Analysis of Random Walks using Tabu Lists*

Karine Altisen, Stéphane Devismes, Antoine Gerbaud, and Pascal Lafourcade

Grenoble Université, VERIMAG, France

Abstract. A tabu random walk on a graph is a partially self-avoiding random walk which uses a bounded memory to avoid cycles. This memory is called a tabu list and contains vertices already visited by the walker. The size of the tabu list being bounded, the way vertices are inserted and removed from the list, called here an update rule, has an important impact on the performance of the walk, namely the mean hitting time between two given vertices.
We define a large class of tabu random walks, characterized by their update rules. We enunciate a necessary and sufficient condition on these update rules that ensures the finiteness of the mean hitting time of their associated walk on every finite and connected graph. According to the memory allocated to the tabu list, we characterize the update rules which yield smallest mean hitting times on a large class of graphs. Finally, we compare the performances of three collections of classical update rules according to the size of their associated tabu list.

1 Introduction

A *random walk* is a mathematical formalization of a route taken by a walker through a topology of locations: at each step, the next destination is randomly chosen. Random walks are commonly used to model phenomena from many fields like physics or economics. Random walks are inherently distributed algorithms which do not need any knowledge except the list of next possible destinations. Random walks are probabilistic algorithms and this is one of their main advantages. Indeed, since they are non-deterministic (and thus non-predictable), they can be used to build *resilient* algorithms in a fault prone environment or facing intruders [1, 2]. Even if there is a change in the environment, *e.g.*, in the topology, or a problem due to an intruder or a failure, then a resilient algorithm still ensures a certain quality of service.

The main drawback of random walks is the large number of steps generally needed to reach one vertex starting from another one, namely the *hitting time*. This is mainly due to the fact that the walker may come back to previously visited vertices, forming loops. We study *partially self-avoiding* random walks on *finite* graphs. They are variants of the *simple random walk*, for which at each step the walker chooses its next destination uniformly at random among all its neighbors.

We add a bounded memory to the walker in order to reduce the number of loops. This memory is called a *tabu list* and contains a part of already visited vertices. We say that a tabu list is of length m, if the list can contain at most m elements. The walker avoids every vertex contained in its tabu list unless it has no choice. One can expect that the tabu list helps to reduce the mean hitting time of the walk.

As the size of the tabu list is bounded, when the list is full, one element has to be removed before any new insertion. We call *update rule* the algorithm which drives

* This work has been partially supported by ANR Project Aresa2 and MSTIC Project Terra.

the policy of insertion and removal in the list. For example, in the $FIFO_m$ update rule, the tabu list is of length m, the current position of the walker is always inserted, and when the list is full, the oldest element is firstly removed to make room for the current position. In particular, $FIFO_1$ yields a *non-backtracking* random walk: the walker never backtracks to the last visited vertex unless it is the only neighbor of the vertex currently visited.

Contribution. A tabu random walk is characterized by its update rule. We define a large class of update rules and analytically study their associated tabu random walks. We compare all these tabu random walks *w.r.t.* the mean hitting time between every two given vertices. Mainly, we try to figure out how to handle the memory of the tabu list and what is the best way to do so. The panel of answers we propose here allows to help the choice of an update rule. More precisely, our contribution is threefold:

1. We provide a necessary and sufficient condition on our update rules that ensures the finiteness of the mean hitting time on every graph.
2. We partially answer to the question "What is the best update rule?" by exhibiting a large collection of graphs indexed by a positive integer m, called m-*free* graphs, in which $FIFO_m$ is the optimal (*w.r.t.* the mean hitting time) update rule, provided that the length of the tabu list is at most m. In particular, the 1-*free* class contains all graphs. Therefore, $FIFO_1$ is the optimal policy on every graph if the tabu list contains at most one element.
3. We compare the performances of three collections of classical update rules, *i.e.*, $FIFO_m$, $RAND_m$, and LRU_m, according to the length m of the tabu list. Our results show that no general answer can be given. For some classes of topologies, the mean hitting time decreases when the size of the memory increases. But, counter-intuitively, there exist cases where having more memory is a penalty: We exhibit topologies where the mean hitting time increases when the length of the tabu list increases.

   A important (perhaps surprising) consequence of our results is that for every update rule $FIFO_m$ with $m \geqslant 2$, there exists a graph and two vertices x, y such that the mean hitting time from x to y using $FIFO_m$ is strictly greater than that of the simple random walk. By contrast, $FIFO_1$ always yields smaller mean hitting times than the simple random walk.

Related Work. For a general account on *simple random walks*, we refer to the survey [3] and the forthcoming book [4].

Random walks with memories have received much less attention. Most analytical results deal with infinite graphs, which are irrelevant for our purposes. For example, there are results on *self-avoiding random walks* for infinite graphs, see the survey [5].

