuka 2012) sampled pathways of genes within significantly coordinated expression profiles. They aimed at identifing the important metabolites which are driving the function of the network by comparing these metabolites to the metabolites in sampled pathways. As a consequence, they have been motivated to search for another bigger structure that can better identify these metabolites.
Although there are many known methods in literature to sample from a graph, most are designed either to sample subgraphs that are representative to the original graph according to the graph properties as in the work of (Hübler *et al.* 2008) or to sample frequent structure patterns as in (Yan and Han 2002) which are not our concern in this work, in addition to the fact that using most of these techniques generally is not efficient to sample subtrees specifically because these subtrees will represent a small proportion of the sampled stuctures.
Our contribution in this work is to present two techniques to sample trees according to a distribution from a graph where the vertices are labelled, i.e the tree structure, a.k.a pattern, is not taken into consideration in our method as well as the graph properties.

3. Sampling methods

Sampling uniformly subtrees of a given size that are obtained from an initial graph is not a trivial problem. A way to solve this problem is to use Markov chain Monte Carlo (MCMC) sampling (For general references on Markov chains and Markov chain Monte Carlo see (Kemeny and Snell 1983), (Robert and Casella 2009) and (Brooks, Gelman, Jones, and Meng 2011)). Here, for this purpose, we present a Metropolis-Hastings (MH) dynamics.
The MH algorithm is an iterative procedure that simulates a Markov chain. If the simulated Markov chain is irreducible and aperiodic, as the configuration space of the chain is finite, the algorithm is convergent. More precisely, this means that the outputs of the algorithm are asymptotically distributed according to the unique invariant distribution of the simulated Markov chain (Häggström 2002).
The principle of the MH algorithm is the following: let π be a distribution of interest and consider x^t the current state of the chain. A new candidate x^* is proposed according to a proposal distribution $q(x^t, x^*)$. The proposed candidate is accepted as the new state of the chain, i.e $x^{t+1} = x^*$ with probability :

$$\alpha = \frac{\pi(x^*)}{\pi(x^t)} \frac{q(x^t, x^*)}{q(x^*, x^t)} \wedge 1 = \frac{q(x^t, x^*)}{q(x^*, x^t)} \wedge 1$$

We present two methods for sampling uniformly trees with k edges from an undirected and unweighted graph. The first method samples uniform trees according to an independent Metropolis-Hastings, whereas the second method does it according to a non-independent Metropolis-Hastings algorithm.
Our Markov chain (X^t) is produced through the transition kernel:

$$X^{t+1} = \begin{cases} x^* \text{ with probability } = \alpha \\ x^t \text{ with probability } = 1 - \alpha \end{cases}$$

For the independent method, we use a special case of the Metropolis-Hastings algorithm where the candidate x^* is independent of the present state of the chain x^t so $q(x^*, x^t) = q(x^*)$ and the transition kernel:

$$X^{t+1} = \begin{cases} x^* \text{ with probability } \alpha = \frac{q(x^t)}{q(x^*)} \wedge 1 \\ x^t \text{ with probability } 1 - \alpha \end{cases}$$

In what follows, first each method is detailed and then their convergence speeds are studied.

3.1. First method: independent uniform trees

In the following we present the proposal distribution for the independent sampler. This method generates the candidate tree x^* in the following way. A vertex v_{i_1} is selected ran-

domly from the original graph and receives a weight $w_{i_1} = k$, next this weight w_{i_1} is distributed among all neighbours in the graph so that each vertex receives a weight, then vertices with weight greater than 0 are selected as neighbours for the first vertex in the generated subtree. The weight k represents the number of vertices we can connect for the whole subtree after choosing a given vertex, this weight is distributed on the selected neighbours according to a uniform multinomial distribution with equal probabilities. This process is conducted iteratively until no weight is left anywhere. Algorithm 1 presents the detailed steps to generate a tree T, it is important to note that A is an ordered set where the last entered vertices are the smallest in the ordering sense.

Algorithm 1 Generate a random tree T

uniform selection of v among V
$T \leftarrow v$
$w(v) \leftarrow k + 1$
the ordered set of active vertices $A \leftarrow v$
while $A \neq \emptyset$ **do**
  select the first vertex v in A
  $A \leftarrow A \setminus v$
  let $n(v) = v_{i_1}, \ldots, v_{i_{|n(v)|}}$ be the neighbours of v in the graph not already selected in the tree
  if $n(v) = \emptyset$ and $w(v) > 1$ **then**
    the algorithm will stop and relaunch again from the beginning
  else
    the weight $w(v) - 1$ is distributed among $n(v)$ using a multinomial law $M(w(v) - 1, |n(v)|^{-1}, \ldots, |n(v)|^{-1})$
    for all $v^* \in n(v)$ **do**
      if $w(v^*) > 0$ **then**
        $A \leftarrow A \cup v^*$
        $T \leftarrow T \cup v^*$
      end if
    end for
  end if
end while

To compute $p(T)$, the probability of generating a subtree T, start from any vertex v of T and compute the probability of generating the subtree T starting from it. Algorithm 2 allows to compute the probability of generating a subtree as detailed in its description. $n_T(v)$ denotes the neighbours of v in subtree T. Also, $p(x^*)$ must be computed to take into account all the possible ways of generating x^*.

Algorithm 2 Compute the probability $p(T)$ of generating T

$p(T) \leftarrow 0$
for all $v \in T$ **do**
  $T' \leftarrow T \setminus v$
  compute $w(v)$
  $p \leftarrow 1$ and $A \leftarrow v$
  for all $v \in A$ **do**
    $A \leftarrow A \setminus v$
    $p \leftarrow p \times \frac{(w(v)-1)!}{w(v_i)!\ldots w(v_{|n_T(v)|})!} \frac{1}{|n_T(v)|^{w(v)-1}}$
    $A \leftarrow A \cup \{v_1 \ldots v_{n_T(v)}\}$
  end for
  $p(T) \leftarrow p(T) + p$
end for

The weights used to compute the probability of generating a subtree are computed as shown in algorithm 3.

The chain is clearly irreducible as each subtree has a positive probability of being sampled at each step. This also implies the aperiodicity since it is always possible to stay at the same state.

Bollobás (2001) showed that for $r \geq 2$ and $n \geq 3$, if Y_i is the number of cycles of length at most i in a $r-$regular graph generated by the Erdős-Rényi random graph, then $Y_3, Y_4, ..., Y_n$ are

Algorithm 3 Compute $w(v, T^{'})$

$w(v) \leftarrow 1$
for all $v_i \in n_T(v) \cap T^{'}$ **do**
  $T^{'} \leftarrow T^{'} \setminus v_i$
  compute weight $w(v_i)$
  $w(v) \leftarrow w(v) + w(v_i)$
end for

asymptotically independent Poisson random variables with mean $\lambda_i = (r-1)^i/2i$. We assume henceforth that these random variables follow a Poisson distribution, indeed in practical applications the graph size is often large. Under these assumptions the following result gives a bound on the speed of convergence for the independent Markov chain sampler.

