# Link Discovery with Guaranteed Reduction Ratio in Affine Spaces with Minkowski Measures

Axel-Cyrille Ngonga Ngomo

Department of Computer Science
University of Leipzig
Johannisgasse 26, 04103 Leipzig
ngonga@informatik.uni-leipzig.de
http://bis.uni-leipzig.de/AxelNgonga

**Abstract.** Time-efficient algorithms are essential to address the complex linking tasks that arise when trying to discover links on the Web of Data. Although several lossless approaches have been developed for this exact purpose, they do not offer theoretical guarantees with respect to their performance. In this paper, we address this drawback by presenting the first Link Discovery approach with theoretical quality guarantees. In particular, we prove that given an achievable reduction ratio $r$, our Link Discovery approach $\mathcal{HR}^3$ can achieve a reduction ratio $r' \leq r$ in a metric space where distances are measured by the means of a Minkowski metric of any order $p \geq 2$. We compare $\mathcal{HR}^3$ and the HYPPO algorithm implemented in LIMES 0.5 with respect to the number of comparisons they carry out. In addition, we compare our approach with the algorithms implemented in the state-of-the-art frameworks LIMES 0.5 and SILK 2.5 with respect to runtime. We show that $\mathcal{HR}^3$ outperforms these previous approaches with respect to runtime in each of our four experimental setups.

## 1 Introduction

One of the key principles of the Linked Data paradigm is the inclusion of links between data sets [1]. While this principle is central for tasks such as federated querying [20], cross-ontology question answering [12], large-scale inferences [22] and data integration [3], it is increasingly tedious to implement manually. One of the main difficulty behind the discovery of links is its intrinsic time complexity. Over the last five years, the Linked Data Web has evolved from 12 knowledge bases (May 2007) to more than 295 knowledge bases in September 2011 which contain more than 31 billion triples[1]. The combination of the mere size of these knowledge bases and the quadratic a-priori time complexity of Link Discovery leads to brute-force algorithms requiring weeks and even longer to compute links between large knowledge bases such as DBpedia[2] and LinkedGeoData[3].

[1] http://www4.wiwiss.fu-berlin.de/lodcloud/state/
[2] http://dbpedia.org
[3] http://linkedgeodata.org

Addressing this challenge demands the development of time-efficient and lossless solutions for the computation of links. Link Discovery frameworks such as LIMES [15,14] and SILK [9] have been designed to address this challenge. Yet, none of the manifold approaches they implement provides theoretical guarantees with respect to their performance. Thus, so far, it was impossible to predict how Link Discovery frameworks would perform w.r.t. time or space requirements. Consequently, the deployment of techniques such as customized memory management [5] or time-optimization strategies [23] (e.g., automated scaling for cloud computing when provided with very complex linking tasks) was rendered very demanding if not impossible.

In this paper, we introduce the novel approach $\mathcal{HR}^3$. Similar to the HYPPO algorithm [14] (on whose formalism it is based), $\mathcal{HR}^3$ assumes that the property values that are to be compared are expressed in an affine space with a Minkowski distance. Consequently, it can be most naturally used to process the portion of link specifications that compare numeric values (e.g., temperatures, elevations, populations, etc.). $\mathcal{HR}^3$ goes beyond the state of the art by being able to carry out Link Discovery tasks with *any achievable reduction ratio* [6]. This theoretical guarantee is of practical importance, as it does not only allow our approach to be more time-efficient than the state of the art but also lays the foundation for the implementation of customized memory management and time-optimization strategies for Link Discovery. The three main contributions of this paper are thus as follows:

1. We present a novel indexing scheme for hypercubes in metric spaces with Minkowski distances. This scheme builds the basis upon which $\mathcal{HR}^3$ discards unnecessary comparisons.
2. We prove formally that $\mathcal{HR}^3$'s reduction ratio can be made arbitrarily close to the optimal reduction ratio. For this purpose, we first define the relative reduction ratio ($RRR$). We then show that $\mathcal{HR}^3$'s $RRR$ converges towards a lower bound and prove this bound to be exactly 1.
3. We show experimentally that in addition to providing theoretical guarantees, our approach outperforms the state of the art. For this purpose, we compare the number of comparisons carried out by $\mathcal{HR}^3$ and HYPPO. In addition, we compare $\mathcal{HR}^3$'s runtime with that of HYPPO (as implemented in LIMES) and SILK[4].

