Multi-agent random walks for local clustering on graphs

Morteza Alamgir
Max Planck Institute for Biological Cybernetics
Tübingen, Germany
Email: morteza@tuebingen.mpg.de

Ulrike von Luxburg
Max Planck Institute for Biological Cybernetics
Tübingen, Germany
Email: ulrike.luxburg@tuebingen.mpg.de

***Abstract*—We consider the problem of local graph clustering where the aim is to discover the local cluster corresponding to a point of interest. The most popular algorithms to solve this problem start a random walk at the point of interest and let it run until some stopping criterion is met. The vertices visited are then considered the local cluster. We suggest a more powerful alternative, the multi-agent random walk. It consists of several "agents" connected by a fixed rope of length l. All agents move independently like a standard random walk on the graph, but they are constrained to have distance at most l from each other. The main insight is that for several agents it is harder to simultaneously travel over the bottleneck of a graph than for just one agent. Hence, the multi-agent random walk has less tendency to mistakenly merge two different clusters than the original random walk. In our paper we analyze the multi-agent random walk theoretically and compare it experimentally to the major local graph clustering algorithms from the literature. We find that our multi-agent random walk consistently outperforms these algorithms.**

***Keywords*-Local Clustering; Graph Clustering; Random Walk; Mixing Time.**

I. Introduction

Graph clustering is an omnipresent problem in data mining ([5], [13]). Given a particular graph, the goal is to find "clusters" or communities in the graph such that the vertices in the same community are highly connected to each other and only sparsely connected to vertices outside of their community. A big problem in practice is that graphs often have an enormous size, which makes the application of standard graph clustering algorithms such as spectral clustering infeasible.

A promising alternative is represented by the class of local clustering algorithms. Here the goal is not to find a global clustering of the whole graph, but to find "the cluster" of a particular vertex of interest. For example, in social network analysis one might want to investigate the community a particular person belongs to. The advantage of local clustering algorithms is that we never consider the graph as a whole. All computations are local in the sense that we only need to have access to and operate on a small "local" part of the adjacency matrix, which leads to efficient space and time complexity of the algorithm. The prize we pay is that we cannot guarantee that the cluster found by the algorithm is a true, global cluster of the graph; we can only say that it looks like a good cluster based on the local information we have seen.

Recently there has been a lot of interest in local algorithms in the data mining and machine learning community. One of the most popular local clustering algorithms is the Nibble algorithm described in [14]. Given the vertex v of interest, one considers a random walk (RW) on the graph that starts at vertex v. This random walk is biased in the sense that it only jumps along "important" edges, unimportant edges (with low weight) are ignored. The random walk runs until a certain stopping criterion is met, for example when the conductance of the set of visited vertices gets too large. The set of visited vertices is then considered the community of v. For further development based on the nibble algorithm see [2] and [3]. Applications of random-walk based clustering algorithms can be found in many areas, for example in image segmentation [8], [12].

The reason why the random walks is a well-suited tool for local clustering is obvious: they locally explore the neighborhood of a vertex and with reasonably high probability stay inside a cluster rather than transitioning to a different cluster. Moreover, they can be implemented with low space complexity. In our paper, we suggest to use a novel kind of random walk for local exploration, called multi-agent random walk (MARW). Consider a "agents", each agent moving like an individual random walk on the graph. The agents are tied together by a "rope" of length l. As long as the distance between all agents is smaller than l, all of them can move independently from each other. However, they are constrained such that the distance between any two agents is never larger than l.

The reason why the MARW is advantageous for local clustering is that the probability of transitioning between two different clusters is significantly reduced in the MARW compared to the standard RW. Consequently, the MARW discovers tighter clusters than the standard RW.

To the best of our knowledge, the idea of studying a MARW is completely new. Only after we had started working on this question, a group of mathematicians submitted two papers on arxiv that study a similar object called "spider walk", but from a completely different perspective ([6],[7]). Another related paper is the one by [1]. It has an intention opposite of ours: they want to construct an alternative to the standard random walk that travels faster between different clusters. To this end, the authors run several random walks completely independently from each other. In this way one can speed up the global exploration of large graphs. This setting can also be covered in our model, namely as the extreme case of having rope of length infinity. See below for further discussion.

The goal of our paper is to introduce multi-agent random walks and study some of their basic properties. We present several theoretical approaches that show that if we consider a MARW with a agents coupled by a small rope length l, then the probability of transitioning between different clusters decreases with the power of a: if this probability is p for the standard RW, it is about p^a for the MARW. We show in experiments on toy data and on real world data that MARWs are well-suited for local clustering indeed. In particular, they outperform the state-of-the-art algorithms by [14] and [2]. In general, we believe that the technique of multi-agent random walks can be very fruitful in several domains of data mining and machine learning.

II. The multi-agent random walk

Let $G = (V, E)$ be a weighted graph with weighted adjacency matrix $W = (w_{ij})_{ij}$. The graph can be directed or undirected. Denote the degree of vertex v_i by $d_i = \sum_j w_{ij}$. Let the degree matrix D be the diagonal matrix with entries d_i. The simple RW on G is defined as the random walk with transition probabilities $P = D^{-1}W$.

To introduce the MARW we need to define a distance function $\mathrm{dist} : V \times V \to \mathbb{R}$ on the original set of vertices. When working on geometric graphs, that is, the vertices have an interpretation as points in some metric space, we use the distance in the underlying space as distance between vertices. Otherwise we use the shortest path distance.

