VoroNet: A scalable object network based on Voronoi tessellations*

Olivier Beaumont[1], Anne-Marie Kermarrec[2], Loris Marchal[3], Étienne Rivière[4]

[1]LaBRI/INRIA
Bordeaux, France
obeaumon@labri.fr

[2]IRISA/INRIA
Rennes, France
akermarr@irisa.fr

[3]ENS Lyon
Lyon, France
loris.marchal@ens-lyon.fr

[4]IRISA/Université de Rennes 1
Rennes, France
eriviere@irisa.fr

Abstract

In this paper, we propose the design of VoroNet, an object-based peer to peer overlay network relying on Voronoi tessellations, along with its theoretical analysis and experimental evaluation. VoroNet differs from previous overlay networks in that peers are application objects themselves and get identifiers reflecting the semantics of the application instead of relying on hashing functions. This enables a scalable support for efficient search in large collections of data. In VoroNet, objects are organized in an attribute space according to a Voronoi diagram. VoroNet is inspired from the Kleinberg's small-world model where each peer gets connected to close neighbours and maintains an additional pointer to a long-range neighbour. VoroNet improves upon the original proposal as it deals with general object topologies and therefore copes with skewed data distributions. We show that VoroNet can be built and maintained in a fully decentralized way. The theoretical analysis of the system proves that routing in VoroNet can be achieved in a poly-logarithmic number of hops in the size of the system. The analysis is fully confirmed by our experimental evaluation by simulation.

1 Introduction

Peer to peer has clearly been recognized as a key communication paradigm to build robust and scalable distributed applications. Searching in large networks is one of the core functionalities offered by peer to peer systems. Among the numerous peer to peer networks that have been proposed in the past 5 years, structured peer to peer networks, such as Pastry [11] or Chord [13], have generated a lot of interest. Most of these overlay networks organize physical nodes in a logical network and provide a scalable support for fully decentralized Distributed Hash Tables (DHT). Such networks heavily rely on the use of hashing functions to ensure load balancing. For example, nodes use hashing functions to get an identifier, so that the identifier space is uniformly populated. Likewise, file identifiers are hashed to produce the key, used to locate objects later on in the DHT. Such networks offer an exact-match interface which make them natural candidates for file systems or archival systems in which requests target well-identified files. While providing efficient key-based lookup, the query mechanism of such systems is often restricted to exact search. More specifically, they are not built to handle range queries on either one or several attributes, mainly due to the hashing mechanism. Either one attribute is associated to a specific node leading to load unbalance for skewed distribution of the attributes, or a node is made responsible for an (attribute, value) pair. This latter choice might require flooding mechanisms or querying the entire set of possible values for that range.

In this paper, we step away from this form of hash-based peer to peer overlay networks and propose the design of VoroNet, an object-based overlay network based on Voronoi tessellations. VoroNet differs from general-purpose peer to peer structured overlays in that it links application objects rather than physical nodes so that objects with similar characteristics are neighbours in the object to object network, providing a natural support for range

* This work was supported in part by French ANR "Masse de données" ALPAGE project.

queries. In addition, VoroNet does not rely on hashing functions to distribute the load or to evenly populate the identifier space. Each object is managed by the physical node hosting it, and each physical node is responsible only for the objects it has published in the overlay. For the sake of clarity, we will concentrate in this paper on the case where each physical node is responsible for exactly one object. Note that all the results presented here still hold true in the more general case (actually, the performance would even be better in practice, since holding several objects at the same node would create extra shortcuts in the overlay). VoroNet specifies a d-dimensional attribute space in which object's virtual coordinates (specifying its identifier) are the attribute values associated to the objects themselves. The attribute space is mapped on a d-dimensional torus. Attributes reflect the application naming space: each object is represented by its attributes values in this space. This may lead to skewed data distribution in the space and yet VoroNet ensures an efficient routing. In this paper, we focus on the special case where $d = 2$, the object space being divided as a Voronoi tessellation.

Inspired by the small-world model proposed by Kleinberg [9], it goes beyond the original proposal by generalizing the approach, initially proposed in the context of a grid-based system. In the seminal paper of Kleinberg [9], each node of a grid mesh knows its four neighbours in the grid as well as an additional remote node carefully chosen according to its distance in the grid. This paper provides clear theoretical bounds on routing performance or node degree distribution as long as the aforementioned assumptions hold. This model is too restricted to be directly transposed to any realistic distribution of objects among the attribute space and any kind of graphs or overlay structure. Instead, in VoroNet, we propose to use a similar design where each object neighbour set, formed by its Voronoi neighbours, is enhanced with a long range contact chosen according to a precise distribution of long range link lengths. This paper provides similar theoretical bounds on the routing performance for any distribution of data among the attribute space.

Contributions The contribution of this paper is twofold. First we propose the design of a fully decentralized peer to peer object-based overlay network, relying on Voronoi tessellations, along with a theoretical analysis and evaluation by simulation. VoroNet is innovative as it links objects rather than peers, enabling the naming space to reflect the application itself and therefore simplifying the range-based search. In addition, VoroNet uses the properties of Voronoi tessellations to handle skewed distribution of data. We propose fully distributed join and leave algorithms, requiring only each object to have a (very) limited knowledge of the system. These algorithms are fully distributed, resilient to calculation degeneracy and evenly distribute the load between peers. In addition they are efficient with respect to network traffic as well. Second, we provide a generalization of Kleinberg algorithm and show that O($log^2|\mathcal{O}|$) routing performance bounds can be achieved, $|\mathcal{O}|$ being the number of objects in the network. Simulation results confirm the theoretical analysis. Note that the specification of associated query mechanisms is out of the scope of this paper.

Roadmap The rest of this paper is organized as follows, in Section 2 we provide some background both on the Kleinberg's model on which our work is largely inspired as well as Voronoi tessellations. Section 3 presents VoroNet in a nutshell and its associated algorithms. The analysis of VoroNet is presented in Section 4. Section 5 provides simulation results of VoroNet. Before concluding and presenting some perspectives of this work, we survey related works.

