Speeding up random walks by choosing unvisited edges or vertices[⋆]

Petra Berenbrink[1], Colin Cooper[2], and Tom Friedetzky[3]

[1] School of Computing Science, Simon Fraser University, Burnaby, Canada
[2] Department of Computer Science, King's College, London, U.K.
[3] School of Engineering and Computing Sciences, Durham University, Durham, U.K.

Abstract. In this paper we use simulations to investigate the vertex cover time of several related random walk related processes. The modifications are as follows: Our first process (E-process) uses unvisited edge whenever possible. Our second process (V-process) uses unvisited vertices whenever possible. Our third process assumes that every vertex has a list with all its neighbours in a random order. Each time a vertex is left during the walk, the next neighbour w.r.t. this permutation is used (modulo degree), in a round-robin fashion.
We investigate the performance of these processes on the following regular graphs of constant degree: the two-dimensional torus and random d-regular graphs, RR[d]. In the case of $d \geq 4$ even it is known theoretically that (almost surely) the E-process has cover time linear in the number of vertices; but these processes are not generally well understood. Our simulations show that in most settings the E-process has the smallest cover time. We will also see that the E-process is especially fast for RR[d] with constant and even d.

1 Introduction

In this paper we compare the cover time of several modified random walk processes. A random walk P on a graph $G = (V, E)$ is defined as follows. At $t = 0$ the walk starts at an arbitrary vertex v_0. In step $t \geq 1$ the process moves from v_{t-1} to a neighbouring vertex v_t. In the case of a standard (unbiased) random walk, the vertex v_t is chosen uniformly at random from the neighbours of v_{t-1}. The *vertex cover time* (which we will often call cover time in the following) $C_P(G)$ of a connected graph is defined as follows. Let $C_P(G, u)$ be the (expected) number of steps needed for P to visit all vertices of V starting from u, then $C_P(G) = \max_{u \in V} C_P(G, u)$. The *edge cover time* $C^e_P(G)$ of a connected graph is defined similarly. Let $C^e_P(G, u)$ be the (expected) number of steps needed for P to visit all edges of V starting from u, then $C^e_P(G) = \max_{u \in V} C^e_P(G, u)$.

For a standard random walk, the cover time $C(G)$ of a connected n vertex graph G satisfies the bounds $n \log n \leq C(G) \leq (4/27)n^3$ [10], [11]. For a general overview of discrete random walks, see e.g. [1,16]. The theoretical minimum cover time of any walk process is n, as each vertex needs to be visited. Strategies that

[⋆] Submitted to SEA'12

might be used to reduce the cover time of a walk process to $o(n \log n)$ include preferring unvisited edges or vertices. In this paper we consider the following three processes, namely the edge, vertex, and Propp processes; and which we now define.

The edge process (or "E-process") moves in step t to a neighbour v_t of the current vertex v_{t-1} chosen as follows: if there are *unexplored edges* incident with the current vertex v_{t-1}, then pick one according to some rule $\mathcal{A}$ and make a transition along this edge, otherwise move to a random neighbour.

The E-process uses unvisited edges whenever possible, and makes a random walk otherwise. The rule $\mathcal{A}$ specifies which unexplored edge to use next. We consider two different rules, $\mathcal{A}_{\text{rnd}}$ and $\mathcal{A}_{\text{det}}$. $\mathcal{A}_{\text{rnd}}$ chooses randomly among the unvisited edges incident with the vertex. $\mathcal{A}_{\text{det}}$ assumes that the edges of every vertex have a numeric ID, and it chooses the unvisited edge with the smallest ID.

The vertex process (or "V-process") moves in step t to a neighbour v_t of the current vertex v_{t-1} chosen as follows: if there are *unvisited neighbours* adjacent to the current vertex v_{t-1} then pick one according to some rule $\mathcal{A}$ and make a transition to this vertex, otherwise move to a random neighbour.

The V-process is conceptually very similar to the E-process. The last type of process is significantly different:

The Propp process can be seen as a surrogate for choosing unvisited edges, and works as follows. At the start each vertex v creates a permutation σ of its list of neighbours (the rotor). Each time the walk leaves vertex v, it moves to the next neighbour in the rotor order. Thus the permutation σ is used to choose neighbours in a round-robin fashion (modulo degree). We consider a Propp process where the permutation σ is random, and refer to this as a standard process ($\mathcal{P}_{\text{std}}$) in this paper. We also consider a variant namely $\mathcal{P}_{\text{rnd}}$, which generates a new random permutation when the rotor is "used up". More precisely, when the walk has left vertex v degree(v) many times, and the rotor has returned to its start position, (i.e. the permutation σ is "used up"), v creates a new random permutation τ and uses this as the rotor for the next degree(v) many steps. Finally we consider a hybrid process, $\mathcal{P}_{\text{par}}(p)$, which once a vertex v has been left degree(v) many times, tosses a coin and with probability $p \in [0, 1]$ creates a new random permutation τ, and with the probability $1 - p$ reuses the old permutation σ.