Theorem 1. *Assume that $Y_k + 1$ is a Poisson distributed random variable representing the number of cycles of length at most $k+1$, then for a random $r-$regular graph in Erdős-Rényi random graph model, where the vertex degree $r \geq 2$ and each vertex is contained in a cycle of length at least $k+1$, we have for any starting state x:*

$$\| M_x^m - \pi \|^2 \leq \frac{1}{4\pi(x)} \left(1 - \frac{k+1}{nr^{k(k+1)/2}} \left(\frac{(r-1)^k}{2} \left(\frac{r-1}{k+1} - \frac{1}{k} \right) - 2\sqrt{\frac{(r-1)^{k+1}}{\alpha(k+1)}} \right) \right)^{2m}$$

with a probability larger than $1-\alpha$, where M_x^m is the m^{th} updated distribution and π is the target distribution.

Proof. For the convergence, we can use the bound on the total variation distance:

$$\| M_x^m - \pi \|^2 \leq \frac{1}{4\pi(x)} (1 - u(1))^{2m}$$

with $u(1) = \min(p(x)/\pi(x))$ (Liu 1996). As $\pi(x)$ is the inverse of the number of subtrees, we need a lower bound on the number of subtrees of a given size k in a graph.
The number of subtrees can be bounded considering that a tree is obtained whenever a cycle of length $k+1$ is deprived of an edge, in this way we can obtain $k+1$ different subtrees of size k. Moreover, if we can have c such cycles we are able of generating $(k+1)c$ subtrees, this is the way we follow to lower bound the number of subtrees although it is a fact that the true number of subtrees in a graph is bigger than the number of subtrees generated in our way.
Let Y_k+1 denote the number of cycles of length at most $k+1$. Let us denote by F the following event: $|Y_{k+1} - Y_k - (E(Y_{k+1}) - E(Y_k))| < \epsilon$ then the number of cycles of size exactly $k+1$ is $Y_{k+1} - Y_k$ which verifies:

$$\begin{aligned}
P(F) &\geq P\left(\left| Y_{k+1} - \frac{(r-1)^{k+1}}{2(k+1)} \right| < \epsilon/2 \cap \left| Y_k - \frac{(r-1)^k}{2k} \right| < \epsilon/2 \right) \\
&\Leftrightarrow P\left(F^C\right) \leq P\left(\left| Y_{k+1} - \frac{(r-1)^{k+1}}{2(k+1)} \right| < \epsilon/2 \cap \left| Y_k - \frac{(r-1)^k}{2k} \right| < \epsilon/2 \right)^C \\
&\Leftrightarrow P(F^C) \leq P\left(\left| Y_{k+1} - \frac{(r-1)^{k+1}}{2(k+1)} \right| > \epsilon/2 \right) + P\left(\left| Y_k - \frac{(r-1)^k}{2k} \right| > \epsilon/2 \right) - \\
&P\left(\left| Y_{k+1} - \frac{(r-1)^{k+1}}{2(k+1)} \right| > \epsilon/2 \cap \left| Y_k - \frac{(r-1)^k}{2k} \right| > \epsilon/2 \right) \\
&\Rightarrow P(F^C) \leq \frac{2(r-1)^{k+1}}{(k+1)\epsilon^2} + \frac{2(r-1)^k}{k\epsilon^2} \text{ using Chebyshev inequality} \\
&\Rightarrow P(F^C) \leq \frac{4(r-1)^{k+1}}{(k+1)\epsilon^2} \\
&\Rightarrow P(F) \geq 1 - \frac{4(r-1)^{k+1}}{(k+1)\epsilon^2} \qquad (1)
\end{aligned}$$

$$\text{let } \epsilon = \sqrt{\frac{4(r-1)^{k+1}}{(k+1)\alpha}}, \tag{2}$$

then we can conclude that with probability larger than $1-\alpha$:

$$\begin{aligned}
&\left|Y_{k+1} - Y_k - \left(\frac{(r-1)^{k+1}}{2(k+1)} - \frac{(r-1)^k}{2k}\right)\right| \leq 2\sqrt{\frac{(r-1)^{k+1}}{\alpha(k+1)}} \\
&\Rightarrow \left|Y_{k+1} - Y_k - \frac{(r-1)^k}{2}\left(\frac{r-1}{k+1} - \frac{1}{k}\right)\right| \leq 2\sqrt{\frac{(r-1)^{k+1}}{\alpha(k+1)}} \\
&Y_{k+1} - Y_k \geq \frac{(r-1)^k}{2}\left(\frac{r-1}{k+1} - \frac{1}{k}\right) - 2\sqrt{\frac{(r-1)^{k+1}}{\alpha(k+1)}}
\end{aligned} \tag{3}$$

Let S denote the set of all possible subtrees of size k that can be obtained from our graph. A given tree of size k can be generated for the proposal distribution in $k+1$ different ways when starting from any vertex in the tree. Additionally, each way involves l steps ($l \in 1, \ldots, k$) representing number of multinomial steps, at least 1 if all weights are distributed at once in a star-like manner, and at most k if weights are given every time to a single vertex producing thus a path of length $k+1$. As a consequence, each subtree has a proposal probability of the form:

$$\begin{aligned}
&\frac{1}{n}\prod_{i=1}^{l}\frac{w_i!}{w_{i1}!\ldots w_{ir_i}!}\frac{1}{r_i^{w_i}} \quad \text{where } i \leq r \\
&\geq \frac{1}{n}\prod_{i=1}^{k}\frac{1}{r_i^{w_i}} \text{ with } w_i = w_{i1} + \cdots + w_{ir_i} \\
&\geq \frac{1}{nr^{\sum_i w_i}} \\
&\geq \frac{1}{n}\frac{1}{r^{k(k+1)/2}}
\end{aligned}$$

Thus the probability $p(T)$ of generating a tree is lower bounded by:

$$\frac{1}{n}\frac{k+1}{r^{k(k+1)/2}} \tag{4}$$

Gathering results of equations 3 and 4 we obtain:

$$\begin{aligned}
\parallel M_x^m - \pi \parallel^2 &\leq \frac{1}{4\pi(x)}(1-u(1))^{2m} \\
&\leq \frac{1}{4\pi(x)}\left(1 - \frac{k+1}{nr^{k(k+1)/2}}\left(\frac{(r-1)^k}{2}\left(\frac{r-1}{k+1} - \frac{1}{k}\right) - 2\sqrt{\frac{(r-1)^{k+1}}{\alpha(k+1)}}\right)\right)^{2m}
\end{aligned}$$

□

Clearly, this bound goes to 0 while assuming that the graph is large; i.e the number of vertices $|V| = n$ is large. We assumed that each vertex is contained in a cycle of length $k+1$ to guarantee that the proposed subtree x^* can be obtained from any of its vertices. In general, the success rate s of algorithm 1 will be strictly lower than 1. It is worth to mention that s is not the Metropolis-Hastings acceptance rate to avoid misleading interpretations. More precisely, algorithm 1 failure results from a shortage of the number of required neighbours for the selected vertex in comparison to the given weight of this vertex that will be distributed among neighbours which will cause algorithm 1 to stop and start again from the beginning. The success rate could be estimated as $s \approx$ (number of acceptances)/(number of steps) which is normally close to 1 unless the border is large as on expander graphs for example.