2 Background

In this section, we provide some background on the Kleinberg's model and the Voronoi diagrams, both on which VoroNet is based.

2.1 Kleinberg's model

Kleinberg proposed a small-world graph model [8, 9] that provides both poly-logarithmic paths between any two vertices (*small-world* property) and the *navigability* property: greedy decentralized path discovery algorithms can find poly-logarithmic paths in the graph between any couple of nodes. The model is a $n \times n$ grid where every vertex has edges to its four direct neighbours and k (typically one) long-range neighbour(s). This long-range neighbour is chosen with a probability proportional to $\frac{1}{d^s}$, where d is the link length, *i.e.* the Euclidean distance between the vertex and its remote neighbour. Figure 1(a) depicts an example of such a network. Only a subset of the long range links are drawn, for the sake of clarity. In [9], the author shows that $s = 2$ enables both *small-world* and *navigability* properties for the 2-dimension grid. A generalization to d-dimensional spaces [4] has shown that, for any d, choosing $s = d$ allows discovered paths of size in $\theta(\log^2 \frac{n}{k})$ using greedy algorithms. VoroNet proposal in this paper extends these results to more general topologies than grids.

2.2 Voronoi Diagrams

A Voronoi diagram [3, 6] is a partition of space $\mathcal{V}(P)$ associated to a given finite set of points $P = \{p_1, \dots, p_n\}$ and a distance measure d. If $d(p_1, p_2)$ denotes the Euclidean distance between p_1 and p_2, each point p_i is associated with a *Voronoi region* $R(p_i) = \{p | d(p, p_i) < d(p, p_j), \forall j \neq i\}$. The partition of the space $\{R(p_1), \dots, R(p_n)\}$ is the Voronoi diagram of P. The boundary between two Voronoi regions is a *Voronoi edge*, and a point where three or more Voronoi regions meet is a *Voronoi vertex*.

The dual of the Voronoi diagram $\mathcal{V}(P)$ is the *Delaunay triangulation* $\mathcal{D}(P)$. Let $\mathcal{C}(\Delta_{p_1p_2p_3})$ be the circumscribed circle of the triangle formed by points p_1, p_2 and p_3. The Delaunay triangulation is the set of triangles $\{\Delta_{p_ip_jp_k}\}$ such as $\Delta_{p_ip_jp_k} \in \mathcal{D}(P) \Leftrightarrow \forall p_l \in P - \{p_i, p_j, p_k\} : p_l \notin \mathcal{C}(\Delta_{p_ip_jp_k})$.

The relation between $\mathcal{V}(P)$ and $\mathcal{D}(P)$ is straightforward: there exists a Voronoi edge $R(p_i) \leftrightarrow R(p_j) \in \mathcal{V}(P)$ if and only if there is a link between p_i and p_j in $\mathcal{D}(P)$. Figure 1(b) depicts an example of a Voronoi diagram for a set of points (bold lines) and the associated Delaunay triangulation (dashed lines).

3 VoroNet in a nutshell

We consider a set of objects $\mathcal{O}$. As already noted, for the sake of clarity, we consider a one-to-one mapping between an object and a physical node. The overlay design space is a d-dimensional torus, each dimension representing one attribute. The coordinates of an object in this space are uniquely specified by its values, one for each attribute. In this paper, we focus on $d = 2$ on the unit torus (where attribute values lie in $[0 \ldots 1] \times [0 \ldots 1]$). We discuss higher dimensions in the perspectives. In addition, a parameter $|\mathcal{O}|_{target}$, known by all objects, indicates for which number of objects the system is optimized (that is, for which the better possible routing is guaranteed).

Each object maintains its *view* of the system, *i.e* links to a set of other objects. Each entry of the view is composed of the IP address of the node hosting the object as well as its coordinates in the unit torus. The size of this set of neighbours is of order O(1) for any reasonable distribution of objects in the attribute space. Objects in VoroNet are organized according to the Voronoi diagram $\mathcal{V}(\mathcal{O})$. The set of neighbours is composed of: (i) **Voronoi neighbours**, which are the objects whose regions share a Voronoi edge with that object's region ; (ii) **Long range neighbour** is an additional remote neighbour providing to the network its small-world characteristics (low diameter and navigability). The way this latter neighbour is chosen in VoroNet is inspired by the method described in [8]: the algorithm for choosing this remote neighbour is depicted further in the paper ; (iii) **Close neighbours** is a small set of neighbours within a very short distance of the object, needed to ensure routing properties, and depicted further in the paper. These set of neighbours represents the view of the system on each object.

Routing in VoroNet is used both for object insertion and message forwarding. We use a simple greedy and fully deterministic algorithm to route a message from a source to a destination. Note that this algorithm is not sensitive to skewed distribution of objects in the space. We prove that the theoretical number of hops between any two objects is in O($log^2|\mathcal{O}|_{target}$), where $|\mathcal{O}|_{target}$ denotes the target number of objects in the overlay.

Joining and leaving the overlay requires to recompute the Voronoi tessellation for a set of objects. This modifies both the Voronoi and long range neighbours. We provide fully distributed algorithms to achieve this: the closest node to the object (being added or deleted) is in charge of recomputing the new partial tessellation. Then, it sends the new diagrams cells to its neighbourhood, so that they can also update their view. The number of associated communications and computations are also of order O(1).

3.1 Object's view management

As previously mentioned, each object in a VoroNet overlay maintains two main sets of neighbours as in Kleinberg's model. In addition, a small set of additional close neighbours is required at each object to ensure routing efficient termination.