On finite graphs, [6] and [7] study random walks that attach memories on the vertices of the graph. Nevertheless, no theoretical analysis of these solutions are yet available.

In [8], the authors study *non-backtracking random walk* on finite and infinite connected graphs with minimum degree two. In particular, they show that for each finite graph, except cycles, $FIFO_1$ is irreducible. Consequently, the mean hitting time of $FIFO_1$ on these graphs is also finite. Our first result is more general than this latter assertion because it deals with all finite connected graphs and a class of update rules that includes $FIFO_1$.

Outline. The description of a random walk equipped with a tabu list is given in Section 2. Section 3, 4 and 5 describe the three main contributions of this paper. Section 6 gives concluding remarks and perspectives.

Due to the lack of space, definitions are intuitively described and only sketches of proof are provided. All formal definitions and proofs are given in the technical report [9].

2 Tabu Random Walks and Update Rules

In our framework, the walker evolves on a simple, undirected and connected graph. Besides, we assume that the vertex set is *finite* and contains at least two vertices.

2.1 Tabu Random Walks

A *tabu random walk* on a simple graph is a partially self-avoiding random walk, where the walker is endowed with a finite memory and can jump from a node to another, provided that they are neighbors. The memory of the walker, called *tabu list*, contains a part of the vertices already visited by the walker. At step n, the position of the walker is represented by the random variable X_n and the current tabu list by the random variable T_n. We will denote by T_n^i the i-th element of T_n. The successive ordered pairs $(X_n, T_n)_{n\geqslant 0}$ is a Markov chain, called *tabu chain*. The tabu random walk is the sequence $(X_n)_{n\geqslant 0}$ of the successive positions of the walker.

At each step, the walker avoids to revisit vertices which are present in the current tabu list, unless he is forced to. More precisely, for every non-negative integer n, the next visited vertex X_{n+1} is uniform on the set formed by the neighbors of the current vertex X_n which are not in the tabu list T_n. If this is not possible because all neighbors of X_n are already in the tabu list T_n, then the next visited vertex X_{n+1} is uniform on the neighborhood of the current vertex X_n. Afterward, the next tabu list T_{n+1} is obtained using an *update rule*.

2.2 Update Rules

The policy to insert or remove occurrences of vertices in the tabu list is called the *update rule* and denoted by R_m, where m is a parameter that gives the maximum number of elements m in the tabu list, that is its length. By extension, m also called the *length of the update rule*. When the dependence on the update rule R_m needs to be emphasized, we will denote $(X_n(R_m), T_n(R_m))_{n\geqslant 0}$ the associated tabu chain.

Every update rule works as follows:

(1) First, concatenate the current vertex X_n and the current tabu list T_n, the concatenation being noted $X_n \cdot T_n$.
(2) Then, possibly remove an element of $X_n \cdot T_n$.

Note that according to (2), for some policies, the first element of the concatenation $X_n \cdot T_n$ might be directly discarded, which means that the current vertex X_n is actually not inserted in the tabu list, *i.e.*, $T_n = T_{n+1}$.

The formal definitions of a tabu random walk and of an update rule are given in [9]. Below, we describe the construction of the tabu list according to a given update rule R_m. A tabu list is said to be *full* if its length is m. At step n, *if the current tabu list T_n is not full* then all elements of T_n distinct from the current vertex X_n are kept, and one of the three disjoint cases occurs:

1. X_n is not inserted and all its occurrences in T_n are kept: $T_{n+1} = T_n$.
2. X_n is inserted and all its occurrences in T_n are kept: $T_{n+1} = X_n \cdot T_n$.
3. X_n is inserted and one of its occurrences in T_n is removed: there exists a random integer C_{n+1} in $\{i \in \{2, \dots, |T_n| + 1\} \,:\, T_n^{i-1} = X_n\}$ such that $T_{n+1} = X_n \cdot T_n^1 \cdots T_n^{C_{n+1}-2} \cdot T_n^{C_{n+1}} \cdots T_n^{|T_n|}$. Besides, the law of the random variable C_{n+1}, conditionally on (X_n, T_n), is fixed by the update rule.

If the tabu list T_n is full, then either case 1 occurs or one element is removed from T_n before X_n is inserted. Formally, there exists a random integer C_{n+1} in $\{1, \dots, m+1\}$ such that

$$T_{n+1} = \begin{cases} T_n & \text{if } C_{n+1} = 1\,, \\ X_n \cdot T_n^1 \cdots T_n^{C_{n+1}-2} \cdot T_n^{C_{n+1}} \cdots T_n^m & \text{if } C_{n+1} \in \{2, \dots, m+1\}\,. \end{cases}$$

Similarly, the law of the random variable C_{n+1}, conditionally on (X_n, T_n), is fixed by the update rule. Note that when the tabu list is full, case 2 is forbidden in order to ensure that the length of T_{n+1} still remains less than or equal to m: X_n cannot be inserted in T_n without removing any element of T_n.