Theorem 2. *Let $Y_k + 1$ be the same random variable defined in Theorem 1 and s represents the algorithm success rate, then for a random $r-$regular graph, where $r \geq 2$, we have for any starting state x:*

$$\| M_x^m - \pi \|^2 \leq \frac{1}{4\pi(x)} \left(1 - \frac{s(k+1)}{nr^{k(k+1)/2}} \left(\frac{(r-1)^k}{2}\left(\frac{r-1}{k+1} - \frac{1}{k}\right) - 2\sqrt{\frac{(r-1)^{k+1}}{\alpha(k+1)}}\right)\right)^{2m}$$

with a probability larger than $1-\alpha$, where M_x^m is the m^{th} updated distribution and π is the target distribution.

Proof. Again, we use the total variation distance to bound the convergence:

$$\| M_x^m - \pi \|^2 \leq \frac{1}{4\pi(x)}(1-u(1))^{2m}$$

with $u(1) = \min(p(x)/\pi(x))$ (Liu 1996). If we ignore the constraint in Theorem 1 stating that each vertex is contained in a cycle of length at least $k+1$ then it is possible at any step during the process that the distributed weight among the chosen vertex is greater than the number of neighbours for this vertex which will cause algorithm 1 to stop and start again from another vertex. The probability of generating a subtree needs to be adjusted, for this, first we compute it conditionally on the success of algorithm 1. It is bounded by

$$\frac{1}{nc}\prod_{i=1}^{k}\frac{w_i!}{w_{i1}!\dots w_{ir}!}\frac{1}{r_i^{w_i}} \geq \frac{1}{n}\prod_{i=1}^{k}\frac{w_i!}{w_{i1}!\dots w_{ir}!}\frac{1}{r_i^{w_i}} \geq \frac{1}{n}\frac{k+1}{r^{k(k+1)/2}}$$

where c is the normalizing constant upper bounded by 1 as the distribution of weights is constrained by the success of algorithm 1. Next, we can use again bound 3 on the number of cycles as well as the success rate s to obtain:

$$\begin{aligned}\| M_x^m - \pi \|^2 &\leq \frac{1}{4\pi(x)}(1-u(1))^{2m} \\ &\leq \frac{1}{4\pi(x)}\left(1 - \frac{s(k+1)}{nr^{k(k+1)/2}}\left(\frac{(r-1)^k}{2}\left(\frac{r-1}{k+1} - \frac{1}{k}\right) - 2\sqrt{\frac{(r-1)^{k+1}}{\alpha(k+1)}}\right)\right)^{2m}\end{aligned}$$

□

The convergence speed for this chain is $\mathcal{O}((s^{-1}nr^{\frac{k^2}{2}}))$.

Corollary 1. *For the random $r-$regular graph in Theorem 1, with a probability larger than $1-\alpha$, the method restricted to path sampling verifies:*

$$\| M_x^m - \pi \|^2 \leq \frac{1}{4\pi(x)}\left(1 - \frac{2}{nr^{k(k+1)/2}}\left(\frac{(r-1)^k}{2}\left(\frac{r-1}{k+1} - \frac{1}{k}\right) - 2\sqrt{\frac{(r-1)^{k+1}}{\alpha(k+1)}}\right)\right)^{2m}$$

Proof. The only thing to do is to replace factor $k+1$ by 2 in the proof of Theorem 1 in equation 4 as there can be only two starting points for a path. □

3.2. Second method: crawling uniform trees

As a first step, we initialize the algorithm by selecting a random vertex and adding k edges greedily to build a tree of order $k+1$. Following, the crawling method modifies the initial subtree by shifting randomly one of its edges, i.e an edge is deleted randomly and another is randomly added. The Metropolis-Hastings acceptance rate is:

$$\frac{q(x^t, x^*)}{q(x^*, x^t)} \wedge 1 \text{ with } q(x^t, x^*) = \begin{cases} \frac{1}{2}\frac{1}{|n_{G'}(x^t)|} & \text{if } x^* \neq x^t \\ \frac{1}{2} & \text{if } x^* = x^t \end{cases}$$

where $n_{G'}(x^t)$ denotes the set of neighbours for the tree x^t in the graph $G^{'}(V^{'}, E^{'})$ representing the Markov chain state space. Let $n_{x^t}(v_i)$ denotes the set of neighbour vertices inside the tree x^t for all vertices belonging to the tree x^t, l_{x^t} represents the set of leaf vertices in the tree which are vertices connected to only one neighbour vertex in the tree x^t and finally let $n_G(v_i)$ the set of neighbours for the tree leaves inside the graph $G(V, E)$ which don't belong to the tree x^t. Algorithm 4 presents the steps to generate a tree x^* from a randomly initialized tree x^t.

Algorithm 4 Generate a a tree x^{t+1}

Require: A tree x^t where $V_T = \{v_1, v_2, \ldots, v_{k+1}\}$ and $E_T = \{e_1, e_2, \ldots, e_k\}$
  Initialize $l_{x^t} = \{v_i \in x^t \text{ s.t } |n_{x^t}(v_i)| = 1\}$
  Initialize $n_G(x^t) = \{v_j \in V \backslash V_T \; s.t. \; e = (v_i, v_j) \in E \text{ where } v_i \in x^t \;\&\; v_j \notin x^t\}$
  while not mixed **do**
    Sample v_i in l_{x^t}
    $x^t \leftarrow x^t \backslash \{v_i\}$
    Update $n_G(x^t)$
    Sample v_j in $n_G(x^t)$
    $x^{t+1} \leftarrow x^t \cup \{v_j\}$ with probability $\alpha \leftarrow min\left\{1, \frac{q(x^t, x^{t+1})}{q(x^{t+1}, x^t)}\right\}$
    $x^{t+1} \leftarrow x^t \cup \{v_i\}$ with probability $1 - \alpha$
  end while

This chain is aperiodic since $q(x^t, x^*) > 0$ for all states x^* which allow it to stay at the same state with a positive probability. Also, it is irreducible as will be seen along the proof of theorem 3.

In order to bound the mixing time of a Markov chain, we will use the second smallest eigenvalue λ_1 by making the Markov chain lazy where a lazy chain stays in the current state at each step with probability at least 1/2. Factor 1/2 is a sufficient condition for the transition probability matrices to have only positive eigenvalues.