We contrast the performance measures obtained for the three walks as described above with those achieved by a standard random walk. We use the two-dimensional torus and random regular graphs with degree d (RR[d]) for our simulation study.

The motivation for this study come from [4], which establishes that for almost all regular graphs of even degree $d \geq 4$, the vertex cover time of the E-process is $O(n)$. The proof techniques of that paper are specific to E-processes on even degree graphs, and have many commonalities with Propp machines. They give no insight into the performance of the E-process on graphs where some vertices are of odd degree, or of the V-process. Moreover it was not clear if a randomized

Propp machine would also have similar properties (linear cover time) for d-even. Our experimental results are as follows:

- Firstly, we think that out experiments are sufficiently correct. The results we obtain for standard random walk cover times match known theoretical results very well (e.g., cover time $\frac{1}{\pi}(n \log^2 n)$ on the n-vertex square torus [9], $\frac{3}{2} n \log n$ on RR[4] [8]).
- The simulations confirm the $O(n)$ cover time of the E-process on random RR$[d]$ (d even), ([4]) as compared to the $\Theta(n \log n)$ cover time for a standard random walk. They suggest that the speed-up occurs even on small graphs (see Figure 1).
- The simulations suggest that with the exception of $d = 3$, the E-process is better than the V-process, which, in turn, is better than a standard random walk on RR$[d]$ for d small. If indeed the cover time of the E-process is directly related to the size of the edge set, then as d increases, the V-process must be faster. The conclusion is that none of these "designer" processes is uniformly good, and their effectiveness depends on the graph structure
- If we compare the Propp process $\mathcal{P}_{\text{std}}$, $\mathcal{P}_{\text{rnd}}$ and $\mathcal{P}_{\text{par}}(p)$ on RR$[d]$, the standard version $\mathcal{P}_{\text{std}}$ has the smallest cover time. The conclusion is that a single (one-off) randomization of the rotor is optimal. Moreover, $\mathcal{P}_{\text{std}}$ seems to converge to a cover time independent of d (see eg.g Figure 11).
- For the E-process on RR$[d]$ graphs (d even), the constant $C(d)$ in cover time $C(d)n$ is at most $d/2$, the size of the edge set. This suggests that the process is predominantly exploring unvisited edges, and the role of the random walk is merely to nudge the process back into this mode. Noting that $\mathcal{P}_{\text{std}}$ was better than $\mathcal{P}_{\text{rnd}}$ leads to the curious conclusion that a little randomness can be helpful, but too much randomness less so.
- The simulations suggest that for d-regular graphs, $d \geq 7$ odd, the cover time of the E-process is $o(n \log n)$, which if true would improve on the $n \log n$ lower bound on the cover time of any n vertex graph by a random walk. Similarly, for $d \geq 4$, the process $\mathcal{P}_{\text{std}}$, where the rotor starting position is randomized exactly once, seems to have a $o(n \log n)$ cover time. Experimentally these process appear to be $O(n \log \log n)$.
- Of course these results do not hold for the torus, which has $d = 4$ and girth 4. For the torus, we show that experimentally the E-process outperforms a standard random walk. The plots suggest a cover time of the random walk of $\Theta(n \log^2 n)$, with the E-process covering quicker by some constant factor.

1.1 Known Results

The focus of this study is random walks which exploit previous history by biassing choice towards unvisited edges or vertices whenever possible.

One simple approach, is not to prevent the random walk from backtracking. A non-backtracking walk does not return at step t over the edge traversed at step $t-1$, unless no other move is possible. Alon et al. [2] consider a non-backtracking random walk on d-regular expanders, and establish that this walks is rapidly mixing. This mixing result can be used to show that a non-backtracking random walk has a cover time of $\Omega(n \log n)$ for most d-regular expanders, and does not significantly improve the $n \log n$ lower bound of standard random walks.