We highlight the fact that the law of the new tabu list only depends on the occurrences of the current vertex in the current tabu list: given these occurrences, the labels of the vertices does not matter. In other words, we exclude from our study the update rules that explicitly use the value of the labels, *e.g.*, a rule which has a special case for a vertex with label "1".

An update rule is *trivial* if the current vertex is never inserted in the tabu list when the tabu list contains no element. For example, the unique update rule with zero length is trivial. For every trivial update rule, if the tabu list is initially empty, then it remains empty forever and the walker performs a *simple random walk*: at each step, the next visited vertex is chosen uniformly at random among the neighbors of the current vertex.

2.3 Examples of Update Rules

For every non-negative integer m, we describe three update rules $FIFO_m$, LRU_m and $RAND_m$ of length m by giving for every non-negative integer n, the law of the next tabu list T_{n+1} conditionally on (X_n, T_n):[1]

$FIFO_m$: The current vertex X_n is inserted at the beginning (left) of the current tabu list T_n. If T_n was already full, then its rightmost element is firstly removed.

LRU_m: All occurrences of the current vertex X_n in T_n are removed. If T_n is still full afterward, then its rightmost element is removed. Then, X_n is inserted at the beginning (left) of T_n.

$RAND_m$: If the current vertex X_n has an occurrence in T_n, then $T_{n+1} = T_n$. Otherwise, if T_n is full, then T_{n+1} is formed by removing one of the $m+1$ elements of $X_n \cdot T_n$ uniformly at random. If T_n is not full, then $T_{n+1} = X_n \cdot T_n$.

Remark that $FIFO_0$, LRU_0 and $RAND_0$ denote the unique trivial update rule with length 0. Note also that when m equals 1 or 2, the update rules $FIFO_m$ and LRU_m coincide and are both distinct from $RAND_m$. However, for every integer $m \geqslant 3$, $FIFO_m$, LRU_m and $RAND_m$ are distinct.

[1] These rules match the requirements given in Subsection 2.2, see [9] for their formal definition.

In general, a random walk equipped with a tabu list is not a Markovian process. However, when using $FIFO_m$ (for some positive integer m), the next visited vertex only depends on the current one and on the m previous steps: this is called a *Markov chain with internal states*; refer to [10, p. 177].

3 Finite Mean Hitting Times

For every update rule R_m of length m, the *hitting time* $H_y(R_m)$ of every vertex y is the random number of steps needed by a walker to reach y. It is defined as the first instant when the tabu random walk $(X_n(R_m))_{n\geqslant 0}$ reaches y: $H_y(R_m) = \inf\{n \geqslant 0 : X_n(R_m) = y\}$. The *mean hitting time* $E_{(x,\varepsilon)}H_y(R_m)$ is the mean hitting time of y when the walker starts at x with an empty tabu list ε. Our goal is to characterize the class of update rules that have finite mean hitting times, for all graphs and all vertices x and y.

Definition 1 *We define the two following conditions:*

- $(\mathbf{C_1})$ *For every tabu list t and every position of the walker x, if t is not full and does not contain x, then applying the update rule on x and t results in inserting x in t with a positive probability.*
- $(\mathbf{C_2})$ *For every tabu list t and every position of the walker x, if t is full and does not contain x, then applying the update rule on x and t results in removing rightmost element from t with a positive probability.*

The conjunction of $(\mathbf{C_1})$ and $(\mathbf{C_2})$ is a necessary and sufficient condition that ensures the finiteness of all mean hitting times for every associated tabu random walk on every graph. The update rules $FIFO_m$, LRU_m and $RAND_m$ satisfy both $(\mathbf{C_1})$ and $(\mathbf{C_2})$. On the contrary, an update rule of length 1 that keeps the unique element of the tabu list until the corresponding vertex is visited again does not satisfy $(\mathbf{C_2})$. Using such an update rule may lead to an infinite mean hitting time. Indeed, on the flower graph F_1 given in Figure 1,[2] if the walker starts at vertex 1 with the empty tabu list and does not hit vertex 0 at his first step, then its tabu list is 1 forever. Thus, the walker never comes back to vertex 1 and, consequently, the walker will never reach vertex 0. Hence, the mean hitting time to reach 0 from 1 is infinite.

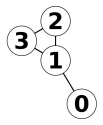

Fig. 1. The flower F_1.