The basic structure of VoroNet is a Voronoi diagram. Each object $o \in \mathcal{O}$ has a set of Voronoi neighbours $\{\text{VN}(o)\}$, which are the objects whose Voronoi region share a Voronoi vertex with o's Voronoi region. This set is maintained locally when inserting and deleting objects to the overlay: each $\{\text{VN}(o)\}$ is modified so that the structure formed by $\{\text{VN}(o)\}$ neighbourhood is exactly the Delaunay triangulation of $\mathcal{O}$. Second, to ensure efficient routing, each object o maintains one long range neighbour $\text{LRN}(o)$. The choice of long range range neighbours is drawn from a generalization of Kleinberg's model to Voronoi tessellations in two dimensional spaces. Finally, each object o needs to maintain a set of close neighbours $\{\text{CN}(o)\}$. These neighbours are in a disk of radius $d_{\min} = \frac{1}{\pi \times |\mathcal{O}|_{target}}$ centered at o: for each close neighbour $o' \in \{\text{CN}(o)\}$, $d(o, o') \leq d_{\min}$. The radius $d_{\min}$ is very small compared to the attribute space size, but depends on the target number of objects in the overlay. We discuss in the perspectives an approach to dynamically adapt this target number of objects. These neighbours are mandatory to ensure a polylogarithmic routing cost in spite of irregularities in the object distribution, because long links lengths are chosen in the range $[d_{\min} : \frac{\sqrt{2}}{2}]$.

These three types of neighbours are illustrated in Figure 2.

Links between Voronoi neighbours and close neighbours are of symmetric nature. Long range neighbours are chosen at a given node and are, by definition, asymmetric. This may be a problem if t, the chosen long range neighbour $\text{LRN}(o)$ leaves the overlay: t will no more be able to contact o to make it choose a new long range neighbour. To overcome this difficulty, we record o as a specific neighbour of t, and will mention it as the "Back Long Range" neighbour, denoted by $\text{BLRN}(o)$. This extra neighbour is nevertheless

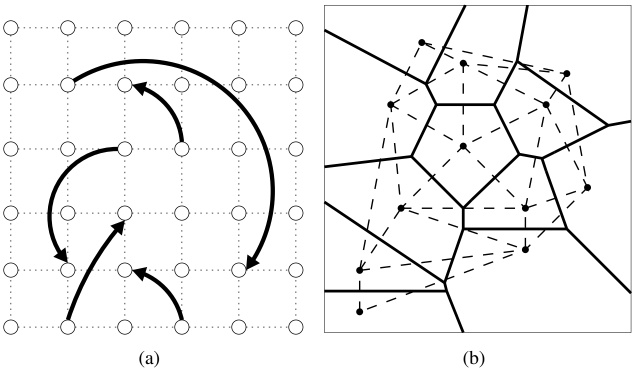


Figure 1. (a) Kleinberg network with some long range links ; (b) Voronoi Diagram in the unit cube.

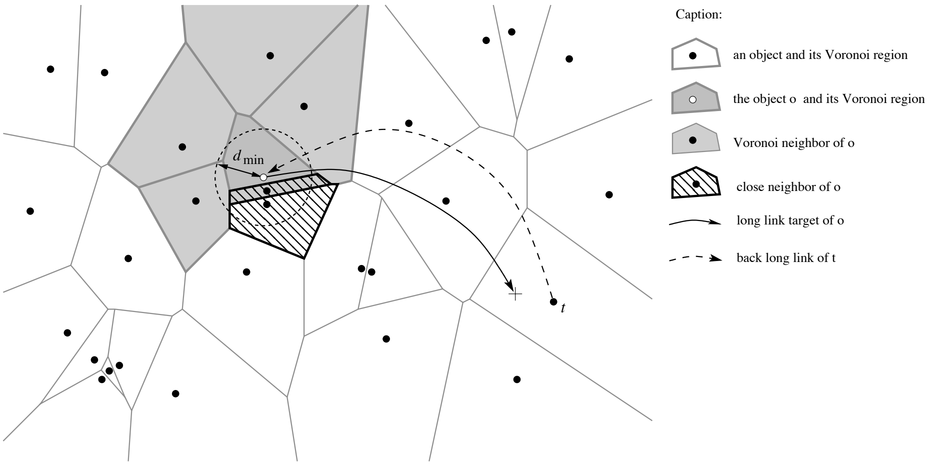


Figure 2. Example of neighbourhood in VoroNet

not used for routing convergence analysis. Obviously, the size of data structures stored at each object o depends on the distribution of objects in the unit torus. However, under reasonable assumptions on the distribution of objects, the size of the data structures stored at each node is of order $O(1)$. Due to lack of space, we refer the interested reader to the companion report [5] for a detailed proof. Roughly, since Voronoi tessellations are planar graphs of unit torus of genus 1, the average number of Voronoi neighbours is bounded by 6. Moreover, the distance $d_{\min}$ is chosen so that for uniform distributions, the average number of nodes in a circle of radius $d_{\min}$, and therefore the number of close neighbours, is 1. At last, each node has exactly 1 long range neighbour, and therefore, for uniform distributions, each node is the long range target of 1 other node. Therefore, for uniform distributions of objects on the torus, the expected number of neighbours is of order $O(1)$.

3.2 Routing in VoroNet: small paths and navigability

The key property of VoroNet is to achieve efficient routing (in terms of number of hops) using a very simple greedy routing algorithm. It works as follows: when an object o receives a message m aimed at a target point P, m is forwarded to the neighbour n of o that minimizes the Euclidean distance between n and P, among the neighbours $\{\text{CN}(o)\}$, $\{\text{VN}(o)\}$ and $\text{LRN}(o)$. If P corresponds to an object, the object is found. If P does not correspond to an object, the routing finished in the Voronoi region where P lies, and the object responsible for this Voronoi region is found (nearest neighbour query). The routing algorithm is formally described in Section 4. In both cases, the number of hops required to route the message is poly-logarithmic in $|\mathcal{O}|$, the overall number of objects in the overlay.