First, let us recall that the Cheeger constant for graph $G(V, E)$ is defined as

$$h = \min_S \frac{|E(S, \bar{S})|}{\min\{\text{vol}(S), \text{vol}(\bar{S})\}}$$

where $\text{vol}(S)$ stands for the sum of degrees of vertices in S and $|E(S, \bar{S})|$ for the number of edges between S and $\bar{S}$. Henceforth S will denote the subset of V realizing this minimum and without loss of generality we consider that $S = \min\{S, \bar{S}\}$ so that

$$h = \frac{|E(S, \bar{S})|}{\text{vol}(S)}.$$

Lemma 1. *Let h represents the Cheeger constant for the graph $G(V, E)$ and $vol(V)$ denotes the sum of degrees of vertices in V, then there exists a set of paths Γ where b is the maximum number of paths containing an edge e such that $b \leq \lfloor \frac{\text{vol}(V)}{h} \rfloor + 1$.*

Proof. This is proven by contradiction. First let us recall that a path is a sequence of edges which connect a sequence of distinct vertices. For convenience, c will stand for $\lfloor \text{vol}(V)/h \rfloor$. Let us suppose that whichever the set of paths Γ there is always an edge e with at least $c+2$ paths containing it. Then, e is an edge between two vertices a and b. Let us suppose that there is a path p between a and b with edges supporting less than $c+1$ paths, then edge e could have less than $c+2$ paths crossing it by replacing a path $x \to a \to b \to y$ which crosses e by the new path $x \to a \to p \to b \to y$. So in any path between a and b one will meet at some point an edge supporting at least $c+1$ paths. The consequence of it, is that there is a graph cut between a and b containing only edges with at least $c+1$ paths. This graph cut creates two subgraphs, A and B containing respectively a and b. We show now that this leads

to a contradiction. Indeed, by definition of S

$$\begin{aligned}
&\frac{|E(S,\bar{S})|}{\text{vol}(S)} \leq \frac{|E(A,B)|}{\text{vol}(A)} \text{ if, without loss of generality, } \text{vol}(A) \leq \text{vol}(B) \\
&\Rightarrow \frac{\text{vol}(A)\text{vol}(B)}{|E(A,B)|} \leq \frac{\text{vol}(S)\text{vol}(B)}{|E(S,\bar{S})|} \\
&\Rightarrow \frac{|A|.|B|}{|E(A,B)|} \leq \left\lfloor \frac{\text{vol}(V)}{h} \right\rfloor + 1 \qquad (5)
\end{aligned}$$

By hypothesis, each edge in $|E(A,B)|$ belongs to at least $c+1$ paths and in particular e supports $c+2$ paths. It is then immediate, that the mean number of paths per edge belonging to $|E(A,B)|$ is strictly greater than $c+1$ which is impossible when observing the last equation. This concludes the proof. □

Theorem 3. *Let a be the maximum number of subtrees associated to a vertex , d_{max} is the graph maximum degree, D is the graph diameter and λ_1 is the second smallest eigenvalue on the graph then for any starting state x such that M_x^m is the m^{th} updated distribution and π is the target distribution we have:*

$$\| M_x^m - \pi \|^2 \leq \frac{1}{4\pi(x)} \left(1 - \frac{1}{K}\right)^{2m}$$

with $K \leq 2a^2 d_{max} k^2 (k+1)^2 (D+k) \left(\frac{2d_{max}}{\lambda_1} + 1\right)$.

Proof. Again, let $G^{'}(V^{'}, E^{'})$ be the Markov chain state space or simply the graph of subtrees with k edges. According to Sinclair (1992), the Markov chain has the following bound on its second smallest eigenvalue:

$$\lambda_1 \leq 1 - \frac{1}{K} \text{ with } K = \max_e Q(e)^{-1} \sum_{\gamma_{xy} \ni e} |\gamma_{xy}| \pi(x)\pi(y)$$

where $|\gamma_{xy}|$ stands for the length of the path γ_{xy} and $Q(e) = \pi(e_1)P(e_1,e_2)$ if e is the edge (e_1,e_2).
K can be bounded in the following way

$$\begin{aligned}
K &\leq \max_e \frac{1}{\pi(e_1)P(e_1,e_2)} \sum_{\gamma_{xy} \ni e} \pi(x)\pi(y)D' \text{ where } D' \text{ is the diameter of } G'(V',E') \\
&\leq \pi(x) b' D' \max_e \frac{1}{P(e_1,e_2)} \qquad (6)
\end{aligned}$$

where $P(e_1,e_2)$ is the transition matrix and b' stands for the maximum number of paths crossing an edge e in the graph $G'(V',E')$.
At this point, we have to make $P(e_1,e_2)$ more explicit, indeed $Q(e_1,e_2)$ is the Markov chain related to the Metropolis-Hastings algorithm being used, that is:

$$\begin{aligned}
P(e_1,e_2) &= \left(\frac{q(e_2,e_1)}{q(e_1,e_2)} \wedge 1\right) q(e_1,e_2) \text{ where } q(e_1,e_2) \text{ is the proposal law} \\
&= q(e_1,e_2) \wedge q(e_2,e_1) \\
&\geq \frac{1}{2|n_{G'}(x^t)|} \\
&\geq \frac{1}{2k^2 d_{max}} \qquad (7)
\end{aligned}$$

where the probability $\frac{1}{2}$ is due to the laziness. Thus,

$$K \leq 2\pi(x) k^2 d_{max} b' D' \qquad (8)$$

First, let us start by bounding the value of $b^{'}$. Here we need to use conductance information on the graph $G(V,E)$ given by b. The set of paths Γ' on $G^{'}(V^{'},E^{'})$ is derived from the set of paths Γ in $G(V,E)$ in the following way. To each vertex $v \in V$ is associated the set of subtrees containing it and denoted by $\mathcal{T}(v)$. So if we want to have a path between two subtrees s and t, we look for $v_s \in V$ and $v_t \in V$ such that $s \in \mathcal{T}(v_s)$ and $t \in \mathcal{T}(v_t)$. Then, if there is a path $v_s = v_1 \to v_2 \cdots \to v_n = v_t$ between v_s and v_t, there is an associated path between s and t, denoted by $s = t(v_1) \to t(v_2) \to \cdots \to t(v_n)$ where $t(v_i) \in \mathcal{T}(v_i)$. This is possible as $v_i \sim v_{i+1}$, so to obtain $t(v_{i+1})$ connected to $t(v_i)$ it is enough to remove the farthest edge of $t(v_i)$ to $t(v_{i+1})$ and replace it by the edge $v_i v_{i+1}$. By the way, this shows that $D^{'} \leq D + k$. The additional term is needed because it may happen that the first subtree reaching v_t is different from t so that additional moves are needed, to a maximum of k. Now, if we denote by a the maximum number of subtrees associated to a vertex we can bound b' noting that an edge e' of $G'(V',E')$ will be crossed as many times as there are paths in Γ' associated to paths in Γ crossing edges v_i, v_j where v_i is a vertex of s and v_j a vertex of t. That makes $(k+1)^2$ pairs multiplied by the number of times these pairs can be used, so:

$$b^{'} \leq b(k+1)^2 a^2. \tag{9}$$

Then, the bound on b of lemma 1 can be injected and we get

$$b^{'} \leq (k+1)^2 a^2 \left(\frac{\mathrm{vol}(V)}{h} + 1 \right).$$

Besides, by Cheeger inequality $\lambda_1 \leq 2h$ so that

$$b^{'} \leq (k+1)^2 a^2 \left(\frac{2\mathrm{vol}(V)}{\lambda_1} + 1 \right). \tag{10}$$

Consequently, injecting bound $D' \leq D + k$ and equation 10 into 8 we get

$$\begin{aligned} K &\leq 2\pi(x) k^2 d_{max} (k+1)^2 a^2 \left(\frac{2\mathrm{vol}(V)}{\lambda_1} + 1 \right) (D+k) \\ &\leq 2\pi(x) d_{max} k^2 (k+1)^2 a^2 (D+k) \left(\frac{2 d_{max} \pi(x)^{-1}}{\lambda_1} + 1 \right) \\ &\leq 2 d_{max} k^2 (k+1)^2 a^2 (D+k) \left(\frac{2 d_{max}}{\lambda_1} + 1 \right) \end{aligned}$$

□

4. Experimental evaluation

In this section we examine the theoretical results using simulations on three different types of graphs: Erdős-Rényi graph, regular graph and a barbell graph variant. The barbell graph consists of two connected grids, each grid contains a finite number of adjacent connected vertices where the connections between any two adjacent vertices represent an undirected edge. We were careful to generate grids with different structures for the same graph, i.e any vertex in the first grid can be connected to another vertex only in a horizontal or vertical direction so the maximum number of edges touching a vertex is 4 whereas any vertex in the second grid can touch a maximum number of 8 neighbour vertices since it is possible for it to connect also diagonally, this is justified because we want to detect the behaviour of the sampling methods with different structures.
During the simulations, we sampled subtrees of size $5, 10$ and 20 from graphs of size between 1000 and 1 million. To guarantee the efficiency of theses simulations, a burn-in period of size 1000 iterations was applied and we sampled only one sample each 1000 iterations.