Formally, a MARW with a agents and rope length l on graph G can be defined as a simple random walk on the state space S consisting of all unordered *valid* a-tuples of the n original vertices (possibly with repeated entries). We call an a-tuple valid if the distance between any two vertices is not greater than l. Then define the probability of transitioning from $s = (s_1, ..., s_a) \in S$ to $t = (t_1, ..., t_a) \in S$ as

$$M(s,t) := \begin{cases} 0 & \text{if } \exists i \text{ such that} \\ & \mathrm{dist}(s_i, t_i) > l \\ \sum_{\sigma \in S_a} \prod_{i=1}^{a} P(s_\sigma(i), t_j) & \text{otherwise.} \end{cases}$$

Here σ runs over the set S_a of all permutations of a elements. Note that the definition is fine for the purpose of defining the MARW, but the matrix M it is completely impractical to work with: in the worst case it has size $O(n^{2a})$ and the permutations are hard to track.

To implement the MARW in practice there are several natural variants: the agents can walk one after the other or all at the same time; and they can be allowed to sit on the same vertex or not. Intuitively, in natural graphs these variants will behave similarly, even though they might behave slightly different in certain pathological examples. However, they have different complexities.

In this paper we focus on the most natural construction: agents move at the same time and are allowed to sit on the same vertex. Assume we start with a agents in locations $v_1, ..., v_a$. We suggest a new step of the MARW by performing a independent simple random walks, one for each agent. Then we check if the new vertices still satisfy the condition on the rope length. If yes, we accept the step, otherwise we discard it. Discarding many suggested new states is not a problem in practice if the out-degree of each vertex is not considerably big.

In applications with large degree vertices, we suggest to use the MARW variant where agents do not move together, but one after the other. Here we randomly select one agent at a time, suggest a new position for this agent by a standard random step and accept it if the rope length condition is satisfied. Note that much less positions are rejected in this version than in the version where all agents move at the same time. It can be seen that both variants of the MARW have similar properties, but the one where we move all agents at the same time is easier to analyze theoretically. This is the case we consider in the rest of the paper.

In order to check the condition of the rope length we need to compute the shortest path distance. This is tractable as we usually work with small rope lengths l and thus only need to compute the shortest paths between vertices in small neighborhoods.

The MARW has two parameters: the number a of agents and the rope length l. Much of the following theory is devoted to understanding the influence and interplay of these two parameters. We are going to characterize the behavior of the MARW by how easily it transitions between different clusters. We provide several arguments that, for reasonably small a and l, the MARW behaves like a "power" of the original RW.

For example, we will see that given a neighborhood graph on a sample from some underlying density g, the MARW with a agents and reasonably small rope length l behaves like a random walk on a sample drawn according to g^a. As the peaks of the density g are amplified in g^a and the valleys get even deeper, the likelihood of transitioning between different clusters of the distribution become smaller for the MARW than for the original RW. This means that clusters are “more expressed” in the MARW than in RW.

Let us introduce one concept that will be used many times in the rest of the paper: the MARW-graph. We have seen above that a MARW on a graph G can be described as a Markov chain on the product space of the vertices of G. To compare the properties of the standard random walk RW and the MARW, it is often convenient to consider the MARW as a simple random walk on some graph G'. For example, it is easy to see that for rope length $l = 0$ and a agents the MARW on G coincides with a simple random walk on the graph G' that has the same vertices as G, but the edge weights w_{ij} are replaced by w_{ij}^a (see next section). More intricate constructions can be made for the case that $l > 0$. In the rest of the paper, whenever we are able to construct a graph G' such that the simple random walk on G' coincides with the MARW on G, we will call G' the *MARW-graph*.

III. How the number of agents affects the mixing time

In this section we want to investigate how the MARW's likelihood to jump outside a cluster behaves in dependency of the number of agents.

A. Intuitive example: a chain graph

We would like to start this section by considering an intuitive example MARW on the directed and weighted graph in Figure 1a with $p < 0.5$. The graph consists of two clusters on the right and left side of the chain, and a boundary vertex B at the middle. Inside each region, the transition probabilities are the same.

For simplicity let us start with **case $\mathbf{l = 0}$**. Here we have a agents that need to move simultaneously. The states of the MARW-graph G' coincide with the vertices of G. Assume all agents sit on vertex i. If P_{ij} denotes the probability that one individual random walker goes along edge ij, the probability that a agents do so simultaneously corresponds to P_{ij}^a. Moreover, there is a considerable probability that the a agents do not agree in which direction they want to walk, namely $1 - \sum_j P_{ij}^a$. If we ignore this self-loop, then the loop-free MARW with rope-length 0 can be interpreted as a simple random walk on the graph G with edge weights w_{ij}^a. In particular, the MARW-graph is just the original graph, but with edge weights taken to the power a (see Figure 1b). Thus, whenever there is a choice of going in the direction of high or low edge weights, the MARW will choose the high weight version even more often than the standard RW.

We now want to argue that in the case of small but positive l the MARW behaves very similarly as in the case $l = 0$. To see this, consider how the center of mass of the agents moves. If all agents move in the same direction, the center of mass does the same. Intuitively, when $l > 0$ we allow the agents to explore the vertices around the mass center, and by increasing l this allowed region will get bigger. We will now show that if the center of mass is not too close to the boundary (compared to l) and there is a “drift” away from the cluster boundary, then the MARW tends to follow this drift. Consequently, it tends to move away from the boundary towards the center of a cluster.