3.2.1 Object insertion

We now introduce the mechanism for inserting a new object in the overlay. Details, proofs of correctness and complexity results are provided in companion report [5]. Suppose a new object wants to join the overlay. Its attributes fully determine the point o where it will be located in the attribute space. We assume that it knows some object x. Starting from this object x, the following operations are performed:

1. The simple greedy routing algorithm is applied to find the object o' satisfying $o \in \mathcal{R}(o')$.

2. o' is responsible for computing the new region associated to o and its neighbourhood. To add o to the current local Voronoi diagram, given the Voronoi region that contains this point in the non-updated diagram, the algorithm proposed by Sugihara and Iri [14] is used. It uses local exploration methods based on combinatorial decisions, and has the strong advantage that it permits reconstruction of local Voronoi regions that are topologically consistent, even if calculation degeneracy takes place.

3. Then o' determines $\{\text{CN}(o)\}$: each new neighbour y of o in the updated Voronoi diagram sends to o' the set of its neighbours z (either in $\{\text{CN}(y)\}$ or in $\{\text{VN}(y)\}$) whose distance to o satisfies $d(o,z) \leq d_{\min}$. o' then declares o as close neighbour to all nodes z such that $z \in \{\text{CN}(o)\}$. Lemma 1 proves that with this method, o gets all its close neighbours: no object located at a distance lower than $d_{\min}$ is ignored.

4. In order to determine the objects in $\text{BLRN}(o)$, each neighbour y (in the updated Voronoi diagram) sends (and removes them from its own list) the set of its neighbours z (in $\text{BLRN}(y)$) such that $d(o, \text{LRT}(z)) < d(y, \text{LRT}(z))$ (*i.e.* the object o is nearer to the long range link destination point than the object y is).

5. Last, o has to choose a target, *i.e.* a point in the unit torus and to determine $\text{LRN}(o)$, the object that is at that time the nearest from that target point.

Lemma 1. *Suppose that a new object p has joined the overlay, and that it has just determined its Voronoi neighbours, namely its neighbours in the Voronoi diagram. Then all the close neighbours of p are either some of its Voronoi neighbours, or some of the close neighbours of its Voronoi neighbours.*

Proof. For the sake of convenience, in this proof, we will call $cell(o)$ the Voronoi cell of object o in the current Voronoi diagram (containing the object x).

We call A the set of Voronoi neighbours of x, plus x. We want to prove that $\forall y$, such that $d(x,y) \leq d_{\min}$, there exists an $z \in A$ with $d(z,y) \leq d_{\min}$

We consider two cases:

- Case 1: y is a a Voronoi neighbour of x. Then $z = x$ satisfies the proposition.

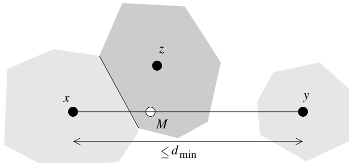


Figure 3. Computing the close neighbours

- Case 2: y is not a Voronoi neighbour of x. We consider the segment $[x,y]$, as illustrated in Figure 3. The parts of this segment belonging to the Voronoi cells of x and y are not contiguous. We consider the object z responsible for the part of the segment contiguous to the part in $cell(x)$, and a point M (which is not an object) in this part of the segment. By triangular inequality, we have: $d(z,y) \leq d(z,M) + d(M,y)$, but as M is in $cell(z)$, we get: $d(z,M) \leq d(x,M)$ together with the previous inequality: $d(z,y) \leq d(x,M) + d(M,y) \leq d_{\min}$. So y is a close neighbour of z, which is a Voronoi neighbour of x. □

3.2.2 Object removal

Let us now consider the operations involved when an object o leaves the overlay. First, since the Voronoi region of o is removed, o is responsible for computing new neighbouring regions and for informing its neighbourhood. Second, if the object o was in charge of a long range link with target point P, belonging to object x, it determines which object o' among its Voronoi neighbours is now in charge of the point P, and delegates the responsibility of the link to o. Note that object x can be reached thanks to the back-long-range link.

4 VoroNet protocol analysis

In this section, we give some key points for the detailed mechanism of insertion/deletion of objects and for the routing mechanism and their justifications. We will not detail all the operations needed for this, but will focus on the choice of a long range link target and the routing mechanism. All details and proofs can be found in [5].

4.1 Choosing a long range target

When an object joins the overlay, it has to find a long range target. This is achieved using function CHOOSE-LRT, depicted in Algorithm 1. This function is defined by analogy to Kleinberg's work [8].

The following lemma describes the distribution of probability built by the function CHOOSE-LRT.

```
CHOOSE-LRT()
    Choose $a$ with uniform probability in $[\ln(d_{\min}), \ln(\frac{1}{\sqrt{2}})]$
    Choose $\theta$ with uniform probability in $[0, 2\pi]$
    Set $\delta = (e^a \cos(\theta), e^a \sin(\theta))$
    Set LRT $= CurrentObject.coordinates + \delta$ (mod 1 for each attribute)
    return
```

Algorithm 1: Algorithm for finding $\text{LRT}(x)$

Lemma 2. *Using function* CHOOSE-LRT*, the probability that* $\text{LRT}(x)$ *belongs to a small surface* $\mathrm{d}S$ *at distance* d *from* x *is given by* $\frac{\mathrm{d}S}{Kd^2}$*, where* $K = \frac{1}{2\pi \ln(\frac{\pi|\mathcal{O}|_{target}}{\sqrt{2}})}$.

An interesting result can be derived from Lemma 2, which provides a lower bound on the probability to choose LRT (x) in a given disk.

Lemma 3. *The probability for* $\text{LRT}(x)$ *to be chosen in a disk of center* y *and radius* fr*, where* $r = d(x, y)$ *is lower bounded by* $\frac{\pi f^2}{K(1+f)^2}$.

This property is useful when proving the efficiency of the greedy routing algorithm.