The rest of this paper is structured as follows: In Section 2, we present preliminaries and the notation used to formalize our approach $\mathcal{HR}^3$. We also introduce the relative reduction ratio $RRR$. We then prove that our algorithm can achieve any $RRR$ score larger than 1 and that we can therewith achieve any possible reduction ratio (Section 3). After a short presentation of the implementation of our algorithm in Section 4, we evaluate our approach against SILK and HYPPO in four experiments in Section 5. Subsequently, in Section 6, we give an overview of previous approaches to Link Discovery. Finally, we discuss our findings and conclude in Section 7.

[4] The algorithm was implemented in the new verison of LIMES, of which a demo is available at `http://limes.aksw.org`

## 2 Preliminaries

In this section, we present the preliminaries necessary to understand the subsequent parts of this work. In particular, we define the problem of Link Discovery, the reduction ratio and the relative reduction ratio formally as well as give an overview of space tiling for Link Discovery. The subsequent description of $\mathcal{HR}^3$ relies partly on the notation presented in this section.

### 2.1 Link Discovery

The goal of Link Discovery is to compute the set of pair of instances $(s,t) \in S \times T$ that are related by a relation $R$, where $S$ and $T$ are two not necessarily distinct sets of instances. One way to automate this discovery is to compare the $s \in S$ and $t \in T$ based on their properties using a distance measure. Two entities are then considered to be linked via $R$ if their distance is less or equal to a threshold $\theta$ [15].

**Definition 1 (Link Discovery on Distances).** *Given two sets $S$ and $T$ of instances, a distance measure $\delta$ over the properties of $s \in S$ and $t \in T$and a distance threshold $\theta \in [0, \infty[$, the goal of Link Discovery is to compute the set $\mathcal{M} = \{(s, t, \delta(s,t)) : s \in S \wedge t \in T \wedge \delta(s,t) \leq \theta\}$.*

Note that in this paper, we are only interested in lossless solutions, i.e., solutions that are able to find all pairs that abide by the definition given above.

### 2.2 Reduction Ratio

A brute-force approach to Link Discovery would execute a Link Discovery task on $S$ and $T$ by carrying out $|S||T|$ comparisons. One of the key ideas behind time-efficient Link Discovery algorithms $\mathcal{A}$ is to reduce the number of comparisons that are effectively carried out to a number $C(\mathcal{A}) < |S||T|$ [21]. The reduction ratio $RR$ of an algorithm $\mathcal{A}$ is given by

$$RR(\mathcal{A}) = 1 - \frac{C(\mathcal{A})}{|S||T|}. \tag{1}$$

$RR(\mathcal{A})$ captures how much of the Cartesian product $|S||T|$ was not explored before the output of $\mathcal{A}$ was reached. It is obvious that even an optimal lossless solution which performs only the necessary comparisons cannot achieve a $RR$ of 1. Let $C_{min}$ be the minimal number of comparisons necessary to complete the Link Discovery task without losing recall, i.e., $C_{min} = |\mathcal{M}|$. We define the relative reduction ratio $RRR(\mathcal{A})$ as the proportion of the minimal number of comparisons that was carried out by the algorithm $\mathcal{A}$ before it terminated. Formally

$$RRR(\mathcal{A}) = \frac{1 - \frac{C_{min}}{|S||T|}}{1 - \frac{C(\mathcal{A})}{|S||T|}} = \frac{|S||T| - C_{min}}{|S||T| - C(\mathcal{A})}. \tag{2}$$

$RRR(\mathcal{A})$ indicates how close $\mathcal{A}$ is to the optimal solution with respect to the number of candidates it tests. Given that $C(\mathcal{A}) \geq C_{min}$, $RRR(\mathcal{A}) \geq 1$. Note that the larger the value of $RRR(\mathcal{A})$, the poorer the performance of $\mathcal{A}$ with respect to the task at hand.