4.1. Sampling from Erdős-Rényi graph

Erdős-Rényi graph $G(n, p)$ is constructed by connecting nodes randomly where each edge is included in the graph with probability p independently from every other edge. In our graph, we fixed the number of vertices and edges $|V| = 60000$ and $|E| = 600000$, respectively. This graph is generated with $p = (2|E|)/(|V||V-1|) \approx 0.00033$.
Figures 1 and 2 present the ACF plots of the diameter and the number of leaves for sampled subtrees using both methods and for different subtrees sizes, respectively. For the same number of iterations, the convergence of the crawling method was quicker than the independent method. It is also clear that the mixing time was longer when increasing the subtree size for both methods.

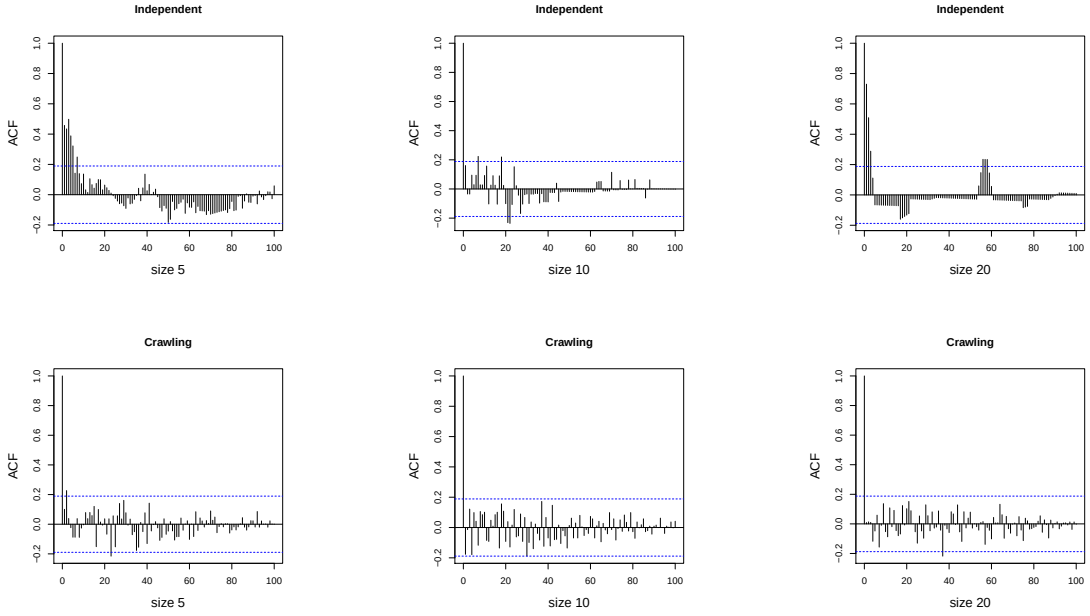

Figure 1: The ACF for the diameter of subtrees sampled using both methods

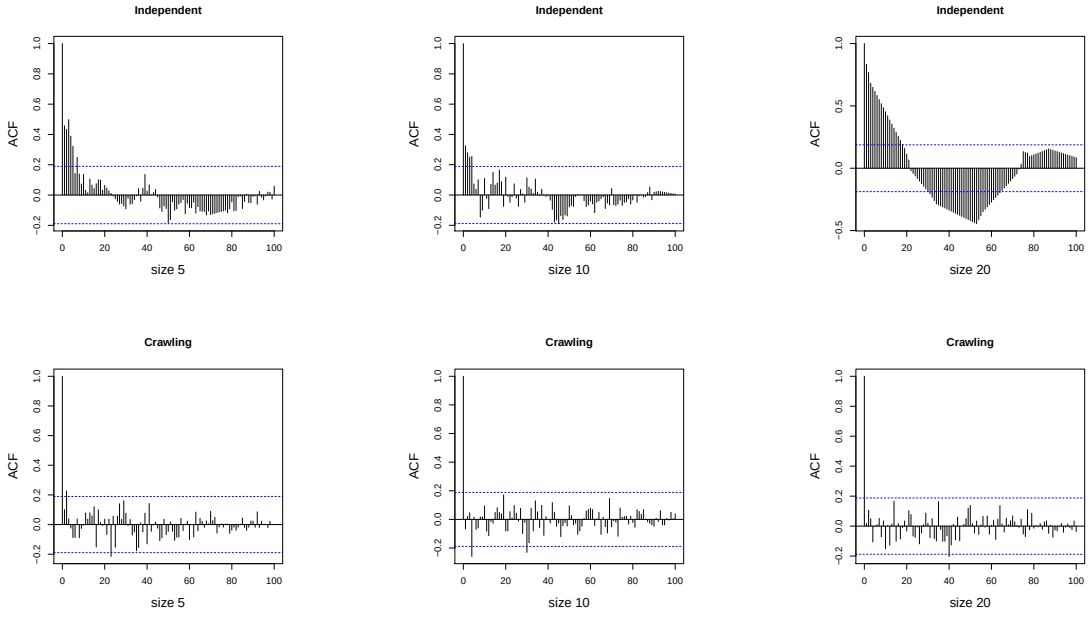

Figure 2: The ACF for the number of leaves of subtrees sampled using both methods

4.2. Sampling from $r-$regular graph

Both sampling methods were applied on a special case of graphs where all edges have the same degree r to see the effect of the vertex degree on the convergence speed. In this graph, we fixed the number of vertices $|V| = 100000$ and the degree $r = 40$.
Figures 3 and 4 present the ACF plots for the diameter and the number of leaves. The crawling algorithm again converged quicker in all presented cases. Unlike the crawling method, the mixing speed of the independent algorithm was slower when the subtree size increased which confirms our theoretical result showing the effect of the size on the convergence speed of the independent method.