Theorem 2 *Let R_m be an update rule of length m. The mean hitting time $E_{(x,\varepsilon)}H_y(R_m)$ is finite on all graphs, for all vertices x and y, if and only if R_m is either trivial or satisfies $(\mathbf{C_1})$ and $(\mathbf{C_2})$.*

Proof. $(\Rightarrow)$ We proceed by establishing the contrapositive. Consider a non trivial update rule that either does not satisfy $(\mathbf{C_1})$ or does not satisfy $(\mathbf{C_2})$. Consider the graph with vertex set $\{0, \ldots, m+2\}$ such that the vertices $1, \ldots, m+2$ form a clique and the vertex 0 has the vertex 1 as unique neighbor. We assume that the walker starts at vertex 1 and does not hit the vertex 0 at its first step. Since the update rule is not trivial, the vertex 1 is inserted in the tabu list with positive probability. Assume that this latter event is realized. Then, the walker needs to return to the vertex 1 before hitting the vertex 0.

[2] F_ℓ is defined in Section 5, for all values of ℓ.

Since all neighbors of the vertex 1 distinct from 0 have degree $m+1$, the removal of the vertex 1 is needed in order to hit the vertex 0. We now show that the vertex 1 stays forever in the tabu list.

- First, note that with $m = 1$, $(\mathbf{C_1})$ is satisfied, since the update rule is not trivial and when the tabu list is not full, it is empty. So, we need only to consider the case where $m > 1$ and that $(\mathbf{C_1})$ is not satisfied. In this case, we can show that with positive probability, the walker can reach a position with a non-full tabu list from which the tabu list will remain almost surely constant: this tabu list will never be full. Consequently, in this scenario, we cannot remove any element in the tabu list, because the only way to do so requires the tabu list to be full.
- Assume that $(\mathbf{C_1})$ is satisfied, but not $(\mathbf{C_2})$. Similarly, we can show that with positive probability, the walker can reach the position $m+1$ with the full tabu list $m, m-1, \dots 1$ (the rightmost element is 1). From this configuration, since $(\mathbf{C_2})$ is false, the rightmost element of the tabu list (vertex 1) will never be removed from the list.

Hence, in both cases, with positive probability, the hitting time of vertex 0 is infinite. This implies that the mean hitting time of vertex 0 is infinite.

($\Leftarrow$) Suppose first that the update rule is trivial. The walker then performs a simple random walk. Since the simple random walk on each finite and connected graph is positive recurrent, all mean hitting times are finite.

Assume now that the update rule is not trivial and satisfies both $(\mathbf{C_1})$ and $(\mathbf{C_2})$. Consider a graph G. An *essential communicating class* (see [11, p. 16]) for the corresponding tabu chain[3] is a set of ordered pairs formed by a vertex and a tabu list such that the tabu chain cannot exit from and visits each of its elements infinitely often. Let x and y be two vertices of G. Starting from x with empty tabu list, the walker reaches an essential communicating class of the tabu chain in mean finite time. Hence, it suffices to show that the mean hitting time to y is finite, starting from some state of an essential communicating class. Therefore, we assume now that the tabu chain starts from a state (z,t) of a communicating class $\mathcal{C}$. (*N.b.*, starting the walk from (z,t), t may not be empty despite all vertices in t have never been visited.)

The restriction of the tabu chain to $\mathcal{C}$ is a *positive recurrent Markov chain* (see [11, p. 11]). So it suffices to show that there exists a tabu list s such that (y,s) belongs to $\mathcal{C}$. Besides, since G is connected, we may assume that y is a neighbor of z. We proceed by contradiction by assuming that there exists one neighbor w of z that will never be reached by the walker. As (z,t) is in $\mathcal{C}$, z is visited infinitely often, so w must have an occurrence in t. Besides, there exists a neighbor of z, w', that does not have any occurrences in t (otherwise t contains all the neighbors of z and there is a positive probability that the walker reaches w). So, when located at z with tabu list t, the walker reaches w' with positive probability. If this event is realized, then we obtain a state (w',u), where u does not contain any occurrence of w'. Moreover, as (z,t) is in $\mathcal{C}$, (w',u) is also in $\mathcal{C}$. Assume now that the walker is in w' with tabu list u. We distinguish between two cases:

- First, assume that u is not full. According to $(\mathbf{C_1})$, w' is inserted in the tabu list with positive probability. Hence, we reach a state (w'',u') in $\mathcal{C}$ with $|u'| = |u|+1$.

[3] Recall that the tabu chain is the Markov chain $(X_n, T_n)_{n \geqslant 0}$.

Since, by definition, the length of the tabu list cannot decrease and since the tabu chain revisits (w', u) almost surely, we reached a contradiction.

– Second, assume that u is full. According to $(\mathbf{C_2})$, the last element of the tabu list is removed with positive probability. On one hand, (w', u) is visited infinitely often and we remove infinitely often the last element of the current tabu list. On the other hand, the walker never reaches the vertex w. Hence the number of occurrences of w in its tabu list decreases. Eventually, all occurrences of w in the tabu list are removed. Since (z, t) is visited infinitely often, the tabu list t cannot contain any occurrence of w. Once again, we reached a contradiction.