Consider the same graph as above **with l=1 and a=2**. Assume that both agents are in the cluster on the right side. Each agent will go with probability p to the left and probability $1-p$ to the right. Consider the center of mass of the two agents. Depending on the movement of agents it will stay, go to the right or go to the left. If both of agents go to the right, which happens with probability $(1-p)^2$, the center of mass will also move to the right. With similar arguments we can see that it moves to the left with probability p^2. It stays on the same vertex when the two agents want to move in different directions. They either change their place (with probability $p(1-p)$) or cannot move because of violating condition on l (with probability $(1-p)p$). Thus, the probability of staying at the same vertex is given as $2p(1-p)$.
We can conclude that the center of mass of the two agents moves like a simple random walk on another graph G', namely the graph whose vertices correspond to the “mid point” between two consecutive vertices (Figure 1c). In G' we have squared transition probabilities, except for the vertices near the cluster boundary where boundary effects play a role.

This example shows that, compared to the original random walk, the probability that a MARW walks in a region of low density is the square of the probability of the original random walk. **Increasing** l will increase the size of the boundary, but otherwise the argument will work as well. If we consider **$\mathbf{a > 2}$ agents**, one can show in the same example that the ratio of the transition probabilities away from / towards the cluster boundary change from $\frac{1-p}{p}$ to $(\frac{1-p}{p})^{O(a)}$. We do not specify the details because of lack of space.

B. More rigorous: bounding the spectral gap

The main reason that we introduced the MARW is because we want to increase the probability of staying within a community. One way to treat this formally is to analyze the spectral gap of the graph, in particular the second eigenvalue λ_2 of the transition matrix. This eigenvalue is both a measure for how clustered the graph is (for example, it can be used to bound the Cheeger cut) and can also be used to bound the mixing time of the random walk. The smaller λ_2, the more clustered the graph is, and the larger the mixing time is. Thus, if λ_2 of the MARW decreases compared to the standard random walk, then we

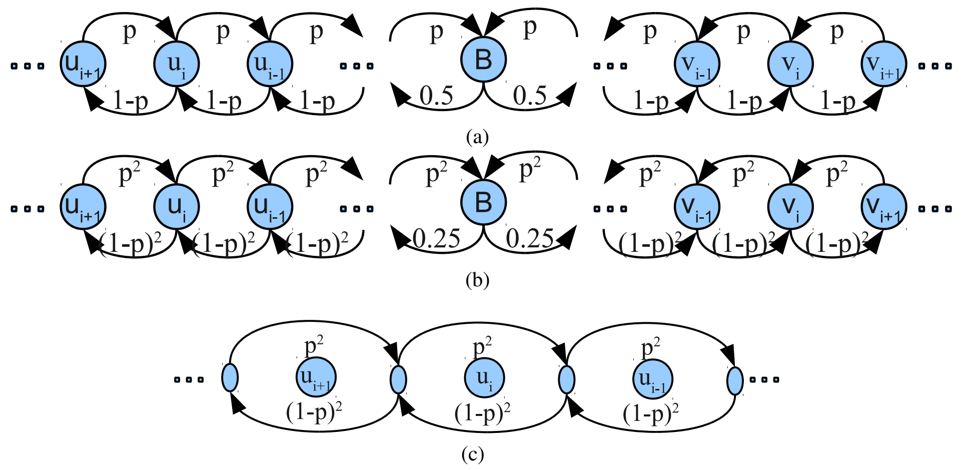


Figure 1: (a) A simple directed graph. (b) Corresponding MARW graph for two agent random walk with $l = 0$. (c) Corresponding MARW graph for two agent random walk with $l = 1$. Initially the two agents are placed in two neighbor nodes. Unnamed nodes correspond to center of masses and the self loops are omitted to keep the figures simple.

can formally conclude that the MARW takes longer to travel between different clusters.

In this section we treat the **special case where $l = 0$.** Our analysis will be mainly useful in weighted graphs. As we have already seen above, for $l = 0$ the MARW-graph G' has the same vertices as G but with edge weights raised to the power of a. We now want to compare the spectral gaps of G and G'. For a fair comparison we need to ensure that we consider the MARW without self-loops (otherwise the mixing time of the MARW increases trivially as the random walk often stays at a vertex without moving). As it is hard to argue about the spectral gap itself, we will compare lower bounds on the spectral gap.

These lower bounds will be provided by means of the Poincaré inequality (see [4] for a general introduction to this technique). The outline of this technique is as follows: for each pair of vertices (X, Y) in the graph we need to select a "canonical path" γ_{XY} in the graph that connects these two vertices. Then we need to consider all edges in the graph and investigate how many of the paths γ_{XY} traverse this edge. We need to control the maximum of this "load" over all edges. Intuitively, a high load means that many paths need to traverse the same edges, that the graph has a bottleneck. Formally, the spectral gap β is related to the maximum average load b as follows (cf. Corollary 1 and 2 in [4]; we adapted their statements to the case of weighted graphs):

$$\beta \geq \frac{\mathrm{vol}(G) w_{min}}{d_{\max}^2 |\gamma_{\max}| b} \tag{1}$$

where $d_i = \sum_j w_{ij}$ is the weighted degree, $\mathrm{vol}(G) = \sum_i d_i$ denotes the volume of the graph, w_{min} the minimal edge weight, d_{max} the maximal degree and b is the maximal load defined by

$$b := \max_{\{e \text{ edge}\}} |\{\gamma_{XY} \mid e \in \gamma_{XY}\}|.$$

Here $|\gamma_{\max}|$ is the maximal number of edges on the canonical paths. We will now compare the Poincaré bounds on the spectral gaps of G and G'.