As previously mentioned, in [4] the authors analyse the *edge process* formally for certain classes of graphs which they call ℓ-good graphs. These graphs are the connected, even degree graphs of constant maximum degree, with the property that every vertex is in at least one vertex-induced (chord-free) cycle of length ℓ. For this class of graphs they characterise the cover time of the edge-process in terms of the conductance, a measure of the edge expansion rate. For n vertex ℓ-good expander graphs G they show that any edge process on G has cover time of $O((n \log n)/\ell + n)$. They prove that this result is independent of the rule $\mathcal{A}$ used to select unvisited edges. For graphs where $\ell = \Omega(\log n)$, this improves the the cover time of the graph by a factor of $\log n$ compared to the $\Omega(n \log n)$ cover time of *any* weighted random walk. An example of this speedup occurs in RR$[r]$ graphs, $r \geq 4$ even. These graphs are, with high probability, $a \log n$-good for some constant $a > 0$; and thus their cover time is linear in n. However, although the authors prove a vertex cover time upper bound of Cn, the constant C obtained in the proof is rather large. Experimentally we find values of $C < |E(G)|/n = r/2$ even for small n.

The authors of [18] consider the *edge* cover time of graphs by the edge process. For d-regular expanders with a degree $d \geq \log n$ and complete graphs they show that the cover time is linear *in the number of edges.* In [3] the authors study a process called RWC(d) (*Random Walk with Choice*) experimentally. Instead of moving to a random neighbour at each step, RWC(d) selects d neighbours uniformly at random and then chooses to move to the least visited vertex among them. For random geometric graphs their simulations suggest a significant improvement of the cover time. For grids their simulations indicate that there is an unbounded improvement in cover time, even with a choice of only two neighbours. In [**?**] the authors study (experimentally) random walks that do not chose amount their neighbours with a uniform probability.

The Propp model for random walks was introduced in by Propp (see [15]). There is a substantial literature, comparing, for example token distribution where the tokens are spread with Propp random walks with standard random walks. In [19,5] the authors consider the so-called lock-in time which is defined as the time that it takes for the Propp walk until it ends in walking an eulerian tour. See [13] for an overview of these results. In [13] the authors present a general technique to derive upper bounds for the edge and the vertex cover time of random walks. They show that the edge and the vertex cover time is $O(n^{1.5})$ for the two-dimensional torus. For expanders they show an edge cover time of $O(n \log n)$.

2 Experiments

Experimental setup. The simulation engine was programmed in Python 2.7.2, using the NetworkX [17] library version 1.5. The graphs used in the experiments, two-dimensional tori and random regular graphs, were generated using NetworkX's built-in graph generators (the random regular graph generator is based on Steger-Wormald [21]). All plots were produced by Gnuplot [12] version 4.4. All data points are the average of 20 independent runs with identical parameters.

Conventions. All plots are scaled in that when vertex cover times are depicted we divide the corresponding values by $n = |V(G)|$, and where edge cover times are depicted we divide by $m = |E(G)|$. We do occasionally use different scalings but will point that out explicitly.

When not specified otherwise, when saying “cover time” we refer to the *vertex* cover time. When referring to *edge* cover time we will make it explicit. Likewise, when saying “E-process” we mean the E-process with rule $\mathcal{A}_{\text{rnd}}$, that is, pick randomly among the as yet unvisited edges incident to some vertex.

Regarding the plots, we omit ticks on the x-axis. It is understood that we always have the order of the graph on the x-axis, and that it always ranges up to 50,000 vertices.

2.1 The E-process

We already know from [4] that the E-process on random regular graphs (actually on a somewhat broader class) of even degree has a linear cover time, independent of which concrete rule $\mathcal{A}$ is being used. Fig. 1 shows the cover times of the E-process ($\mathcal{A}_{\text{rnd}}$) on random regular graphs of varying degrees. The graph for degrees 4 and 6 (the two solid ones) are clearly constant, whereas for degrees 3,5 and 7 (dashed) we the runtime seems to increase with n. Also, note that the asymptotic constant in the $O(n)$ runtime is between two and three, even for very small values of n. For odd-degree random regular graphs the cover time of the E-process grows with n.

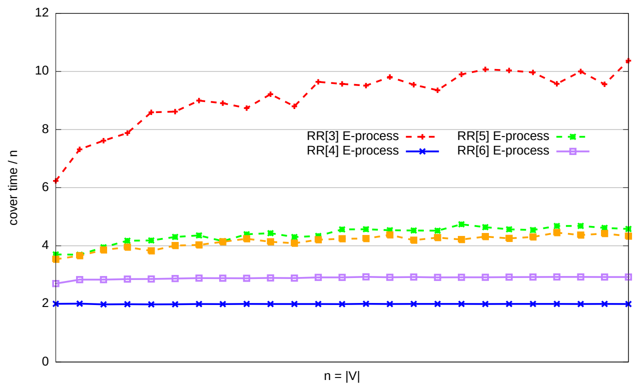

Fig. 1. The E-process ($\mathcal{A}_{\text{rnd}}$) on RR[3]-RR[7]