4 Optimal Update Rule for m-Free Graphs

For each positive integer m, we identify a non trivial class of graphs on which $FIFO_m$ gives the smallest mean hitting time, among all update rules of length at most m. Then, we describe a class of update rules that yield tabu random walks with the same law than $FIFO_m$ on this class of graphs.

Definition 3 (m-Free Graphs) *Let m be a positive integer. A graph is m-free if there does not exist any path $x_0, \ldots, x_k$ with length $k \geqslant 1$ that satisfies the four following conditions:*

1. *The vertex x_k has degree at least two.*
2. *For all integers j in $\{0, \ldots, k-1\}$, $x_j \neq x_k$.*
3. *All neighbors of x_k of degree at least two belong to $\{x_0, \ldots, x_{k-1}\}$.*
4. *$k + 2d \leqslant m$, where d is the number of neighbors of x_k of degree one.*

The idea behind Definition 3 is that for each m-free graph with $m \geq 2$, a walker who uses $FIFO_m$ always selects a destination that is not in his current tabu list. Hence, he avoids cycles of size less than or equal to m.

As direct consequences of the definition, note that *all* graphs are 1-free and that all $(m+1)$-free graphs are also m-free, for every positive integer m. Furthermore:

– A graph is 2-free if and only if it does not contain any triangle with one vertex of degree exactly two, namely, if and only if it does not possess any vertex x such that $V_x = \{y, z\}$ and $\{x, z\} \subseteq V_y$, where V_x and V_y are, respectively, the set of neighbors of x and y.
– Every $(m+1)$-*regular* graph, namely with all vertices of degree $m+1$, is m-free. Indeed, assume that $x_0, \ldots, x_k$ is a path that satisfies the four conditions stated above. Since $d = 0$, we infer that $k \leqslant m$. Yet, x_k has $m + 1 > k$ neighbors of degree at least two while the set $\{x_0, \ldots, x_{k-1}\}$ has k elements. Thus, the condition 3 is not satisfied and we reach a contradiction.
– Every graph with *girth* strictly greater than $m+1$, that is, where every cycle has at least $m+2$ edges, is m-free.

The following theorem states that for m-free graphs, and with no more than m memories, $FIFO_m$ is the optimal update rule.

Theorem 4 *Let m be a positive integer and R_k be an update rule of length $k \leqslant m$. On a m-free graph, for every two vertices x and y, $E_{(x,\varepsilon)}H_y(FIFO_m) \leqslant E_{(x,\varepsilon)}H_y(R_k)$.*

Note that being m-free is not a necessary condition to have, for every two vertices x and y, $E_{(x,\varepsilon)}H_y(FIFO_m) \leqslant E_{(x,\varepsilon)}H_y(R_k)$ (Theorem 4 only gives a sufficient condition). Indeed, consider the clique with vertex set $\{0,1,2\}$. The path $(x_0,x_1,x_2) = (0,1,2)$ ensures that the graph is not 2-free. Yet, for every two distinct vertices x and y and for every update rule R, $E_{(x,\varepsilon)}H_y(R) \geqslant 3/2$, while $E_{(x,\varepsilon)}H_y(FIFO_2) = 3/2$.

We now sketch a proof of Theorem 4.

Proof. Let m be a positive integer and let G be a m-free graph. Since G is m-free, we can show, by contradiction, that for every tabu random walk associated to $FIFO_m$, the walker does not visit a vertex if at least one occurrence of that vertex is in the current tabu list, except if the current vertex has degree one. Now, consider an ordered pair (x,y) of vertices of G and an update rule R_k of length k in $\{0,\ldots,m\}$. Let $(X_n(R_k))_{n=0}^{H_y(R_k)}$ be the associated tabu random walk on G that starts at x with empty tabu list and stops when it reaches y. Assume that i is an integer such that $X_i(R_k)$ has degree at least two while $X_{i+1}(R_k)$ has an occurrence in the tabu list $T_i(R_k)$. In particular, this implies that all neighbors of $X_i(R_k)$ have an occurrence in $T_i(R_k)$. We set $j = \min\{\ell \geqslant 2 \,:\, X_i(R_k) = X_{\max\{i-\ell,0\}}(R_k)\}$. We remove all steps of $(X_n(R_k))_{n=0}^{H_y(R_k)}$ from $i-j+1$ to i. By applying iteratively this operation, we obtain a random walk $(\widetilde{X}_n)_{n=0}^{\widetilde{H}_y}$ such that the walker does not visit a vertex if at least one occurrence of that vertex is in the current tabu list, except if the current vertex has degree one.

Applying the same scheme recursively, we can prove that $(\widetilde{X}_n)_{n=0}^{\widetilde{H}_y}$ follows the same law than the first $H_y(FIFO_m)$ steps of the tabu random walk $(X_n(FIFO_m))_{n=0}^{H_y(FIFO_m)}$ associated to the update rule $FIFO_m$, starting at (x,ε). Therefore, we infer that $E_{(x,\varepsilon)}\widetilde{H}_y = E_{(x,\varepsilon)}H_y(FIFO_m)$. By construction $\widetilde{H}_y \leqslant H_y(R_k)$ then we obtain $E_{(x,\varepsilon)}H_y(R_k) \leqslant E_{(x,\varepsilon)}H_y(FIFO_m)$.