Theorem 1 (Spectral gap of the MARW in case $l = 0$): Consider the lower bound on the spectral gap provided by the Poincaré technique. If this bound is u for the standard RW, then it is uC^{a-1} for the MARW with a agents and rope length $l = 0$. In general, C is a constant with $C \leq 1$ and for weighted graphs with non-uniform weights, it always satisfies $C < 1$.

Proof: As explained above, the standard random walk and the MARW can both be seen as random walks on the same graph, just the edge weights are different. For this reason, we can use the same set of canonical paths in both graphs (and we do not need to specify it explicitly in our proof). In particular, the maximal load b and the maximal path length $|\gamma_{\max}|$ coincide in both graphs. What is different for both random walks are the minimal and maximal weights and degrees. Denoting the various quantities for G by normal letters and the corresponding ones of the MARW with a tilde on top, we obtain the following relations:

$$\tilde{w}_{min} = w_{min}^a \tag{2}$$

$$\tilde{d}_{max} \le d_{max}^a \tag{3}$$

$$\begin{aligned} vol(\tilde{G}) &= \sum_{\text{edge } e} \tilde{w}_e = \sum_{\text{edge } e} w_e^a \\ &\ge \frac{(\sum_{\text{edge } e} w_e)^a}{|E|^{a-1}} = \frac{vol(G)^a}{|E|^{a-1}} \end{aligned} \tag{4}$$

$$\begin{aligned} \tilde{\beta} &\ge \frac{\text{vol}(\tilde{G})\tilde{w}_{min}}{\tilde{d}_{\max}^2 |\gamma_{\max}| b} \\ &\ge \frac{vol(G)^a w_{min}^a}{|E|^{a-1} d_{\max}^{2a} |\gamma_{\max}| b} \\ &= \beta \Big(\frac{vol(G) w_{min}}{|E| d_{\max}^2}\Big)^{a-1} \end{aligned} \tag{5}$$

If we can show that

$$C := \frac{vol(G) w_{min}}{|E| d_{\max}^2} \overset{!}{\le} 1 \tag{6}$$

then the bound for spectral gap of MARW will exponentially decrease with a. Indeed,

$$vol(G) w_{min} = w_{min} \sum_e d_e \le w_{min} |E| w_{max} \le |E| d_{max}^2$$

■

This proposition shows that for a MARW with a agents on a weighted graph, a lower bound on the spectral gap will decrease with "the a-th power" of defined C. In particular, as the gap of the standard random walk is always smaller than 1, this means that the bound on the spectral gap of the MARW is a power of a smaller than the gap in the original graph. Even though this does not prove that the spectral gap itself has the same behavior, it is a strong indication that this might be the case. In this case, we can conclude that the "bottlenecks" in the MARW are much harder to overcome than the ones in the original random walk.

Note that if we consider unweighted graphs (that is $w_{ij} \in \{0, 1\}$)), then the special case of the loop-free MARW with $l = 0$ corresponds to the standard random walk. This just shows that in this case, the technique of bounding the spectral gap the way we do it is suboptimal and leads to the trivial result that both gaps are bounded by the same quantity.

In **case where $\mathbf{l > 0}$** but l is still small we expect a similar behavior as for the case $l = 0$: the mixing time will be much smaller than the one for the original random walk. Only when l becomes bigger, this behavior starts to change. On the other end, when l tends to ∞, the effect is turned around: the random walk mixes much faster than the original random walk, see [1] and next section.

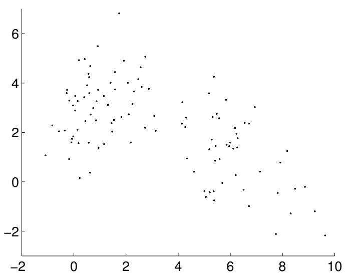
IV. Analyzing MARWs on ε-neighborhood graphs

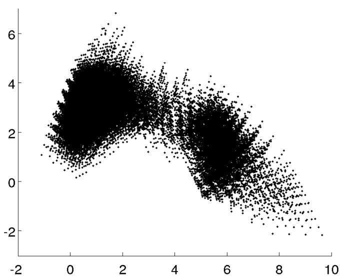
In this section we analyze the case of random geometric graphs, in particular ε-graphs. These graphs are widely used in data mining and machine learning, and the effect of replacing the RW by the MARW can be visualized in a very intuitive manner. Assume we are given a sample of points drawn i.i.d. from some underlying probability distribution with density f on $\mathbb{R}^d$. The ε-graph takes these points as vertices and connects two points by an unweighted edge if the points have distance at most ε from each other. Intuitively, the ε-graph is supposed to show the structure of the underlying probability density f. As an example consider the density in Figure 2d. It consists of two clearly separated high density regions corresponding to two clusters. A sample of points from this density can be seen in Figure 2a and expresses the cluster structure as well. If we now build an ε-graph on this sample (with a reasonable choice of ε, say $\varepsilon = 1$), then this graph consists of two distinct clusters as well: each cluster has many connections inside the cluster and only few connections to the other cluster.

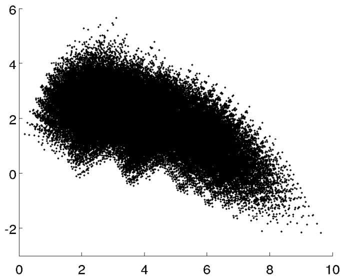
As we are going to explain now, the advantage of the geometric setting is that replacing a standard RW by a MARW can be interpreted as changing the cluster structure of the underlying density. We will see that for reasonably small l, replacing the standard RW by the MARW has a similar effect as replacing a random walk on n points drawn from the original density g by a random walk of $O(\tilde{n})$ points drawn from density g^a. Depending on the structure of graph, $\tilde{n}$ can vary from n to n^a. In particular, the peaks of g^a will be much sharper than the ones of g, leading to a much lower probability of transitioning between different modes of the underlying distribution.