4.2 Routing complexity

In this section, we give an insight on the proof for the general routing mechanism. We consider the ROUTE function as the general framework for all routing functions (see [5] for details): this general framework comes into several flavors, for routing a message to an object, for finding the region where a node has to be added or for finding the object responsible for a new long range target. In this function, *Target* stands for the target point (which has a different meaning depending on the action to perform). $\text{DISTANCETOREGION}(x)$ is a function used to compute the distance between a given point x and the Voronoi region of the current object (denoted by *CurrentObject* in the algorithm): if this distance is achieved at point z, then $d(x, \mathit{CurrentObject}) = d(x, z)$ with $z \in R(\mathit{CurrentObject})$, and DISTANCETOREGION outputs z. If x belongs to $R(x)$, then DISTANCETOREGION outputs x. Last, $\text{GREEDYNEIGHBOUR}(x)$ is the closest neighbour (using Euclidean norm) from x among the neighbours of *CurrentObject*: $\{\text{VN}(x)\}$, $\{\text{CN}(x)\}$ and $\text{LRN}(x)$.

The algorithm is correct if it can be proven that once the algorithm stops spawning processes it is possible to add z to the overlay, since it is close enough from *CurrentObject*. Then it is possible to prove that *Target* is close enough from z and can be added to the overlay. This is summarized by the following lemma, formally proved in [5].

```
ROUTE($x$, $Target$)
    $z = \text{DISTANCETOREGION}(Target)$
    if $d(z, Target) > \frac{1}{3}d(Target, CurrentObject)$ and $d(Target, CurrentObject) > d_{\min}$ then
        Spawn the process ROUTE($x$, $Target$) on object returned by GREEDYNEIGHBOUR($Target$)
    else
        add a fictitious object to the overlay at point $z$
        add an object to the overlay at point $Target$
        perform some local computations local computation depending on the operation at $z$
        remove the fictitious object at $z$
        (depending on the action, remove the object at $Target$)
    return
```

Algorithm 2: Framework for routing algorithms starting at x

Lemma 4. *Let us assume that*

$$d(\text{DISTANCETOREGION}(\mathit{Target}), \mathit{Target}) \quad (1)$$
$$\leq d(\mathit{Target}, \mathit{CurrentObject})/3 \quad (2)$$

or

$$d(\mathit{Target}, \mathit{CurrentObject}) \leq d_{\min} \quad (3)$$

Then z and Target can be successively added to the overlay.

We now analyze the maximal number of steps (i.e. of calls to GREEDYNEIGHBOUR) of the algorithm. Lemma 5 asserts that the number of steps is poly-logarithmic in $|\mathcal{O}|_{target}$.

Lemma 5. *The number of calls to* GREEDYNEIGHBOUR *in Algorithm 2 is of order* $O(\ln^2 |\mathcal{O}|_{target})$.

Proof. The proof is directly adapted from the proof proposed by Kleinberg [9] in the case of 2D grids. The main difficulty when analyzing the number of steps needed by Algorithm 2 is that *Target* is not *a priori* an existing object, so that we cannot converge toward *Target*. Nevertheless, we can prove that the number of steps needed to meet the condition $d(z = \text{DISTANCETOREGION}(\mathit{Target}), \mathit{Target}) \leq d(\mathit{Target}, \mathit{CurrentObject})/3$ is of order $O(\ln^2 |\mathcal{O}|_{target})$, and previous lemma asserts that once this condition is satisfied, it is actually possible to add the object *Target*.

Let us consider a step of the algorithm, executed at object *CurrentObject*. Let us denote $d = d(\mathit{CurrentObject}, \mathit{Target})$.

The probability that the long-link target $\text{LRT}(\mathit{CurrentObject})$ belongs to the disk of center *Target* and radius $\frac{d}{6}$ is lower bounded (Lemma 3) by

$$\frac{1}{98 \ln(\frac{\pi|\mathcal{O}|_{target}}{\sqrt{2}})}.$$

Let X denote the total number of calls to GREEDYNEIGHBOUR before reaching an object s such that LRT(s) belongs to the disk of center *Target* and radius $\frac{d}{6}$. The expectation $E(X)$ of X is given by

$$\begin{aligned} E(X) &= \sum_{i=1}^{+\infty} \Pr[X \geq i] && (4) \\ &\leq \sum_{i=1}^{+\infty} \left(1 - \frac{1}{98\ln(\frac{\pi|\mathcal{O}|_{target}}{\sqrt{2}})}\right)^{i-1} && (5) \\ &= 98\ln(\frac{\pi|\mathcal{O}|_{target}}{\sqrt{2}}) && (6) \end{aligned}$$

Let us assume now that we have reached an object *CurrentObject* such that LRT(*CurrentObject*) belongs to the disk of center *Target* and radius $\frac{d}{6}$. We can prove (see details in [5]) that

- either GREEDYNEIGHBOUR(*CurrentObject*) satisfies $d($GREEDYNEIGHBOUR$(\textit{CurrentObject}), \textit{Target})$ $\leq \frac{5}{6}d(\textit{CurrentObject}, \textit{Target})$
- or the following condition is fulfilled $d(z =$ DISTANCETOREGION$(\textit{Target}), \textit{Target})$ $\leq \frac{1}{3}d(\textit{Target}, \textit{CurrentObject})$.

Then, continuing routing from *CurrentObject*, after an expected number of $98\ln(\frac{\pi|\mathcal{O}|_{target}}{\sqrt{2}})$ calls to GREEDYNEIGHBOUR, either Algorithm 2 stops because $d(z =$ DISTANCETOREGION$(\textit{Target}), \textit{Target})$ $\leq \frac{1}{3}d(\textit{Target}, \textit{CurrentObject})$ is satisfied or the distance between *CurrentObject* and *Target* is divided by $\frac{6}{5}$. Let us call a super-step such a sequence of calls to GREEDYNEIGHBOUR.