The main observation that led to this work is that while most algorithms aim to optimize their $RR$ (and consequently their $RRR$), current approaches to Link Discovery do not provide any guarantee with respect to the $RR$ (and consequently the $RRR$) that they can achieve. In this work, we present an approach to Link Discovery in metric spaces whose $RRR$ is guaranteed to converge to 1.

## 2.3 Space Tiling for Link Discovery

Our approach, $\mathcal{HR}^3$, builds upon the same formalism on which the HYPPO algorithm relies, i.e., space tiling. HYPPO addresses the problem of efficiently mapping instance pairs $(s, t) \in S \times T$ described by using exclusively numeric values in a $n$-dimensional metric space and has been shown to outperform the state of the art in previous work [14]. The observation behind space tiling is that in spaces $(\Omega, \delta)$ with orthogonal, (i.e., uncorrelated) dimensions[5], common metrics for Link Discovery can be decomposed into the combination of functions $\phi_{i,i\in\{1...n\}}$ which operate on exactly one dimension of $\Omega : \delta = f(\phi_1, ..., \phi_n)$. For Minkowski distances of order $p$, $\phi_i(x, \omega) = |x_i - \omega_i|$ for all values of $i$ and $\delta(x, \omega) = \sqrt[p]{\sum\limits_{i=1}^{n} \phi_i^p(x, \omega)^p}$. A direct consequence of this observation is the inequality $\phi_i(x, \omega) \leq \delta(x, \omega)$. The basic insight that results this observation is that the hypersphere $H(\omega, \theta) = \{x \in \Omega : \delta(x, \omega) \leq \theta\}$ is a subset of the hypercube $V$ defined as $V(\omega, \theta) = \{x \in \Omega : \forall i \in \{1...n\}, \phi_i(x_i, \omega_i) \leq \theta\}$. Consequently, one can reduce the number of comparisons necessary to detect all elements of $H(\omega, \theta)$ by discarding all elements which are not in $V(\omega, \theta)$ as non-matches. Let $\Delta = \theta/\alpha$, where $\alpha \in \mathbb{N}$ is the *granularity parameter* that controls how fine-grained the space tiling should be (see Figure 1 for an example). We first tile $\Omega$ into the adjacent hypercubes (short: cubes) $C$ that contain all the points $\omega$ such that

$$\forall i \in \{1...n\}, c_i\Delta \leq \omega_i < (c_i + 1)\Delta \text{ with } (c_1, ..., c_n) \in \mathbb{N}^n. \quad (3)$$

We call the vector $(c_1, ..., c_n)$ the coordinates of the cube $C$. Each point $\omega \in \Omega$ lies in the cube $C(\omega)$ with coordinates $(\lfloor \omega_i/\Delta \rfloor)_{i=1...n}$. Given such a space tiling, it is obvious that $V(\omega, \theta)$ consists of the union of the cubes such that $\forall i \in \{1...n\} : |c_i - c(\omega)_i| \leq \alpha$.

Like most of the current algorithms for Link Discovery, space tiling does not provide optimal performance guarantees. The main goal of this paper is to build upon the tiling idea so as to develop an algorithm that can achieve any possible $RR$. In the following, we present such an algorithm, $\mathcal{HR}^3$.

[5] Note that in all cases, a space transformation exists that can map a space with correlated dimensions to a space with uncorrelated dimensions.

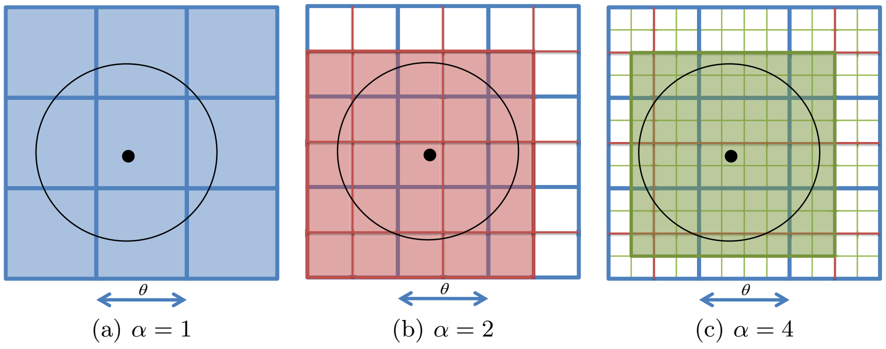


**Fig. 1.** Space tiling for different values of $\alpha$. The colored squares show the set of elements that must be compared with the instance located at the black dot. The points within the circle lie within the distance $\theta$ of the black dot. Note that higher values of $\alpha$ lead to a better approximation of the hypersphere but also to more hypercubes.