In Fig. 2 we isolate degree-7 expanders, and scale on the y-axis such that we divide the measured cover time by $f(n) := n \log(n)/\sqrt{\log\log(n)}$. As we can see, the resulting graph looks constant, suggesting that the cover time itself is close to $f(n)$. Of course it is notoriously difficult to fit functions to observed data of this kind, and we make no claims whatsoever that the function used in the figure

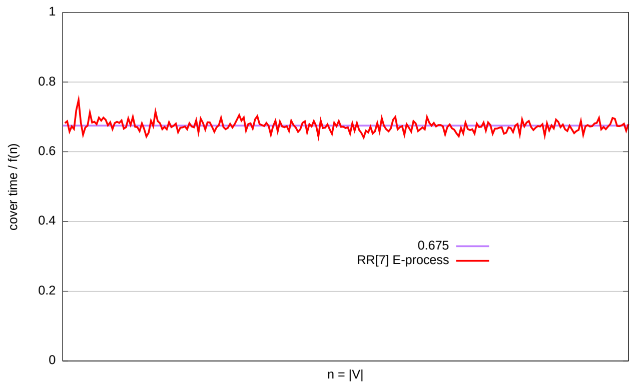

Fig. 2. The E-process ($\mathcal{A}_{\text{rnd}}$) on RR[7] under the microscope

is the "true" one. The fit does, however, very strongly suggest that we do not observe a linear cover time function but an asymptotic cover time that is $\omega(n)$.

In Fig. 3 we compare the cover time of the E-process (again, assuming $\mathcal{A}_{\text{rnd}}$) with that of the standard random walk on random regular graphs. It is known that on this type of graph, the ordinary random walk has a cover time of $\Theta(n \log n)$. Observe how the dashed graphs, representing the standard walk on random regular graphs of various degrees, are invariably above the corresponding E-processes (solid graphs). Hence, the E-process results in a much better cover time, even of graphs of odd degrees. Nevertheless, the plot also suggests that the cover time of the E-process on RR[d] for odd $d \geq 5$ may not be quite $\Omega(n \log n)$.

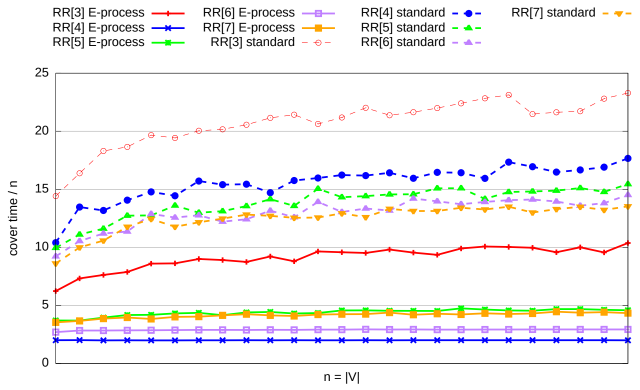

Fig. 3. The E-process ($\mathcal{A}_{\text{rnd}}$) vs standard random walk

In Fig. 4 we study the cover time of the E-process for different rules $\mathcal{A}$ to choose among the as yet unvisited edges. From [4] we know that asymptotically all rules are the same. We depict the cover times of the E-process for $\mathcal{A}_{\text{rnd}}$ and

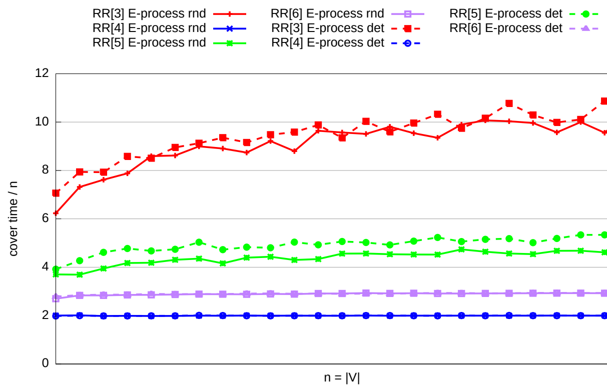

Fig. 4. $\mathcal{A}_{\text{rnd}}$ vs $\mathcal{A}_{\text{det}}$

$\mathcal{A}_{\text{det}}$, respectively. For even degrees (4- and 6-regular in the plot) there is no discernible difference, whereas for odd degrees (3- and 5-regular in the plot) the deterministic rule $\mathcal{A}_{\text{det}}$ behaves slightly worse; the corresponding cover time seems to be larger by a moderate constant factor.