Since after each super-step, either the algorithm stops or the distance between *CurrentObject* and *Target* has been divided by $\frac{6}{5}$, the number of super-steps is upper bounded by

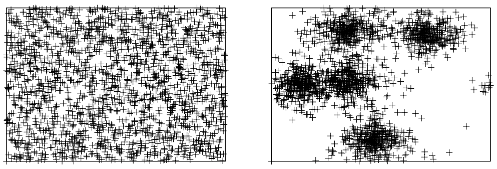
$$\frac{\ln(\frac{\pi|\mathcal{O}|_{target}}{\sqrt{2}})}{\ln(\frac{6}{5})}.$$

Indeed, the initial distance is smaller than $\frac{1}{\sqrt{2}}$ and the algorithm stops as soon as the distance is smaller than $d_{\min}$.

Therefore, by linearity of expectations, the expected number of steps N of the algorithm is given by

$$\begin{aligned} E(N) &\leq \frac{\ln(\frac{\pi|\mathcal{O}|_{target}}{\sqrt{2}})}{\ln(\frac{6}{5})} \times 98\ln(\frac{\pi|\mathcal{O}|_{target}}{\sqrt{2}}) && (7) \\ &\leq \alpha \ln^2(|\mathcal{O}|_{target}) && (8) \end{aligned}$$

for a suitable choice of α, which achieves the proof. □



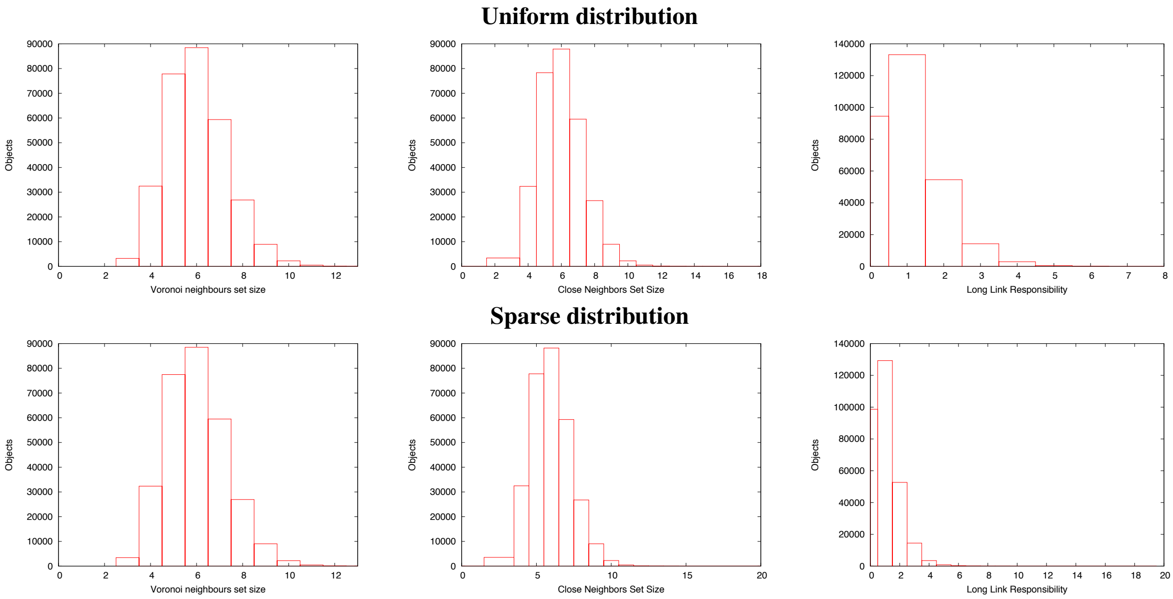
(a) Uniform (b) Sparse

Figure 4. Subset of the two distributions, uniform and sparse with 5 hotspots.

5 Experimental results

In this section, we present the results of experimental studies of VoroNet, obtained through extensive simulations. All simulations were run using $|\mathcal{O}|_{target} = 300.000$. To determine the impact of object distributions in VoroNet, we used two distributions of object values in the unit torus: uniform (equal probability of any point in the square) or sparse. The latter is obtained by picking uniformly at random a point around five random "hot-spots" that represents popular regions of values. Distances of points to the center of a hot-spot follow a powerlaw distribution, where the frequency of the i^{th} most popular value is proportional to $\frac{1}{i^{\alpha}}$, with $\alpha = 5$. Figure 5 presents a 2000 point sample of the two distributions. VoroNet was evaluated with two metrics, (i) the distribution of neighbourhood size and (ii) the routing performance in terms of logical hops.

Distribution of neighbourhood sizes In an attempt to confirm the O(1) size of the each node's view of the system, we evaluate the distribution of neighbourhood size on all nodes. Figure 5.1, 5.2 and 5.3 show that the average and maximal number of neighbours in the overlay is kept small. As expected, for both distributions, the average number of Voronoi neighbours (Figure 5.1) is less than 6, the maximal degree is 24 and only 10 objects (among 300000) have degree higher than 12. The same holds for close neighbours (see Figure 5.2): the maximal degree (in the case of a sparse distribution) is less than 20 and less than 20 nodes have a degree higher than 15. Figure 5.3 depicts the number of back long range neighbours, *i.e.* the number of nodes that chose a particular node as their long link neighbour. In the uniform case, the number of such neighbours is small, as expected (see Section 3). The situation for the sparse distribution is more interesting. Indeed, in this case, some nodes are responsible for relatively large Voronoi cells, and one may expect their number of back long range neighbours to be rather high. Figure 5.3 proves that even for such strongly non uniform distributions, this is not the case (less than 30 nodes have a degree higher than 11). In fact, nodes that lie in large Voronoi cells do not have close neighbours, and since the probability of having a long link of distance d is