## 3 Approach

The goal of the $\mathcal{HR}^3$ algorithm is to efficiently map instance pairs $(s, t) \in S \times T$ that are described by using exclusively numeric values in a $n$-dimensional metric space where the distances are measured by using any Minkowski distance of order $p \geq 2$. To achieve this goal, $\mathcal{HR}^3$ relies on a *novel indexing scheme* that allows achieving any *RRR* greater than or equal to than 1. In the following, we first present our new indexing scheme and show that we can discard more hypercubes than simple space tiling for all granularities $\alpha$ such that $n(\alpha-1)^p > \alpha^p$. We then prove that by these means, our approach can achieve any *RRR* greater than 1, therewith proving the *optimality of our indexing scheme* with respect to *RRR*.

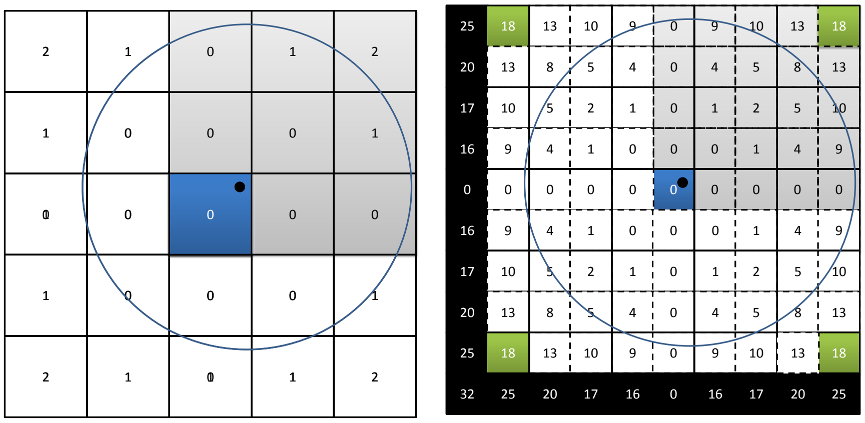
### 3.1 Indexing Scheme

Let $\omega \in \Omega = S \cup T$ be an arbitrary reference point. Furthermore, let $\delta$ be the Minkowski distance of order $p$. We define the *index* function as follows:

$$index(C, \omega) = \begin{cases} 0 \text{ if } \exists i : |c_i - c(\omega)_i| \leq 1 \text{ with } i \in \{1, ..., n\}, \\ \sum\limits_{i=1}^{n} (|c_i - c(\omega)_i| - 1)^p \text{ else,} \end{cases} \tag{4}$$

where $C$ is a hypercube resulting from a space tiling and $\omega \in \Omega$. Figure 2 shows an example of such indexes for $p = 2$ with $\alpha = 2$ (Figure 2(a)) and $\alpha = 4$ (Figure 2(b)).

Note that the blue square with index 0 contains the reference point $\omega$. Also note that our indexing scheme is symmetric with respect to $C(\omega)$. Thus, it is sufficient to prove the subsequent lemmas for hypercubes C such that $c_i > c(\omega)_i$. In Figure 2, this is the upper right portion of the indexed space with the gray



(a) $\alpha = 2$ (b) $\alpha = 4$

**Fig. 2.** Space tiling and resulting index for a two-dimensional example. Note that the index in both subfigures was generated for exactly the same portion of space. The black dot stands for the position of $\omega$.

background. Finally, note that the maximal index that a hypercube can achieve is $n(\alpha - 1)^p$ as $\max|c_i - c_i(\omega)| = \alpha$ per construction of $H(\omega, \theta)$.