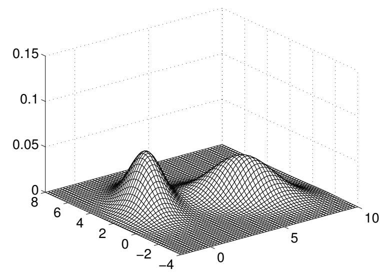
For the ease of presentation we start with the **case $\mathbf{a = 2}$ and $\mathbf{l = \varepsilon}$**. Here we define the distance between two vertices in the metric of the underlying space (this corresponds to $l = 1$ if the distance between two vertices in the graph is measured by the shortest path distance). The general case will be treated later.

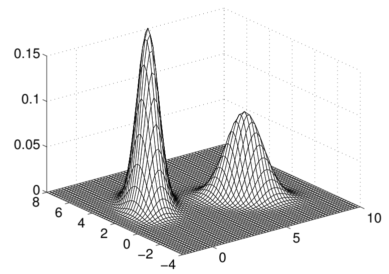
Let us describe the MARW-graph G' for the special case. G' will contain two types of vertices: vertices corresponding to the situation that the two agents sit on the same vertex of G, and vertices corresponding to the situation that the two agents sit at the two ends of an edge of G. We denote the set of first type vertices by $U_1 = \{u_{x,x} \mid x \in V\}$ and the second type of vertices by $U_2 = \{u_{x,y} \mid e_{x,y} \in E\}$. Two vertices u_{x_1,y_1} and u_{x_2,y_2} are connected by a directed edge in G' whenever it is possible to move the two agents from $\{x_1, y_1\}$ to $\{x_2, y_2\}$ in one step in the graph G. The weight of an edge in G' will be defined as the probability


(a) Original sample from density f. Shows moderate cluster structure.


(b) Vertices of the MARW graph for small l. Shows high cluster structure ($a = 3$ and $l = 3$).


(c) Vertices of the MARW graph for large l. No cluster structure any more ($a = 3$ and $l = 7$).


(d) Density f


(e) Density f' for small l ($a = 3$ and $l = 3$).

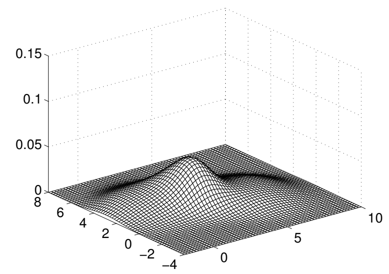

(f) Density f' for large l ($a = 3$ and $l = 7$). By choosing a large l the density smoothes heavily.

Figure 2: Density function and samples from a mixture of gaussians and the corresponding MARW's.

that the MARW performs the corresponding step on G. Note that we define G' without self-loops. As the original graph G is unweighted, it is easy to see that the weights of all outgoing edges of a vertex $u_{x,y}$ are identical. We can represent G' as a geometric graph by identifying each vertex $u_{x,y}$ in G' with a point in $\mathbb{R}^d$, namely with the center of mass of x and y. For an example consider Figure 2. We sampled 110 random points from a mixture of two Gaussians with different covariances in 2 dimensions and built an unweighted kNN-graph with $k = 15$ on this sample. The sample itself is shown in Figure 2a. Now consider a MARW with $a = 3$ and $l = 1$ on G (and shortest path as distance function). In Figure 2b we show the vertices of the corresponding MARW-graph G' (where we represented the vertices $u_{x,y,z}$ by the center of mass of x, y, z).

We now want to compare the geometric properties of the two graphs G and G'. Consider the ball B of radius $\varepsilon/2$ centered at a particular point $x \in G$. Denote the number of sample points in B by k. By definition, all sample points in B have distance at most ε to each other and thus are connected in G. The graph G' will have k type-1 vertices corresponding to the points in B, and $k(k-1)/2$ type-2 vertices corresponding to pairs of points in B. This shows that the number of vertices in B increases from $O(k)$ in G to $O(k^2)$ in G'.

Now recall that our sample points have been sampled from an underlying density f. Assume that ε is small enough such that the density on an ε-ball can be considered approximately constant. Consider two ε-balls B_1 and B_2 centered x_1, x_2. By the definition of a density, the numbers of sample points in B_1 and B_2 are roughly related by

$$\frac{\text{\# vertices of } G \text{ in } B_1}{\text{\# vertices of } G \text{ in } B_2} \approx \frac{f(x_1)}{f(x_2)}$$

What would be a density f' corresponding to the vertices in G'? As we have seen above, when k is the number of vertices of G inside B, then the number of vertices of G' inside B is of the order k^2. Hence we deduce

$$\frac{f'(x_1)}{f'(x_2)} \approx \frac{\text{\# vertices of } G' \text{ in } B_1}{\text{\# vertices of } G' \text{ in } B_2} \approx \frac{f(x_1)^2}{f(x_2)^2}$$

Consequently, G' can be interpreted as a geometric graph based on the underlying density $f' = f^2$. Squaring a density leads to much sharper peaks in the modes of the distribution and deeper valleys between the modes, that is the cluster structure is much better expressed. For an illustration compare the densities in Figures 2d (original density) and 2e (density corresponding to a MARW with small rope length). It is clear that a random walk on a sample of f' (corresponding to the MARW on G) has a much harder time than the original RW on a sample of f (corresponding to the RW on G) to transition from one cluster to the other one. Additionally, the number of vertices in G' corresponds to $|V| + |E|$, that is, it is of the order $O(|E|)$ rather than of the order $O(|V|)$ as in the original graph G. This effect can be seen in Figures 2a (vertices of G) and 2b (vertices of G').