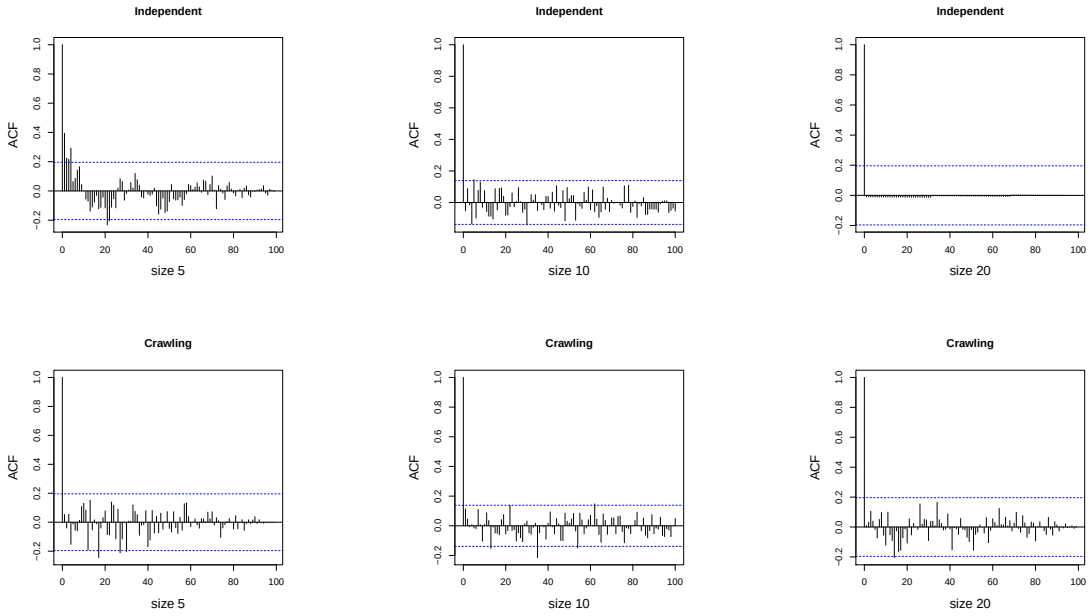

Figure 3: The ACF of the diameter of subtrees sampled using both methods

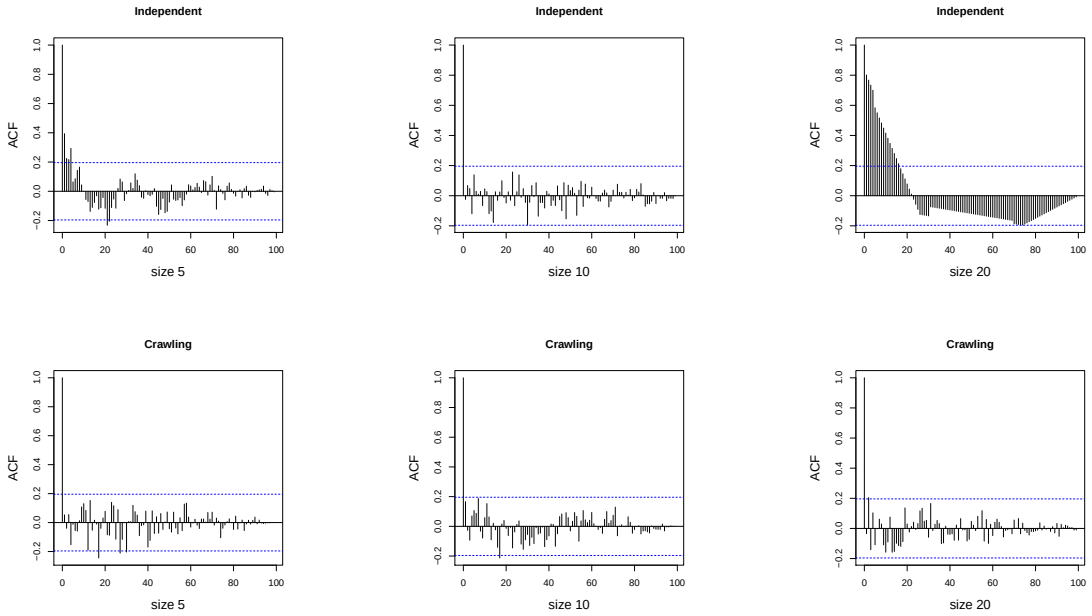

Figure 4: The ACF plot for the number of leaves of subtrees sampled using both methods

Figure 5 presents the ACF for the diameter of subtrees of the same size sampled from graphs with different vertex degrees. When sampling from a $4-$regular graph, it is clear that the chain converged quickly unlike the case when we increased the degree, $r = 40$, then chain took more time to converge. This result is compatible to the one achieved in the theoretical part which proved the effect of the vertex degree r in the $r-$regular graphs on the convergence speed of the independent method and it was assured that the convergence speed is at most $\mathcal{O}(nr^{k^2/2})$.

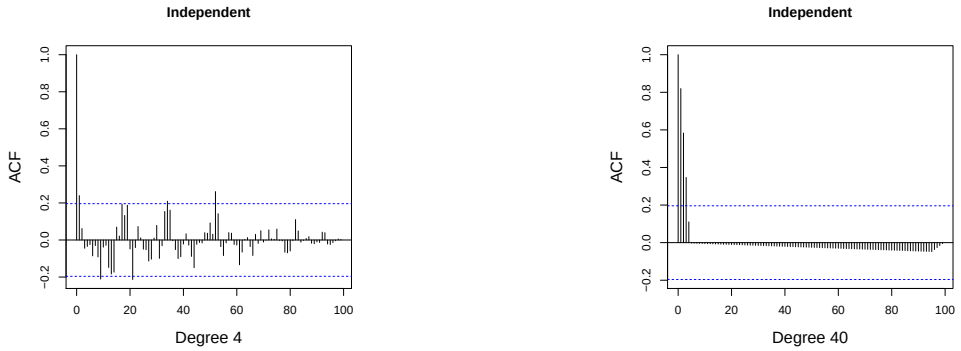

Figure 5: The ACF plot for the diameter of subtrees sampled by the independent methods from two graphs with different vertex degree

A simulation has been performed on a complete graph, which is the worst case for the independent method, in that case only star subgraphs have been produced.

4.3. Sampling from a barbell graph variant

This part was designed to study the effect of the graph's bottleneck on the efficiency of the presented sampling methods. Many cases were considered in this part to monitor the effect of the bottleneck on the mixing time. More precisely, we connected both grids, to construct the graph, by only one edge, 17 edges and then 34 edges and each grid consists of $(174 * 174)$ vertices.
Although the ACF in the plots of the diameter presented in Figure 6 seems to converge when sampling using the crawling method it is not in reality. This result was concluded after reviewing the sampled subtrees, indeed all subtrees were sampled only from the first grid of the graph which means that the chain didn't move to the second grid so the resulted sample does not represent the whole set of graph subtrees of size k.
For the independent method, although it was successful to sample subtrees from this type of graphs it is clear that the chain took more time to converge when increasing the number of edges between both grids. The reason of this behaviour is that the structure of the graph is more complicated for the independent method and to reach convergence the chain needs to run for more iterations.