The indexing scheme proposed above guarantees the following:

**Lemma 1.** *$index(C, \omega) = x \rightarrow \forall s \in C(\omega)\ \forall t \in C\ \delta^p(s,t) > x\Delta^p$.*

*Proof.* This lemma is a direct implication of the construction of the index. $index(C, \omega) = x$ implies that

$$\sum_{i=1}^{n} (c_i - c(\omega)_i - 1)^p = x.$$

Now given the definition of the coordinates of a cube (Eq. 3), the following holds:

$$\forall s \in C(\omega)\ \forall t \in C\ |s_i - t_i| \geq (|c_i - c(\omega)_i| - 1)\Delta.$$

Consequently,

$$\forall s \in C(\omega)\ \forall t \in C\ \sum_{i=1}^{n} |s_i - t_i|^p \geq \sum_{i=1}^{n} (|c_i - c(\omega)_i| - 1)^p \Delta^p.$$

By applying the definition of the Minkowski distance of the index function, we finally get $\forall s \in C(\omega)\ \forall t \in C\ \delta^p(s,t) > x\Delta^p$. □

Note that given that $\omega \in C(\omega)$, the following also holds:

$$index(C, \omega) = x \rightarrow \forall t \in C : \delta^p(\omega, t) > x\Delta^p. \tag{5}$$

## 3.2 $\mathcal{HR}^3$

The main insight behind $\mathcal{HR}^3$ is that in spaces with Minkowski distances, the indexing scheme proposed above allows to safely (i.e., without dismissing correct matches) discard more hypercubes than when using simple space tiling. More specifically,

**Lemma 2.** *$\forall s \in S : index(C, s) > \alpha^p$ implies that all $t \in C$ are non-matches.*

*Proof.* This lemma follows directly from Lemma 1 as

$$index(C, s) > \alpha^p \rightarrow \forall t \in C, \delta^p(s, t) > \Delta^p \alpha^p = \theta^p. \quad (6)$$

For the purpose of illustration, let us consider the example of $\alpha = 4$ and $p = 2$ in the two-dimensional case displayed in Figure 2(b). Lemma 2 implies that any point contained in a hypercube $C_{18}$ with index 18 cannot contain any element $t$ such that $\delta(s, t) \leq \theta$. While space tiling would discard all black cubes in Figure 2(b) but include the elements of $C_{18}$ as candidates, $\mathcal{HR}^3$ discards them and still computes exactly the same results, yet with a better (i.e., smaller) *RRR*.

One of the direct consequences of Lemma 2 is that $n(\alpha - 1)^p > \alpha^p$ is a necessary and sufficient condition for $\mathcal{HR}^3$ to achieve a better *RRR* than simple space tiling. This is simply due to the fact that the largest index that can be assigned to a hypercube is $\sum\limits_{i=1}^{n}(\alpha-1)^p = n(\alpha-1)^p$. Now, if $n(\alpha-1)^p > \alpha^p$, then this cube can be discarded. For $p = 2$ and $n = 2$ for example, this condition is satisfied for $\alpha \geq 4$. Knowing this inequality is of great importance when deciding on when to use $\mathcal{HR}^3$ as discussed in Section 5.

Let $\mathcal{H}(\alpha, \omega) = \{C : index(C, \omega) \leq \alpha^p\}$. $\mathcal{H}(\alpha, \omega)$ is the approximation of the hypersphere $H(\omega) = \{\omega' : \delta(\omega, \omega') \leq \theta\}$ generated by $\mathcal{HR}^3$. We define the volume of $\mathcal{H}(\alpha, \omega)$ as

$$V(\mathcal{H}(\alpha, \omega)) = |\mathcal{H}(\alpha, \omega)|\Delta^p. \quad (7)$$

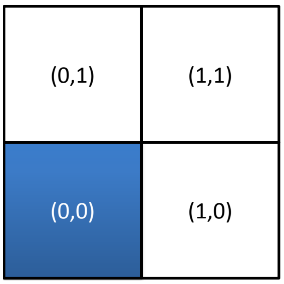
To show that given any $r > 1$, the approximation $\mathcal{H}(\alpha, \omega)$ can always achieve a an $RRR(\mathcal{HR}^3) \leq r$, we begin by showing that