In **case of general** $a > 1$ **and rope length** $l = \varepsilon$ a similar argument shows that the distribution of vertices in G' is raised to the power of a: denoting the number of cliques of size t by V_t, the number of vertices in G' is $\sum_{i=1}^{a} \binom{a-1}{i-1} |V_i|$. To see this, note that the number of placements for a agents on i vertices with at least one agent at each vertex is $\binom{a-1}{i-1}$. Depending on the exact numbers $|V_i|$, the number of vertices in G' can vary between $O(n)$ to $O(n^a)$.

The above argument still holds **if the rope length** l **is larger than** ε. In this case, however, we can no longer argue about the densities f and f' themselves, but rather about their smoothed counterparts obtained by convoluting f and f' with the function that is constant on an l-ball. Denote the characteristic function of a ball with radius l and centered at x_0 by $\chi_{B(x_0,l)}$:

$$\chi_{B(x_0,l)}(x) = \begin{cases} 0 & \text{if } d(x, x_0) > l \\ 1 & \text{otherwise.} \end{cases}$$

Then the density of G' satisfies

$$\frac{f'(x_1)}{f'(x_2)} \approx \frac{(\int f(t)\chi_{B(x_1-t,l)}dt)^a}{(\int f(t)\chi_{B(x_2-t,l)}dt)^a}.$$

That means G' can be interpreted as a geometric graph based on the underlying distribution

$$f'(x) \propto (f(x) * \chi_{B(x,l)})^a \tag{7}$$

where $*$ is the convolution operator. For small l these convolutions will be reasonably close to the original density functions, and lead to the same intuition as above. However, as soon as l gets very large (for example, l is as large as the distance between different modes of the underlying distribution), then the interpretation changes completely. In this case, the density f' is smoothed so heavily that the different modes are not visible any more and the distribution becomes unimodal in the extreme case. Then, the random walk on G' mixes very fast as there are no bottlenecks any more, and transitions easily between points corresponding to different clusters of the original density f. In this case, we completely changed the behavior of the MARW: instead of mixing slower than the original RW it now mixes much faster (and this is the case considered in the paper [1]).

This effect is demonstrated in Figure 2. Plot 2f shows the smoothed density f'. As we can see, the clusters have vanished completely and are replaced by a unimodal distribution. The same effect can be seen for the points of the MARW-graph G' in Figure 2c.

These insights can now be used to understand **the choice of** l. There is a trade-off between two goals. The rope length has to be considerably smaller than the diameter of the cluster and the distance between different clusters one wants to find (otherwise the smoothed distributions loose their cluster properties as in Figure 2f). On the other hand, it should be as large as possible in order to allow G' to have many vertices (otherwise G' will look almost as G, and the effect of sharpening the peaks by squaring the distribution will not occur).

While the above argument was for ε-graphs, one can observe similar effect on other types of random geometric graphs such as kNN graphs or Gaussian neighborhood graphs. By a similar analysis one can also discuss the behavior of a **MARW on a general, unweighted non-metric graph**. Even though we do not have a geometric interpretation in terms of an underlying distribution, the effect of the rope length and the trade-off between the different aims when choosing a good rope length is the same as described above.

V. Experiments

Here we demonstrate different aspects of the MARW in simulations and experiments and compare it to other algorithms from the literature. By using expensive partitioning algorithms, the authors of [11] found that even in massive graphs the typical clusters have a relatively small size. This shows that in real world datasets one should look for clusters in the range of hundreds to thousands of nodes. To validate local clustering algorithms it is thus enough to study graphs up to a couple of thousand vertices, which is what we do in our experiments.

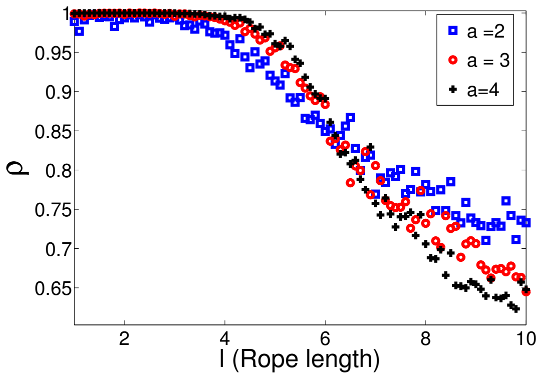
Whenever we know the true cluster labels in a data set, we measure the performance of a local clustering algorithm by the following procedure. We call the starting vertex x_s, the true label of vertex x by $y(x)$ and the set of visited vertices by S. Then we define r as the ratio

$$r = \frac{|\{x \mid y(x) = y(x_s)\}|}{|S|}$$

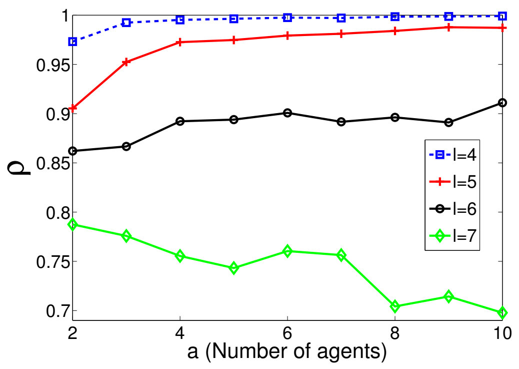
which is the fraction of the visited points belonging to the same cluster as x_s.