In Fig. 5 (note: y-axis scaled by $(n\log(n))^{-1}$), we compare the cover times of the E-process using $\mathcal{A}_{\text{rnd}}$ on RR[4] and the two-dimensional torus. We also add the same measures for the standard random walk. We know for the cover times that, provably, (i) $\mathcal{A}_{\text{rnd}}$ is linear on RR[4] , (ii) the standard random walk is $\frac{1}{\pi}(n\log^2 n)$ on the n-vertex square torus [9], and (iii) the standard random walk is $\frac{3}{2}n\log n$ on RR[4] graphs [8]. We therefore expect, and indeed see, a graph parallel to the x-axis for (i), a graph proportional to $\Theta(n\log^2 n)$ in (ii) (we also plot in dashed this function with normalising constant coefficient), and a graph proportional to $\Theta(n\log n)$ in (iii) (we plot in dashed this function with normalising constant coefficient as well). It is not clear what the cover time of $\mathcal{A}_{\text{rnd}}$ on the torus is, but it may very well also be in the order of $n\log^2(n)$. The picture suggests again that the E-process outperforms the standard random walk on the torus. Note that the torus can be regards as a very bad input for the E-process since it contains many short cycles.

2.2 The V-process

Recall that the V-process differs from the E-process in that now we give preference to the unvisited *neighbouring vertices* instead of unvisited *incident edges*. Due to lack of space we restrict our attention to one particular V-process, namely the one that picks randomly among the unvisited neighbouring vertices (if any).

In Fig. 6 we compare the cover times achieved by the V-process and those of the E-process (using $\mathcal{A}_{\text{rnd}}$), on RR[d] for $d = 3, \ldots, 7$. Our experiments suggest that the E-process is better for odd degrees (perhaps by some constant factor), but strictly worse for even degrees. In the later case the cover time of the E-process is linear, whereas V-process's cover time does not appear to be linear on any RR[d] graph (apart, of course, from the trivial case $d = 2$).

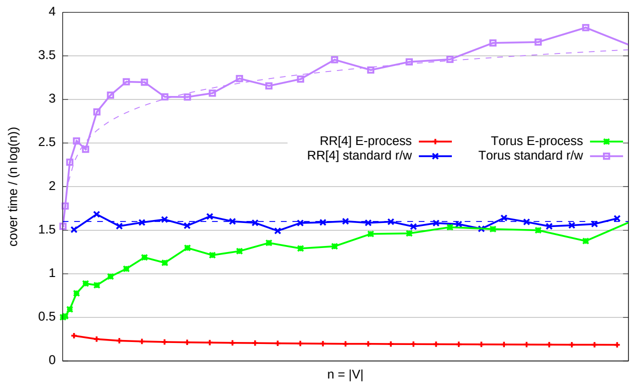

Fig. 5. E-process ($\mathcal{A}_{\mathrm{rnd}}$) and standard random walk on RR[4] and 2D-torus

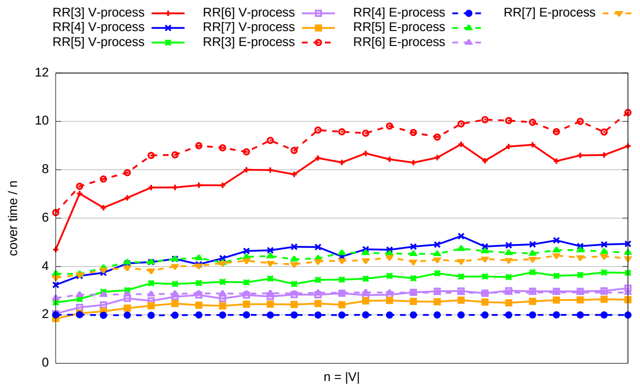

Fig. 6. V-process vs E-process ($\mathcal{A}_{\mathrm{rnd}}$) on RR[3]-RR[6]

In Fig. 7 we compare the E-processes, the V-process and standard random walks on the torus. Both V- and E-process beat the standard walk comfortably. Moreover, similar to random regular graphs, the E-process covers quicker than the V-process. These results suggest that for even-degree regular graphs it may be sensible to opt for the E-process, whereas for odd-degree regular graphs the V-process is slightly ahead.