From the above theorem, we can deduce that the non-backtracking random walk ($FIFO_1$) is the fastest among all update rules of length less or equal to 1, since every graph is 1-free.

Corollary 5 *For every update rule R of length 0 or 1 and for every two vertices x and y of every graph, $E_{(x,\varepsilon)}H_y(FIFO_1) \leqslant E_{(x,\varepsilon)}H_y(R)$.*

Proposition 6 completes Corollary 5 and states that $FIFO_1$ is the only update rule of length 1 such that, on all graphs, all hitting times are smaller than those associated to the simple random walk, here represented by $FIFO_0$.

Proposition 6 *If R is an update rule of length 1 distinct from $FIFO_1$, then there exists a positive integer ℓ such that on the graph F_ℓ,*[4] *$E_{(1,\varepsilon)}H_0(FIFO_0) < E_{(1,\varepsilon)}H_0(R)$.*

Theorem 4 shows that $FIFO_m$ is the best update rule of length at most m for m-free graphs. In Theorem 7 below, we characterize a larger class of update rules which are optimal policies in that specific case; for example, LRU is in this class as stated by Corollary 8 (as a direct application of Theorem 7).

[4] Flower graphs F_ℓ are defined in Section 5.

Theorem 7 *Consider a positive integer m and a m-free graph. Every update rule of length k in $\{0, \ldots, m\}$ yields tabu random walks with the same law as those associated to $FIFO_k$ if and only if it satisfies the two following conditions:*

- *If the tabu list is not full and does not contain the current vertex, then it is inserted.*
- *If the tabu list is full and does not contain the current vertex, then the last element is removed and the current vertex is inserted.*

Proof. The two conditions stated above imply that the walker never has any occurrence of its current position in its tabu list when the graph is m-free. Hence, the law of the tabu random walk is entirely determined by the update rule when the current vertex does not have any occurrence in the tabu list.

Conversely, if an update rule of length k differs from $FIFO_k$ when the current vertex does not have any occurrence in the tabu list, then the associated tabu random walks on a clique with $m + 3$ vertices (which is a m-free graph) follow two distinct laws.

Corollary 8 *Let m be a positive integer. On every m-free graph, for every integer k in $\{0, \ldots, m\}$, the update rules LRU_k and $FIFO_k$ yield tabu random walks with same law.*

As a conclusion, within the class of m-free graphs and no more than m memories, we identified a class of optimal update rules which contains $FIFO_m$. For other cases, namely, for graphs that are not m-free or using more that m memories, the question is still open.

5 Impact of the Length of the Update Rules

We study the effect of the size of the memory of the walker, on the mean hitting times using 3 collections of update rules: $(FIFO_m)_{m\geqslant 0}$, $(LRU_m)_{m\geqslant 0}$, and $(RAND_m)_{m\geqslant 0}$. Our results shows that there is no general trend, *i.e.*, having more memory does not always increase the performances. To see this, we present comparisons based on four particular collections of graphs: the cliques $(K_r)_{r\geqslant 2}$, the lollipops $(L_r)_{r\geqslant 3}$, the lines $(P_r)_{\geqslant 2}$, and the flowers $(F_\ell)_{\ell\geqslant 1}$. For a given update rule, R_m of length m, we study the mean hitting time $h(R_m) = E_{(1,\varepsilon)}H_0(R_m)$ of the vertex 0 for a walker that starts from vertex 1 with an empty tabu list. The next paragraphs present the graphs and the comparisons. All the results are summed up in Proposition 9.

m-Free Graphs. Within the class of m-free graphs and with a $FIFO$ update rule of length less than m, the greater the length is, the more efficient the algorithm is. Precisely, for every positive integer m, and for all integers k such that $k \leq l \leq m$, we have $h(FIFO_l) \leq h(FIFO_k)$ on every m-free graph. This is a direct application of Theorem 4.

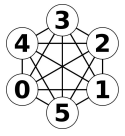

Fig. 2. The clique K_6.

Cliques. For every integer $r \geq 2$, the clique K_r is the complete graph with vertex set $\{0, \ldots, r-1\}$: each vertex is neighbor of all other vertices. As an example, the clique K_6 is drawn in Figure 2. Note also that K_r, for $r > 2$ is $(r-2)$-free.

In our technical report [9], we compute on the clique K_r an *analytic expression* of $h(R_m)$ for a family of update rules that contains $(FIFO_m)_{m\geqslant 0}$, $(LRU_m)_{m\geqslant 0}$, and $(RAND_m)_{m\geqslant 0}$. Using this expression, we compare all these update rules and conclude that the larger the length of the update rule is, the smaller the mean hitting times are.