Uniform distribution

Sparse distribution

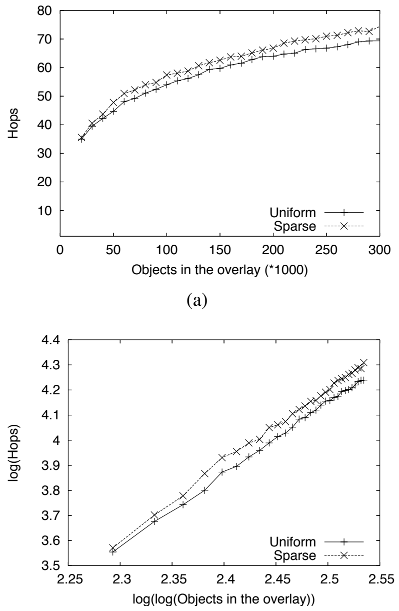
(5.1) $|\{\text{VN}(o)\}|$ distribution (5.2) $|\{\text{CN}(o)\}|$ distribution (5.3) $|\{\text{BLRN}(o)\}|$ distribution

Figure 5. Distribution of the sizes of neighbours set of nodes. (1) Voronoi neighbours ; (2) Close neighbours ; (3) Back long range neighbours, for Uniform and Sparse distributions of objects.

of order $\frac{1}{d^2}$, most long links are ”too short” to reach them. This explains why the overall number of neighbours of any node is small in the overlay, whatever the distribution.

Poly-logarithmic routing costs, long range neighbours Figure 6(a) presents the evolution of H, the number of hops, against the number of objects in the overlay. The evolution presents a polylogarithmic shape. To ensure this property, figure 6(b) depicts the evolution of $\log(H)$ against $\log(\log(|\mathcal{O}|))$, where $|\mathcal{O}|$ denotes the actual number of nodes in the overlay. In such a plot, a line shape corresponds to a routing in $O(\log^x(|\mathcal{O}|)$, where x is the slope of the line. These results prove that, for both distributions our algorithm achieves poly-logarithmic routing. Note that the value of $|\mathcal{O}|_{target}$ is set to 300000 during all the simulation, irrespective of the actual number of objects $|\mathcal{O}|$ in the overlay. Therefore, even if the theoretical results on $O(\log^2(|\mathcal{O}|))$ routing hold true only assuming that $|\mathcal{O}|$ is close to $|\mathcal{O}|_{target}$, the simulation results prove that our algorithm achieves poly-logarithmic routing even if $|\mathcal{O}|$ is much smaller than $|\mathcal{O}|_{target}$, during bootstrap for instance. In this case, the slope of the line is close to 2.8 (meaning that H is of order $O(\log(|\mathcal{O}|)^{2.8})$ for small values of $|\mathcal{O}|$ and becomes closer to 2 when $|\mathcal{O}|$ gets closer to $|\mathcal{O}|_{target}$.

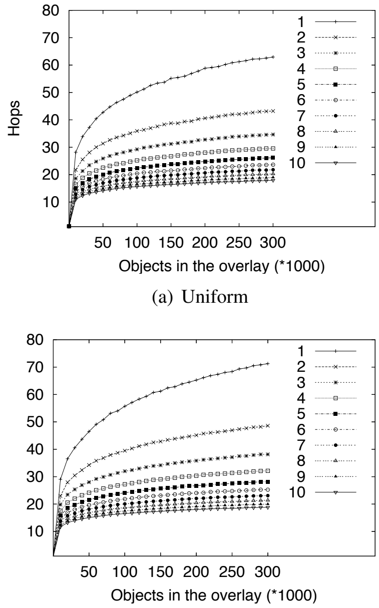
Finally, we analyze the impact of using more than one long range neighbour per node. All links are drawn us-



(a)

(b)

Figure 6. Mean route lengths as a function of the overlay size for the two distributions: (a) linear plot and (b) log(H) as a function of log(log($|\mathcal{O}|$)).



(a) Uniform

(b) Sparse

Figure 7. Study of the influence of the number of long range links on routing.

ing the same algorithm. Figure 5 shows that increasing the number of long range neighbours consistently improves the routing performance: roughly, as expected theoretically (for small values of k), using k long links per node improves the routing length by a factor k. Practically, such a configuration would be privileged to reduce the stress on the network and latency.

6 Related Work

In this section, we survey some routing protocols and overlays based on Voronoi tessellations or their dual, Delaunay triangulations, and extensions of the Kleinberg's model, that are related to our work. First, Steiner and Biersack [12] propose to use Delaunay triangulations in two or three dimensions to build peer to peer networks in the context of virtual community networks. Their approach deals with the three dimensional case, but does not deal with long range contacts nor calculation degeneracy. Effectively, the proposed algorithm is based on edge flipping, which is known to behave badly in presence of calculation degeneracy [15]. Delaunay-based networks are also used in ad-hoc networks, see [10], where the construction of the overlay is constrained by ranges of transmission allowed to nodes (sensors). On the other hand, some work has been done to adapt the Kleinberg model to higher dimensions. It has been demonstrated in [7] that it is possible to extend Kleinberg's model to more general graphs such as Cayley graphs or graphs whith certain ball growth. Also, Barrière *et al.* work [4] on generalization of Kleinberg's results to tori of higher dimensions is of particular interest in our context. These theoretical frameworks present proofs of feasibility, but no detailed protocols providing desired properties. Another method for adding long range links to a general graph is the HopLevel protocol [1], where long range contacts are created in a lazy way. The main drawback of this method is its lack of theoretical analysis, an aspect VoroNet is particularly aiming at. In [2], the authors introduce Skip Webs, an efficient data structure for multi-dimensional data sets. The approach proposed by the authors is completely different from ours (it is based on a generalization of skip lists rather than Kleinberg's model), although they share the same goal of dealing efficiently with range and neighbour queries on multi-dimensional attribute spaces. Routing in Skip Webs is slightly more efficient than in VoroNet ($O(\log N)$ instead of $O(\log^2(|\mathcal{O}|))$) but the size of data structures stored at each node is higher ($O(\log N)$ instead of $O(1)$). To the best of our knowledge, VoroNet is the first work to combine these aspects all together, using Voronoi diagrams and extension of the Kleinberg's model for poly-logarithmic routing, along with proofs and evaluations.