A. Effect of rope length and number of agents in a toy example

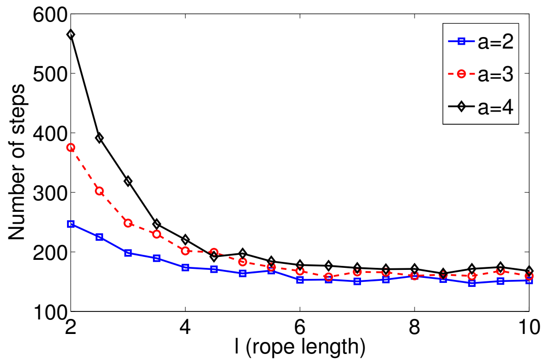
We sample 200 points from a mixture of Gaussians (similar to Figure 2a) in $\mathbb{R}^2$ and build the kNN graph for $k = 10$. We run the MARW with $a = [2, 3, 4]$ for rope lengths in the interval $[1, 10]$. We allow the MARW to take 200 steps and record the set of visited points.



(a)



(b)



(c)

Figure 3: Influence of rope length l and number a of agents on the MARW

In our first experiment we fix the number of agents and investigate the effect of the rope length. We say that a MARW is accepted if $r \geq 0.9$. We repeat this procedure 100 times with random starting vertices. The performance measure ρ is defined by:

$$\rho = \frac{\text{Number of accepted } MARW\text{s}}{\text{Number of } MARW\text{s}}$$

The results are shown in the Figure 3a. We can observe very good performance of MARW for $l \approx [1, 4]$. In this regime, nearly all MARWs get accepted. The performance drops by increasing l. Note that in this data set, the width of the bottleneck region is about 3 (cf. Figure 2a). This corroborates our rules of thumb stated in Section 4: as soon as l gets considerably wider than the bottleneck, the random walk mixes too fast, hence many MARWs get rejected in our experiment.

As a baseline we also used a simple RW on the same data. It leads to performance $\rho = 55\%$., which is way worse than the MARW results. Additionally we observed that the variance of the value ρ (measured over different starting points) is much higher for RW than for the MARW. This shows that MARWs not only give better performance, but also lead to a more stable behavior.

Next we investigate the effect of the number of agents on the same graph by running the MARW for $a \in [2, 10]$ and $l \in \{4, 5, 6, 7\}$. The results are shown in Figure 3b. When l is still small, by increasing a we observe an increase of the performance ρ. That is, in this regime more agents lead to more accepted MARWs. As soon as the rope length gets too long ($l = 7$) the performance decreases with the number of agents. This is the regime where the MARW mixes faster than the orignial RW. All in all, this experiment shows that the rope length l governs whether the cluster structure of the MARW is more or less expressed than in the original density. The number of agents acts as an amplifier.

Note that while increasing the number of agents amplifies the performance of the MARW, it also has a negative side effect. The larger a, the slower the MARW moves. This has two effects. First, when we implement the MARW, much more proposal steps get rejected for violating the rope length condition (leading to an increase in running time). Second, the tendency of staying in a high density region might be so pronounced in a MARW with large a that it barely moves around but often visits vertices it has already seen. This effect is demonstrated in Figure 3c. We run MARW with

Table I: Percentage of correctly visited vertices for different real world datasets. N is the number of vertices, E is the number of edges, D is the dimensionality of data and K is the number of clusters. The number in parentheses shows the cluster size we used as stopping criterion

Dataset	N	E	D	K	Nibble	Apr.PageRank	MARW $a = 3$	MARW $a = 4$
Wine (40)	178	1268	13	2	82.31	86.8	88.02	91.76
Breast Cancer(160)	683	10022	9	2	93.26	94.66	94.37	96.02
USPS (50)	7291	106722	256	10	88.5	92.78	93.37	97.43
USPS (100)	7291	106722	256	10	81.48	88.15	90.85	92.21

different number of agents and rope lengths until they visit 80 different nodes. The number of steps during the visit shows that by increasing the number of agents, they tend to stay in local neighborhoods. This effect vanishes when we increase the rope length. When the rope length gets long enough (in this experiment, around 6), agents move with more freedom and less tendency to stay in the same place.

B. Real world examples

We now compare performance of various algorithms on some real world datasets: our MARW, the *Nibble* algorithm from [14] and *Apr.PageRank* from [2]. Note that all algorithms have different stopping criteria. In order to make them comparable, we simply run all algorithms from the same starting point until they discovered a given number of vertices. Then we analyze the quality of the clusters that are discovered by the algorithms. We set the parameters in the algorithms as follows. In *Nibble* we optimize the threshold parameter by approximately finding smallest threshold that allows us to discover the mentioned number of vertices. For approximate PageRank, using the value of α suggested by the authors led to poor results. So instead of using their values we report the best result obtained for a range of parameters α.

We use the *Wine* and *Breast Cancer* datasets from UCI repository, the USPS handwritten digit recognition dataset [9] and a collaboration network of arxiv papers [10].