2.3 The Propp process

We designed the randomised Propp process $\mathcal{P}_{\mathrm{rnd}}$ in the hope that it might outperform the standard one, $\mathcal{P}_{\mathrm{std}}$. Recall that $\mathcal{P}_{\mathrm{rnd}}$ reshuffles a new permutation for a vertex whenever its old permutation has been used up, whereas $\mathcal{P}_{\mathrm{std}}$ fixes a random permutation for each vertex once and for all. It turns out that the advantage is with $\mathcal{P}_{\mathrm{std}}$. In Fig. 8 we depict the cover times of $\mathcal{P}_{\mathrm{rnd}}$ (solid graphs) vs $\mathcal{P}_{\mathrm{std}}$ (dashed). Notice that the y-axis is scaled by $(n \log\log(n))^{-1}$. Although the gap isn't large (all graphs are essentially flat, that is, all cover times seem to be essentially $n \log\log(n)$), it is obvious that $\mathcal{P}_{\mathrm{std}}$ solidly outperforms $\mathcal{P}_{\mathrm{rnd}}$ in every case.

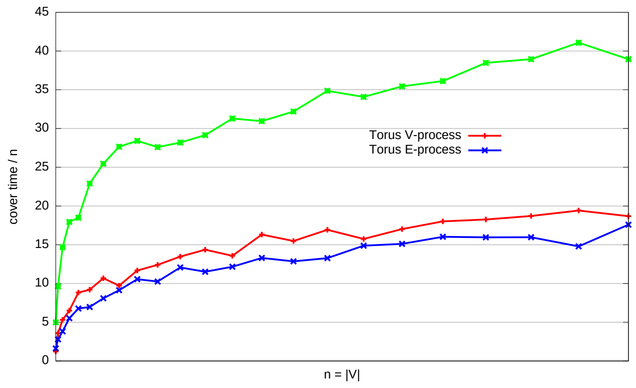

Fig. 7. V-process vs E-process ($\mathcal{A}_{\text{rnd}}$) on 2D-torus

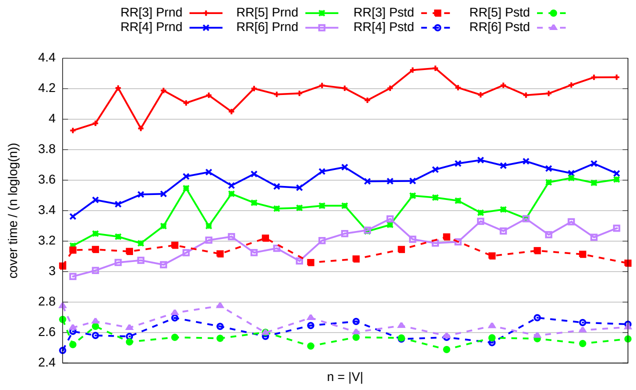

Fig. 8. $\mathcal{P}_{\text{std}}$ vs $\mathcal{P}_{\text{rnd}}$ on RR[3]-RR[6]

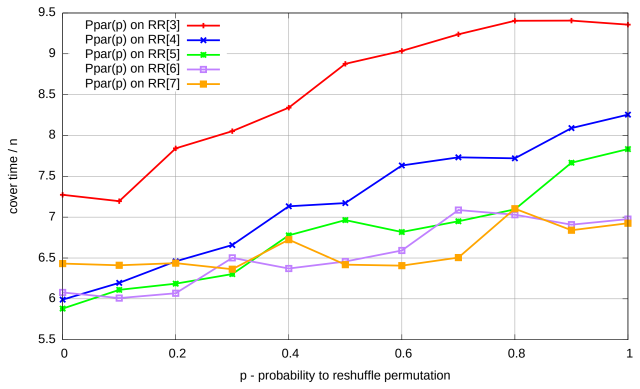

Fig. 9. $\mathcal{P}_{\text{par}}(p)$

This behaviour is further demonstrated in Fig. 9, where we investigate $\mathcal{P}_{\text{par}}(p)$. We fix the number of vertices to 20,000 and depict the cover time of $\mathcal{P}_{\text{par}}(p)$

on RR[d] as a function of $p \in [0,1]$. We have one graph for each value of $d \in \{3,\ldots,7\}$. Recall that in $\mathcal{P}_{\mathrm{par}}(p)$, each time a permutation is used up, a vertex tosses a coin and with probability p rolls a new random permutation. We observe that for each value of d the cover time gets worse as p increases. We seem to have conclusive experimental evidence that introducing the extra randomness not only does not help, but that, in fact, it is actively harmful. Note that in [7] the authors observed a similar effect for broadcasting. Here, a process similar to $\mathcal{P}_{\mathrm{std}}$ where vertices communicate with their neighbours in a round robin fashion, outperforms a protocol where vertices communicate with randomly chosen neighbours.