Lollipops. For every integer $r \geqslant 3$, L_r denotes the lollipop graph with vertex set $\{0, \ldots, r-1\}$ such that the vertices $2, \ldots, r-1$ form a clique with $r-1$ elements and the vertex 1 has 0 and 2 as neighbors. As an example, the lollipop L_6 is drawn in Figure 3.

We use lollipop graphs to compare:

1. $RAND_3$ against $RAND_2$;
2. $RAND_k$ with $k \geq 4$ against $RAND_m$ with $1 \leq m < k$; and
3. LRU_k with $k \geq 3$ against LRU_m with $1 \leq m < k$.

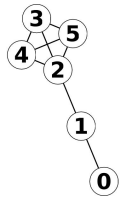

Fig. 3. The lollipop L_6.

A walker on the lollipop L_r that starts at the vertex 1 and does not hit the vertex 0 at its first move, must stay on the set of vertices $\{3, \ldots, r-1\}$ until its tabu list is full. Thus, increasing the length of the update rule may raise the mean hitting time and this is actually the case for the comparisons above.

Lines. For every integer $r \geqslant 2$, P_r denote the graph line with vertex set $\{0, \ldots, r-1\}$ and edge set $\{\{i, i+1\},\ i \in \{0, \ldots r-2\}\}$. As an example, the graph P_4 is drawn in Figure 4.

Line graphs are used to compare:

1. $FIFO_k$ with $k \geq 3$ against $FIFO_m$ with $0 \leq m < k$;
2. LRU_k with $k \geq 3$ against LRU_0 (the simple random walk);
3. $RAND_k$, with $k \geq 4$ against $RAND_0$ (the simple random walk); and
4. $RAND_3$ against $RAND_m$ with $m = 0, 1$.

Fig. 4. The line P_4.

In the above comparison, the length of the update rule raises the mean hitting time on a line. Indeed, assume that m is a positive integer and consider a walker on P_r that starts at the vertex 1 and does not hit the vertex 0 at its first move. First, the walker must go to the end of the line, that is to say the vertex $r-1$, without backtracking. Then, its tabu list is almost surely $(r-2) \cdots (r-m-1)$. Thus, the walker performs a simple random walk, until it reaches a vertex with a neighbor not included in the tabu list. The duration of the simple random walk behavior increases with the length of the update rule. Next, the walker goes to the vertex 0 without backtracking.

Flowers. For all positive integers ℓ, we define the graph F_ℓ with vertex set $\{0, \ldots, 2\ell+1\}$ as follows. Initially, the vertices 0 and 1 are isolated and for each integer x in $\{1, \ldots, \ell\}$, the vertices $2x$ and $2x+1$ are neighbor. Then, each vertex is linked to the vertex 1, except the vertex 1 itself. As an example, the flower F_3 is drawn in Figure 5.

We deal with the flower graphs to compare:

1. $FIFO_2$ against $FIFO_k$, $k = 0, 1$;
2. LRU_2 against LRU_k, $k = 0, 1$; and
3. $RAND_2$ against $RAND_k$, $k = 0, 1$.

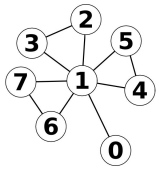

Fig. 5. The flower F_3.

For the above comparison, increasing the length of the update rule raises the mean hitting time in a flower graph. Indeed, consider a walker in F_ℓ that starts at the vertex 1

and does not hit the vertex 0. Without loss of generality, assume that he reaches vertex 2 at its first move. Now, the mean return time to the vertex 1 increases with the length of the update rule. After having returned to the vertex 1, the walker either hits 0 or reaches again the previous situation and must return again to vertex 1.

Results. The next proposition summarizes the above results and presents a complete view of the impact of the length on the update rules for $(FIFO_m)_{m\geqslant 0}$, $(LRU_m)_{m\geqslant 0}$ and $(RAND_m)_{m\geqslant 0}$.

Proposition 9 *On the graph written at k-th row and m-th column,*

1. $h(FIFO_k) > h(FIFO_m)$*:*

k \ m	0	1	2	$m \geqslant 3$
0	$\emptyset$	K_3	K_3	K_3
1	$\emptyset$	$\emptyset$	K_3	K_3
2	F_7	F_4	$\emptyset$	K_4
$k \geqslant 3$	P_4	P_4	P_4	$\begin{cases} P_{m+2} & \text{if } m < k\,, \\ \emptyset & \text{if } m = k\,, \\ K_{k+2} & \text{if } m > k\,. \end{cases}$