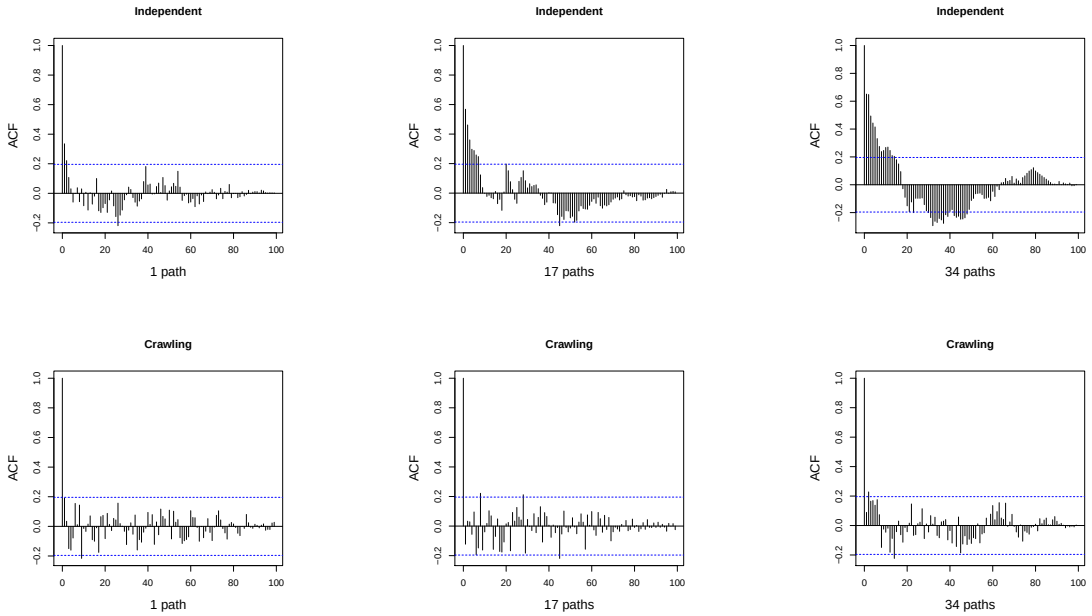

Figure 6: The ACF for the diameter of subtrees sampled using both methods from a graph consists of two different grids connected by only one edge, 17 edges and 34 edges respectively

From Figures 7 and 8 down below we can see the effect of the subtree size on the convergence speed of the independent method. On the other side, we didn't consider the plots of the ACF when sampling using the crawling method because of the same reason clarified above which is related to the bottleneck.
The crawling algorithm was more successful in sampling subtrees of different sizes from all type of graphs except from the grid graph, the sampled subtrees originated from the first grid and it was hard for the chain to move to the second grid. Moreover, we see that both methods took more mixing time when we increased the subtree size.

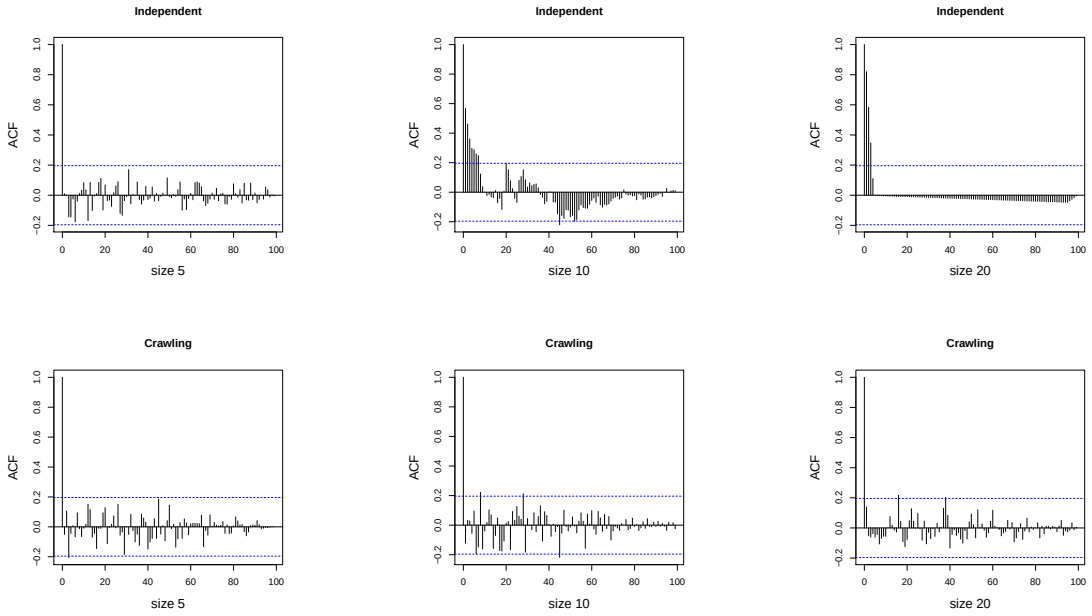

Figure 7: The ACF plot for the diameter of subtrees sampled using both methods

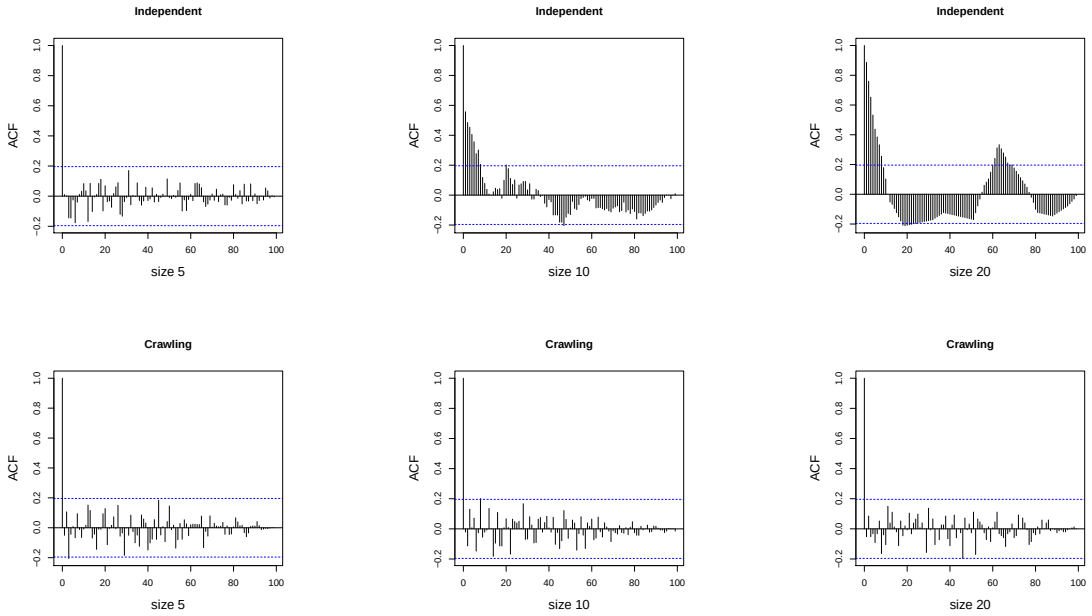

Figure 8: The ACF plot for the number of leaves of subtrees sampled using both methods

5. Conclusion

We presented two Markov chains where the convergence speed for the independent chain is $\mathcal{O}(s^{-1}nr^{k^2/2})$ whereas it is $\mathcal{O}(\lambda_1^{-1}a^2d_{max}^2k^5)$ for the non-independent method which could be very large when the bottleneck is narrow.
Considering the bound on the total variation distance in the crawling method different parameters appear, λ_1 which is the second smallest eigenvalue that can be computed on the graph $G(V, E)$ and a the maximum number of subtrees associated to a vertex. Parameter a is more difficult (impossible often in practice) to compute. Indeed, for a $r-$regular graph it is obvious that for a tree of size k, $a \leq r^k$ which can be large. So, for a general random graph where a is known to be not too large then the crawling method could be better.
According to our simulations, the theoretical results were confirmed. The independent method showed better performance only when sampling subtrees from a graph with a narrow bottleneck whereas the crawling method was the best in the case of sampling from other types of graphs. As a result, we recommend to use a combination of both methods for sampling uniform subtrees, i.e to sample trees in two steps. Firstly one should sample subtrees globally from the graph using the independent method and after that to sample locally through the crawling method.