**Lemma 3.** *$\forall \alpha > 1$ $V(\mathcal{H}(\alpha, \omega)) > V(\mathcal{H}(2\alpha, \omega))$.*

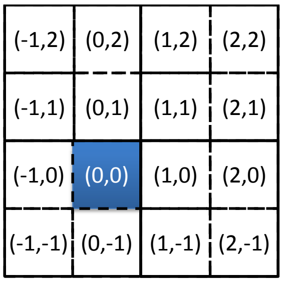
*Proof.* Any cube $C$ discarded by $\mathcal{HR}^3(\alpha)$ is split into $2^n$ cubes $\mathcal{C}$ by $\mathcal{HR}^3(2\alpha)$, each of which has the coordinates $2c_i$ or $2c_i + 1$. In the worst case for $\mathcal{HR}^3$, $\omega$ is assigned the coordinates $2c_i(\omega) + 1$. Figure 3 exemplifies this property of our indexing scheme. Figure 3(b) is an indexing of the same space with the the twofold granularity.

When processed by $\mathcal{HR}^3(2\alpha)$, the minimal index of a hypercube $\mathcal{C}$ is then given by

$$\min index(\mathcal{C}) = \sum_{i=1}^{n}(2c_i - (2c_i(\omega) + 1) - 1)^p = \sum_{i=1}^{n} 2^p(c_i - c(\omega) - 1)^p.$$



(a) Initial coordinates



(b) Coordinates with two-fold granularity

**Fig. 3.** Commparison of coordinates for granularities $\alpha$ and $2\alpha$

Given that C was discarded, we know that $\sum_{i=1}^{n}(c_i - c_i(\omega) - 1) > \alpha^p$. Consequently,

$$\min index(\mathcal{C}) > (2\alpha)^p.$$

This leads to all $\mathcal{C}$ that were discarded by $\mathcal{HR}^3(\alpha)$ also being discarded by $\mathcal{HR}^3(2\alpha)$. Proving our lemma is consequently equivalent to showing that there is a hypercube $C' \in \mathcal{H}(\alpha, \omega)$ that is such that one of the $2^n$ cubes $\mathcal{Q}$ it is split into gets discarded by $\mathcal{HR}^3(2\alpha)$. An example of such a case for $p = 2$ and $n = 2$ is shown in Figures 4(a) and 4(b). For $\alpha = 4$, the cubes that are adjacent to the corner and do not lie on the diagonal of the square are not excluded. Yet, for $\alpha = 8$, 2 of the hypercubes in which they are split are discarded.

Let $C = (c_1, ..., c_n) \notin \mathcal{H}(\omega, \alpha)$ while $C' = (c_1 - 1, ..., c_n) \in \mathcal{H}(\omega, \alpha)$. In the following, we show that the hypercube $\mathcal{Q} = (2c_1 - 1, 2c_2 + 1, ..., 2c_n + 1)$, which is one of the hypercubes that $C'$ gets split into by virtue of its coordinates,[6] will be discarded by $\mathcal{HR}^3(2\alpha)$, i.e., $\mathcal{Q} \notin \mathcal{H}(\omega, 2\alpha)$.

We know that $C = (c_1, ..., c_n)$ gets discarded, i.e., $\sum_{i=1}^{n}(c_i - c_i(\omega) - 1)^p > \alpha^p$.

Now, $\min index(\mathcal{Q}) = (2c_1 - (2c_1(\omega)+1) - 1)^p + \sum_{i=2}^{n}(2c_i + 1 - (2c_i(\omega)+1) - 1)^p$.

Consequently, $\min index(\mathcal{Q}) = 2^p(c_1 - c_1(\omega) - 1)^p + \sum_{i=2}^{n}[2(c_i - c_i(\omega) - 1) + 1]^p$.