7 Conclusion and perspectives

In this paper, we have proposed VoroNet, a object-based scalable overlay network relying on Voronoi tessellations. VoroNet links application objects rather than physical nodes in the attribute space, thus allowing a support for further research on efficient query mechanisms, such as range queries. It is based on Voronoi diagrams, and inheritates their structural properties, such as O(1) direct neighbour set. Moreover, VoroNet is the basis for the proposal of a generalization of Kleinberg's work to generic data distributions, providing poly-logarithmic greedy routing and fair distributions of neighbourhood sizes, even with highly sparse distributions.

This research opens the way to many perspectives. First, the nature of the network enables to design and evaluate rich query mechanisms. The simplest one is a range query on one of the attribute: this query may be represented as a segment in the unit square. Then all objects lying on this segment can be reached easily by forwarding the query along this line, potentially splitting the line in different subsets for reduced latencies. Moreover, Delaunay triangulation is known to be a t-spanner [10, 6], that is for any subset of the graph, it is possible to efficiently build a spanning tree. Since every square-like subset of the network is itself a Delaunay triangulation, we may use this property to build efficient range query mechanisms. Even richer query mechanisms, such as radius queries, where all objects in a given disk are queried, can also be considered. We are currently designing these query mechanisms using the VoroNet network capabilities as the key to efficiency and comprehensiveness of the queries.

Second, we want to investigate the support of fault-tolerance mechanisms. In the current version of VoroNet, it is assumed that a physical node leaving the network leaves fairly, that is it redistributes the Voronoi region of all its objects to other objects in their local vicinity, and therefore recomputes the local Voronoi diagrams of other objects concerned by its objects departure. This will not happen if the physical node crashes, and this may lead to routing failures, or degeneracy of the diagram. A straightforward solution would be to use the fact that objects already know the zones, addresses and points of their Voronoi neighbours. A simple heartbeat-based monitoring protocol between an object and one of its (elected) Voronoi neighbours can detect physical nodes failures and object deletion. A proactive reconstruction of the local diagram at the initiative of the monitoring neighbour, which require to query for the failed object old neighbourhood (by constrained flooding, for instance), would permit to repair the overlay rapidly with a limited cost. Long range links and close neighbours may be repaired in a lazy way, as they permit efficient routing but are not vital to its correct termination. Another possibility for supporting fault tolerance as a key design decision could be to use gossip-based overlay construction protocols, and to construct Voronoi cells and long range contacts locally at each object by gossiping with peers about their actual partial view of the network, and locally evolve towards globally desired properties.

Third, the generalization of VoroNet to upper dimensions for the attribute space is also under work. Generalizations of Voronoi diagrams in high dimensions exist, but the average number of neighbours is not bounded in this case, so that it is impossible to guarantee that all data structures are of order $O(1)$. Moreover, computing the Voronoi diagram in dimension d is of cost $O(n \log n + n^{\lceil \frac{d}{2} \rceil})$, and is prone to higher number of error propagations. Nevertheless, Voronoi diagrams have strong theoretical properties, whereas the main issue in our routing algorithm is to find at each node at least one close neighbour that is closer to the target point. Indeed, the use of long range links is enough to ensure polylogarithmic routing. Because generalizing directly the Voronoi diagram approach to dimension higher than 2 poses limitations, we are currently investigating the possibility to use a constrained size set of neighbors at each node, on a higher dimension attribute space, while keeping the (loose) property of routing effectiveness and providing routing efficiency by a similar use of long range contacts.

References

[1] F. Araújo and L. Rodrigues. Long range contacts in overlay networks. In *Euro-par 2005*, pages 1153–1162, Lisbon, Portugal, August 2005. Springer-Verlag, LNCS 3648.

[2] L. Arge, D. Eppstein, and M. T. Goodrich. Skip-webs: efficient distributed data structures for multi-dimensional data sets. In *PODC '05: Proceedings of the twenty-fourth annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 69–76, New York, NY, USA, 2005. ACM Press.

[3] F. Aurenhammer and R. Klein. Voronoi diagrams. In J. Sack and G. Urrutia, editors, *Handbook of Computational Geometry*, pages 201–290. Elsevier Science Publishing, 2000.

[4] L. Barrière, P. Fraigniaud, E. Kranakis, and D. Krizanc. Efficient routing in networks with long range contacts. In *DISC '01: Proceedings of the 15th International Conference on Distributed Computing*, pages 270–284, London, UK, 2001. Springer-Verlag.

[5] O. Beaumont, A.-M. Kermarrec, L. Marchal, and É. Rivière. VoroNet: A scalable object network based on voronoi tessellations. Research report, INRIA, Feb 2006.

[6] J.-D. Boissonnat and M. Yvinec. *Algorithmic Geometry*. Cambridge University Press, 1998.

[7] P. Duchon, N. Hanusse, E. Lebhar, and N. Schabanel. Could any graph be turned into a small world? *Theoretical Computer Science*, 2005.

[8] J. Kleinberg. Navigation in a small world. *Nature*, 406, 2000.

[9] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proc. 32nd ACM Symposium on Theory of Computing*, 2000.

[10] X.-Y. Li, Y. Wang, and O. Frieder. Localized routing for wireless ad hoc networks. In *Proc. of IEEE ICC*, 2003.

[11] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proc. of Middleware*, 2001.

[12] M. Steiner and E.-W. Biersack. A fully distributed peer to peer structure based on 3D Delaunay triangulation. In *Algotel 2005, 7emes Rencontres Francophones sur les aspects Algorithmiques des Télécommunications, May 11-13, 2005 - Presqu'île de Giens, France*, May 2005.

[13] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. of SIGCOMM'01*, 2001.

[14] K. Sugihara. Robust geometric computation based on topological consistency. In *ICCS '01: Proceedings of the International Conference on Computational Sciences-Part I*, pages 12–26, London, UK, 2001. Springer-Verlag.

[15] K. Sugihara and M. Iri. Construction of the Voronoi Diagram for "One Million" Generators in Single-Precision Arithmetic. *Proceedings of the IEEE*, 80:1471–1484, 1992.