Acknowledgements We would like to show our gratitude to Prof. R. Stoica (Université de Lorraine) for his comments and discussions during the course of this research and his suggestion of investigating the crawling method. The first author also would like to thank Dr. M. Hindawi (INSA Rennes) for fruitful discussions and valuable suggestions which helped in solving problems in the programming part.

References

Ahmed N, Neville J, Kompella R (2011). "Network Sampling via Edge-Based Node Selection with Graph Induction." *Department of Computer Science Technical Reports*. URL https://docs.lib.purdue.edu/cstech/1747.

Ahmed NK, Neville J, Kompella R (2012). "Network Sampling: From Static to Streaming Graphs." *arXiv:1211.3412 [physics, stat]*. ArXiv: 1211.3412, URL http://arxiv.org/abs/1211.3412.

Bollobás B (1979). *Graph Theory: An Introductory Course*. Graduate Texts in Mathematics. Springer-Verlag, New York. ISBN 978-1-4612-9969-1. URL https://www.springer.com/fr/book/9781461299691.

Bollobás B (2001). *Random Graphs*. 2 edition edition. Cambridge University Press, Cambridge ; New York. ISBN 978-0-521-80920-7.

Brooks S, Gelman A, Jones G, Meng XL (2011). *Handbook of Markov Chain Monte Carlo*. CRC Press. ISBN 978-1-4200-7942-5. Google-Books-ID: qfRsAIKZ4rIC.

Ebbes P, Huang Z, Rangaswamy A, Thadakamalla HP (2008). "Sampling Large-scale Social Networks: Insights from Simulated Networks." *2008 Workshop on Information Technologies and Systems, WITS 2008*, pp. 49–54. URL https://pennstate.pure.elsevier.com/en/publications/sampling-large-scale-social-networks-insights-from-simulated-netw.

Häggström O (2002). *Finite Markov Chains and Algorithmic Applications*. 1 edition edition. Cambridge University Press, Cambridge ; New York. ISBN 978-0-521-89001-4.

Hancock T, Wicker N, Takigawa I, Mamitsuka H (2012). "Identifying Neighborhoods of Coordinated Gene Expression and Metabolite Profiles." *PloS One*, **7**(2), e31345. ISSN 1932-6203. doi:10.1371/journal.pone.0031345.

Hu P, Lau WC (2013). "A Survey and Taxonomy of Graph Sampling." *arXiv:1308.5865 [cs, math, stat]*. ArXiv: 1308.5865, URL http://arxiv.org/abs/1308.5865.

Hübler C, Kriegel HP, Borgwardt K, Ghahramani Z (2008). "Metropolis Algorithms for Representative Subgraph Sampling." In *2008 Eighth IEEE International Conference on Data Mining*, pp. 283–292. IEEE, Pisa, Italy. ISBN 978-0-7695-3502-9. doi:10.1109/ICDM.2008.124. URL http://ieeexplore.ieee.org/document/4781123/.

Kemeny JG, Snell JL (1983). *Finite Markov Chains: With a New Appendix "Generalization of a Fundamental Matrix"*. 1st ed. 1960. 3rd printing 1983 edition. Springer, New York. ISBN 978-0-387-90192-3.

Leskovec J, Faloutsos C (2006). "Sampling from Large Graphs." In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '06*, p. 631. ACM Press, Philadelphia, PA, USA. ISBN 978-1-59593-339-3. doi:10.1145/1150402.1150479. URL http://portal.acm.org/citation.cfm?doid=1150402.1150479.

Liu JS (1996). "Metropolized Independent Sampling with Comparisons to Rejection Sampling and Importance Sampling." *Statistics and Computing*, **6**(2), 113–119. ISSN 1573-1375. doi:10.1007/BF00162521. URL https://doi.org/10.1007/BF00162521.

Parthasarathy S, Tatikonda S, Ucar D (2010). "A Survey of Graph Mining Techniques for Biological Datasets." In *Managing and Mining Graph Data*. doi:10.1007/978-1-4419-6045-0_18.

Rehman SU, Khan AU, Fong S (2012). "Graph Mining: A Survey of Graph Mining Techniques." In *Seventh International Conference on Digital Information Management (ICDIM 2012)*, pp. 88–92. doi:10.1109/ICDIM.2012.6360146.

Robert CP, Casella G (2009). *Introducing Monte Carlo Methods with R*. 2010 edition edition. Springer Verlag, New York. ISBN 978-1-4419-1575-7.

Shapiro BA, Zhang K (1990). "Comparing Multiple RNA Secondary Structures Using Tree Comparisons." *Bioinformatics*, **6**(4), 309–318. ISSN 1367-4803. doi:10.1093/bioinformatics/6.4.309. URL https://academic.oup.com/bioinformatics/article/6/4/309/356690.

Sinclair A (1992). "Improved Bounds for Mixing Rates of Markov Chains and Multicommodity Flow." *Combinatorics, Probability and Computing*, **1**(4), 351–370. ISSN 1469-2163, 0963-5483. doi:10.1017/S0963548300000390. URL https://www.cambridge.org/core/journals/combinatorics-probability-and-computing/article/improved-bounds-for-mixing-rates-of-markov-chains-and-multicommodity-flow/0AD89F9FA25567A5CBBF672C371FBC2B.

Takigawa I, Hashimoto K, Shiga M, Kanehisa M, Mamitsuka H (2010). "Mining Patterns from Glycan Structures." In *Proceedings of the International Beilstein Symposium on Glyco-Bioinformatics*, pp. 13–24. Beilstein Institute.

Yan X, Han J (2002). "gSpan: graph-based substructure pattern mining." In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pp. 721–724. doi:10.1109/ICDM.2002.1184038.

Yoon S, Lee S, Yook SH, Kim Y (2007). "Statistical properties of sampled networks by random walks." *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics*, **75**(4 Pt 2), 046114. ISSN 1539-3755. doi:10.1103/PhysRevE.75.046114.

Affiliation:

Abdelrahman Eid
Le laboratoire de mathématiques Paul Painlevé
Université de Lille
E-mail: abed.eid@najah.edu
URL: http://math.univ-lille1.fr/~abdeid/

Hiroshi Mamitsuka
Bioinformatics Center
Kyoto University
E-mail: mami@kuicr.kyoto-u.ac.jp
URL: http://www.bic.kyoto-u.ac.jp/pathway/mami/

Nicolas Wicker
Le laboratoire de mathématiques Paul Painlevé
Université de Lille
E-mail: nicolas.wicker@univ-lille.fr
URL: http://math.univ-lille1.fr/~wicker/