In Fig. 10 we compare the cover times of the E-process (using $\mathcal{A}_{\mathrm{rnd}}$) with those achieved by the best Propp variant, $\mathcal{P}_{\mathrm{std}}$. It appears that, with the notable exception of $\mathcal{A}_{\mathrm{rnd}}$ on RR[3], the E-process beats the Propp process by at least a constant factor for odd degrees, and by a factor of $\omega(1)$ in the even case.

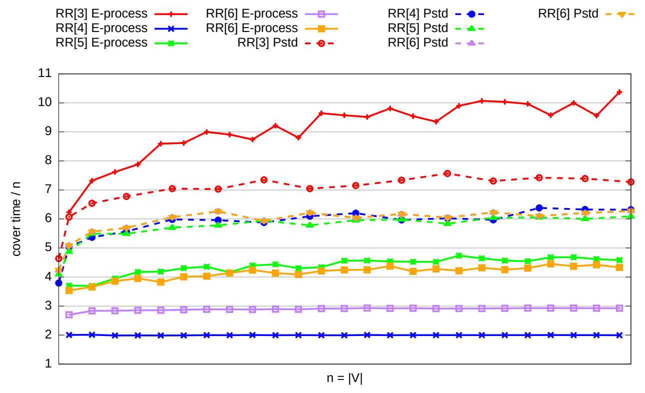

Fig. 10. $\mathcal{P}_{\mathrm{std}}$ vs E-process ($\mathcal{A}_{\mathrm{rnd}}$) in RR[3]-RR[6]

2.4 Edge cover time

So far we have focussed our attention on the vertex cover time. In this section we will briefly discuss some findings for the edge cover time. In Fig. 11 we consider the E-process ($\mathcal{A}_{\mathrm{rnd}}$), the V-process and the standard Propp process ($\mathcal{P}_{\mathrm{std}}$) in RR[$d$] for $d = 3,4,5,6$. We notice that, apart from $d = 3$, the E-process outperforms the others, with the V-process being the worst of the selection (which may not be all that surprising, considering that we wish to cover all *edges*). It is perhaps interesting that the E-process on even-degree graphs is very close to the optimum; the corresponding two graps are only just above the 1-mark (notice that we are now scaling the y-axis by $1/|E(G)|$, that is, being close to 1 means that we are close to finding an Euler tour – hardly any edges are being used twice).

The picture is not quite so bright when looking at the same processes running on the two-dimensional torus, see Fig. 12. Here we do not scale by $|E(G)|^{-1}$ but

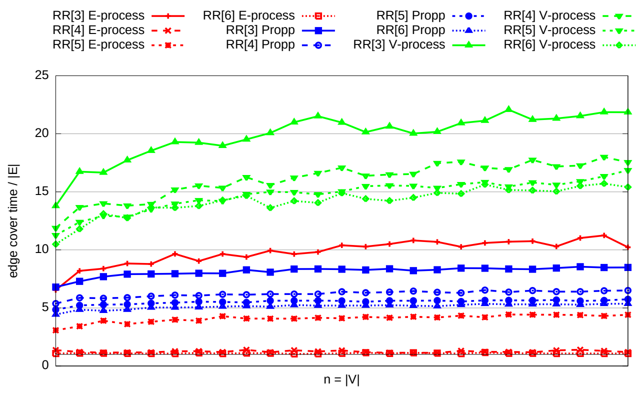

Fig. 11. Edge cover time for E-process ($\mathcal{A}_{\text{rnd}}$) vs V-process vs Propp ($\mathcal{P}_{\text{std}}$) on RR[3]-RR[6]

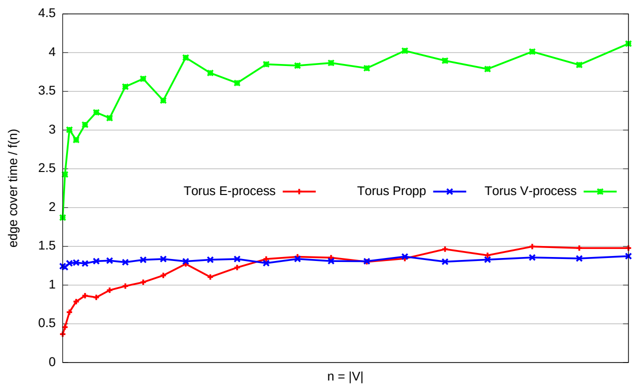

Fig. 12. Edge cover time for E-process ($\mathcal{A}_{\text{rnd}}$) vs V-process vs Propp ($\mathcal{P}_{\text{std}}$) on 2D-torus

rather by $f(n) := (n\log(n)/\sqrt{\log\log n})^{-1}$. It appears as though in the long run the Propp process is actually the best (flat graph indicating an edge cover time of close to $f(n)$), whereas both the V as well as the E-process seem to be growing as $\omega(f(n))$. (Notice that there is only a constant factor between scaling by $f(|V(G)|)$ and $f(|E(G)|)$ as $|E(G)| = \Theta(|V(G)|$.)