For the first three data sets the true cluster labels are known. For these data sets we first constructed an unweighted kNN graph ($k = 10$). We started the algorithm from a random starting point. We ran each algorithm until visiting a given number of points: approximately $\frac{2}{3} \times$ *size of smallest cluster* vertices in Wine and Breast Cancer datasets, and 50 resp. 100 points in the USPS data set. Then we used r, the percentage of visited points from the correct cluster, as our performance measure. We repeated the experiments for 100 random starting points. For our algorithm we simply set the rope length to $l = 1$ (we did not optimize over l or a in any way). For the other algorithms we chose the parameters as described above. The results for $a = 3, 4$ are shown in Table I. We can see that our algorithm achieves the best results on all data sets.

It worth to note that even though using more agents improves the performance, it also increases the running time of the algorithm. There is a trade off between performance versus running time and visiting majority of points in the cluster. MARW with big a will move slower and will mainly remain in the center of a cluster. This effect is demonstrated in Table II. In this table we fix the number of *different* vertices we want to visit. We then report how many steps the random walk had to perform to achieve this (a step is simply one "hop" of the random walk). For example, in the Wine data set we can see that in order to visit 40 different vertices, the MARW with $a = 3$ had to take about 500 steps while the MARW with $a = 4$ already needed about 1100 steps. This shows that for larger a, the random walk tends to visit the same vertices more often and is more reluctant to move away from high-density regions.

The *collaboration network* dataset consists of an undirected non-metric graph with about 12000 vertices. As ground truth is not known on this data set, we use the conductance of the set of visited vertices as a performance measure. The conductance of a cluster is defined as the ratio of the number of its external connections to the number of its total connections. Formally, the conductance of a subgraph S from graph $G = \{V, E\}$ is defined as:

$$\Phi(S) = \frac{\sum_{i \in S, j \in V-S} w_{ij}}{\min(\mathrm{vol}(S), \mathrm{vol}(V-S))} \tag{8}$$

$$\mathrm{vol}(A) = \sum_{i \in A, j \in V} w_{ij} \tag{9}$$

The conductance is a popular quantity to measure the quality of a cluster in a graph. The aim is to visit a set with small conductance.

We run two rounds of experiments, one aiming for 200 and one for 500 vertices. For the MARW we use the shortest path distance as underlying distance function and rope lengths $l = 1$ (for the 200 vertices) and $l = 2$ (for the 500 vertices). We set the number of agents to 3. The results are shown in Table III. In this example by visiting

Table II: We fixed the number of *different* vertices we wanted to visit (reported in parentheses). The numbers in the table show how many *steps* the MARW needed to make in order to visit this number of different vertices, on average over different starting points. The rope length l was fixed to $l = 1$.

Number of agents	Wine (40)	Breast Cancer (100)	USPS (50)	USPS (100)
$a = 3$	487	2417	156	361
$a = 4$	1142	19798	281	752

Table III: Conductance of the set of visited points in the collaboration dataset. N is the number of vertices and E is the number of edges.

Dataset	N	E	Nibble	Apr.PageRank	MARW $a = 3$
Collaboration (200), l=1	12008	237010	0.5068	0.4916	0.4342
Collaboration (500), l=2	12008	237010	0.4703	0.4565	0.194

500 vertices, MARW works significantly better than other algorithms and the average conductance is less than half of conductance reached in other algorithms.

VI. Discussion

In this paper we introduce the new concept of the multi-agent random walk. We provide several ways to analyze the behavior of the MARW. The bottom line of these analyses is always the same: using the MARW instead of the standard RW enhances the cluster properties of the graph. Hence, the MARW is well-suited for local clustering. Our experiments show that it outperforms the major other approaches from the literature.

The concept of a MARW is new, and we believe that it has many advantages and can be applied to other data mining problems. For example, one might be able to derive a spectral clustering algorithm based on the MARW-Laplacian instead of the standard random walk Laplacian, and similar ideas might work for semi-supervised learning. Essentially, any algorithm that works with random walks or related matrices might be adapted to the MARW. It will be a matter of future work to fully explore the possibilities of multi-agent random walks in data mining and machine learning.

References

[1] N. Alon, C. Avin, M. Koucky, G. Kozma, Z. Lotker, and M. Tuttle. Many random walks are faster than one. In *SPAA '08: Proceedings of the twentieth annual symposium on Parallelism in algorithms and architectures*, pages 119–128, 2008.

[2] R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *FOCS*, pages 475–486, 2006.

[3] R. Andersen and Y. Peres. Finding sparse cuts locally using evolving sets. In *STOC*, pages 235–244, 2009.

[4] P. Diaconis and D. Stroock. Geometric bounds for eigenvalues of Markov chains. *The Annals of Applied Probability*, pages 36–61, 1991.

[5] C. Ding, X. He, H. Zha, M. Gu, and H. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *ICDM*, pages 107–114, 2001.

[6] C. Gallesco, S. Müller, and S. Popov. A note on spider walks. *ArXiv e-prints*, October 2009.

[7] C. Gallesco, S. Müller, S. Popov, and M. Vachkovskaia. Spiders in random environment. *ArXiv e-prints*, January 2010.

[8] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1768–1783, 2006.

[9] J. J. Hull. A database for handwritten text recognition research. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(5):550–554, 1994.

[10] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*, 1(1):2, 2007.

[11] J. Leskovec, K. Lang, A Dasgupta, and M. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 695–704, 2008.

[12] J. Pan, H. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-modal correlation discovery. In *KDD*, pages 653–658, 2004.

[13] J. Ruan and W. Zhang. An efficient spectral algorithm for network community discovery and its applications to biological and social networks. In *ICDM*, pages 643–648, 2007.

[14] D. Spielman and S. Teng. A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning. *CoRR*, abs/0809.3232, 2008.