2. $h(LRU_k) > h(LRU_m)$*:*

k \ m	0	1	2	$\geqslant 3$
0	$\emptyset$	K_3	K_3	K_3
1	$\emptyset$	$\emptyset$	K_3	K_3
2	F_7	F_4	$\emptyset$	K_4
$\geqslant 3$	P_4	L_4	L_4	$\begin{cases} K_{k+2} & \text{if } m > k\,, \\ \emptyset & \text{if } m = k\,, \\ L_{m+2} & \text{if } m < k\,. \end{cases}$

3. $h(RAND_k) > h(RAND_m)$*:*

k \ m	0	1	2	3	$m \geqslant 4$
0	$\emptyset$	K_3	K_3	K_3	K_3
1	F_5	$\emptyset$	K_3	K_3	K_3
2	F_5	F_6	$\emptyset$	K_4	K_4
3	P_4	P_4	L_5	$\emptyset$	K_5
$k \geqslant 4$	P_4	L_5	L_5	L_5	$\begin{cases} K_{k+2} & \text{if } m > k\,, \\ \emptyset & \text{if } m = k\,, \\ L_{m+2} & \text{if } m < k\,. \end{cases}$

In each table, the symbol $\emptyset$ means that no such graph exists. Actually, symbol $\emptyset$ appears in two disjoint cases : when $k = m$ or when we compare $FIFO_0$ to $FIFO_1$. (In this latter case, for all graphs and all vertices x and y, $E_{(x,\varepsilon)}H_y(FIFO_0) \geqslant E_{(x,\varepsilon)}H_y(FIFO_1)$, by Corollary 5.)

The above proposition shows that, for the three studied collections of update rules, there is no general trend: increasing the length of the memory does not always lead to a gain of performance and may even be a penalty in some cases.

6 Conclusion and Perspectives

We analyzed classes of tabu random walks characterized by their update rules. Our goal was to study the impact of the choice of an update rule on the performance of tabu random walks. We focus on classes of update rules, for which we give a necessary and sufficient condition that ensures the finiteness of the mean hitting time of the associated tabu random walk on every graph. Then, we exhibit non-trivial classes of graphs, namely the m-free graphs, on which we exhibit the optimal update rules among those of length at most m. Finally, we study the impact of the tabu list length on the efficiency of the walk. This latter study shows that, except in one case (namely $FIFO_1$), modifying the length of the tabu list does not guarantee a better hitting time in all cases.

Our results could be extended to a larger class of update rules for which more removals are allowed; for example, the list could be reset regularly. A preliminary study shows that this extension should be carefully done: we believe that the necessary and sufficient condition of Theorem 2 could be adapted and that the result on m-free graphs still holds. In future works, we would also like to compare the relative performance of different update rules of same length. As a first step, we know that given a positive integer m, $FIFO_m$ is faster than LRU_m on line graphs, and LRU_m is faster than $RAND_m$ on lollipop graphs (see [9]).

References

1. Ochirkhand, E.O., Minier, M., Valois, F., Kountouris, A.: Resilient networking in wireless sensor networks. Research report, Inria (2010)
2. Ponsonnet, C.: Secure probabilistic routing in wireless sensor networks. Master thesis, Laboratoire Verimag (2011)
3. Lovász, L.: Random walks on graphs: A survey. In D. Miklós, V.T. Sós, T.S., ed.: Combinatorics, Paul Erdős is Eighty. Volume 2 of Bolyai Society Mathematical Studies., János Bolyai Mathematical Society (1993) 1–46
4. Aldous, D., Fill, J.: Reversible Markov Chains and Random Walks on Graphs. Book in preparation, `http://www.stat.berkeley.edu/~aldous/RWG/book.html` (20XX)
5. Slade, G.: The self-avoiding walk: A brief survey. To appear in Surveys in Stochastic Processes, Proceedings of the Thirty-third SPA Conference in Berlin, 2009, to be published in the EMS Series of Congress Reports, eds. J. Blath, P. Imkeller, S. Roelly (2010)
6. Li, K.: Performance analysis and evaluation of random walk algorithms on wireless networks. In: IPDPS Workshops. (2010) 1–8
7. Altisen, K., Devismes, S., Lafourcade, P., Ponsonnet, C.: Routage par marche aléatoire à listes tabous. In: Algotel'2011 (23–26 Mai 2011, Cap Estérel). (2011)
8. Ortner, R., Woess, W.: Non-backtracking random walks and cogrowth of graphs. Canad. J. Math. **59**(4) (2007) 828–844
9. Altisen, K., Devismes, S., Gerbaud, A., Lafourcade, P.: Comparisons of mean hitting times for tabu random walks on finite graphs. Technical report, Laboratoire Verimag (2011) `http://www-verimag.imag.fr/TR/TR-2012-6.pdf`.
10. Hughes, B.: Random walks and random environments. Volume 1. Oxford University Press (1995)
11. Levin, D., Peres, Y., Wilmer, E.: Markov chains and mixing times. American Mathematical Society (2009)