3 Conclusions

Several interesting results arose from this study, which require theoretical confirmation. It would be very interesting to show that the cover time of the E-process is not linear on random r-regular graphs wit r being odd, and that the cover time of the V-Process is not linear on random r-regular graphs for any $r \geq 3$. It would also be nice to get tight upper and lower bounds for the cover time of $\mathcal{P}_{\text{rnd}}$, and also for the so-called lock-in time. The latter is defined as the time it takes until all rotors in the nodes are directed in a way that they define an Eulerian cycle (see [19,5]).

References

1. D. Aldous, J. Fill. *Reversible Markov Chains and Random Walks on Graphs*, 2001. `http://stat-www.berkeley.edu/users/aldous/RWG/book.html`
2. N. Alon, I. Benjamini, E. Lubetzky, and S. Sodin. *Non-backtracking random walks mix faster.* Communications in Contemporary Mathematics, 9:585–603, 2007.
3. C. Avin and B. Krishnamachari. *The Power of Choice in Random Walks: An Empirical Study.* MSWiM 06.
4. P. Berenbrink, C. Cooper, and T. Friedetzky. *Random walks which prefer unvisited edges, and exploring high girth even degree expanders in linear time.* Manuscript.
5. S. Bhatt, S. Even, D. Greenberg, and R. Tayar. *Traversing Directed Eulerian Mazes.* Journal of Graph Algorithms and Applications, 6(2), 157173, 2002.
6. C. Cooper and A. Frieze. *The cover time of random regular graphs.* SIAM Journal on Discrete Mathematics, 18, 728-740 (2005).
7. P. Berenbrink, R. Elsässer and T. Sauerwald. Randomised broadcasting: Memory vs. randomness. In Proceedings of LATIN, pp. 306–319, 2010.
8. C. Cooper and A. M. Frieze. *The cover time of random regular graphs.* SIAM Journal on Discrete Mathematics, 18 (2005) 728-740.
9. A. Dembo, Y. Peres, J. Rosen, and O. Zeitouni. *Cover Times for Brownian Motion and Random Walks in Two Dimensions.* Ann. Math., 160 (2004) 433–464.
10. U. Feige. *A tight upper bound for the cover time of random walks on graphs.* Random Structures and Algorithms 6 51–54 (1995).
11. U. Feige. *A tight lower bound for the cover time of random walks on graphs.* Random Structures and Algorithms 6 433–438 (1995).
12. Gnuplot, a freely available, portable command-line driven graphing utility. http://www.gnuplot.info.
13. T. Friedrich and T. Sauerwald. The Cover Time of Deterministic Random Walks. In Electronic Journal of Combinatorics, R167, 2010.
14. S. Ikeda, I. Kubo, N. Okumoto, and M. Yamashita. *Impact of Local Topological Information on Random Walks on Finite Graphs.* Proceedings of the 32st International Colloquium on Automata, Languages and Programming (ICALP), 1054-1067, (2003).
15. M. Kleber. *Goldbug Variations.* Im Mathematical Intelligence, 27, 2005.
16. L. Lovász. *Random walks on graphs: A survey.* Bolyai Society Mathematical Studies, 2:353–397, Budapest, 1996.
17. The NetworkX library for Python. http://networkx.lanl.gov.
18. T. Orenshtein, I. Shinkar *Greedy Random walk.* arXiv:1101.5711v3
19. V. B. Priezzhev, D. Dhar, A. Dhar, and S. Krishnamurthy. *Eulerian walkers as a model of selforganized criticality* Physics Review Letters, 77, 50795082, 1996.
20. I. Safro, P. Hovland, J. Shin, and M. Strout. *Improving Random Walk Performance.* Preprint, ANL/MCS-P1585-0209, Laboratory for Advanced Numerical Simulations, Argonne National Laboratory, USA.
21. A. Steger and N.C. Wormald. *Generating Random Regular Graphs Quickly.* Combinatorics, Probability & Computing 8(4): 377-396 (1999).