This value is obviously larger than $\sum_{i=1}^{n}[2(c_i - c_i(\omega) - 1)]^p$. From the premise that $\sum_{i=1}^{n}(c_i - c_i(\omega) - 1)^p > \alpha^p$, we can finally infer that $\min index(\mathcal{Q}) > (2\alpha)^p$. Thus, we can conclude that $\forall\alpha > 1\ V(\mathcal{H}(\alpha, \omega)) > V(\mathcal{H}(2\alpha, \omega))$. □

One of the consequences of Lemma 2 w.r.t. $RRR(\mathcal{HR}^3, \alpha)$, i.e., the $RRR$ achieved by $\mathcal{HR}^3$ when the granularity is set to $\alpha$, is

$$\forall\alpha > 1: \ RRR(\mathcal{HR}^3, \alpha) > RRR(\mathcal{HR}^3, 2\alpha). \quad (8)$$

[6] Note that $2c_1 - 1 = 2(c_1 - 1) + 1$.

Note that this inequality is not sufficient to prove that we can achieve any $RRR$ greater than 1, as series can converge to any real number. Consequently, we still need to show the following:

**Lemma 4.** $\lim\limits_{\alpha\to\infty} RRR(\mathcal{HR}^3, \alpha) = 1$.

*Proof.* The cubes that are not discarded by $\mathcal{HR}^3(\alpha)$ are those for which $(|c_i - c_i(\omega)| - 1)^p \leq \alpha^p$. When $\alpha \to \infty$, $\Delta$ becomes infinitesimally small, leading to the cubes being single points. Each cube $C$ thus contains a single point $x$ with coordinates $x_i = c_i\Delta$. Especially, $c_i(\omega) = \omega$. Consequently,

$$\sum_{i=1}^{n}(|c_i - c_i(\omega)| - 1)^p \leq \alpha^p \leftrightarrow \sum_{i=1}^{n}\left(\frac{|x_i - \omega_i| - \Delta}{\Delta}\right)^p \leq \alpha^p. \tag{9}$$

Given that $\theta = \Delta\alpha$, we get

$$\sum_{i=1}^{n}\left(\frac{|x_i - \omega_i| - \Delta}{\Delta}\right)^p \leq \alpha^p \leftrightarrow \sum_{i=1}^{n}(|x_i - \omega_i| - \Delta)^p \leq \theta^p. \tag{10}$$

Finally, $\Delta \to 0$ when $\alpha \to \infty$ leads to

$$\sum_{i=1}^{n}(|x_i - \omega_i| - \Delta)^p \leq \theta^p \wedge \alpha \to \infty \to \sum_{i=1}^{n}|x_i - \omega_i|^p \leq \theta^p. \tag{11}$$

This is exactly the condition for linking specified in Definition 1 applied to Minkowski distances of order $p$. Consequently, $\mathcal{H}(\omega, \infty)$ is exactly $H(\omega, \theta)$ for any $\theta$. Thus, the number of comparisons carried out by $\mathcal{HR}^3(\alpha)$ when $\alpha \to \infty$ is exactly $C_{min}$, which leads to the conclusion $\lim\limits_{\alpha\to\infty} RRR(\mathcal{HR}^3, \alpha) = 1$. □

Our conclusion is illustrated by Figure 4, which shows the approximations computed by $\mathcal{HR}^3$ for different values of $\alpha$ with $p = 2$ and $n = 2$. The higher $\alpha$, the closer the approximation is to a circle. Note that these results allow to conclude that for any $RRR$-value $r$ larger than 1, there is a setting of $\mathcal{HR}^3$ that can compute links with a $RRR$ smaller or equal to $r$.

## 4 Implementation

The $\mathcal{HR}^3$ algorithm was implemented as shown in Algorithm 1. It is important to notice that the memory requirements of $\mathcal{HR}^3$ are smaller than those of most other approaches and especially than those of simple space tiling for any $\alpha$ such that $n(\alpha - 1)^p > \alpha^p$, as $\mathcal{HR}^3$ then generates less hypercubes. Yet, $\mathcal{HR}^3$ requires one supplementary computational step as it has to compute the index of cubes before discarding the unnecessary ones. Consequently, although we have shown that $\mathcal{HR}^3$ can achieve any $RRR > 1$, the question that remains to elucidate is whether this theoretical guarantee also offers a practically superior algorithm w.r.t. its runtime. That is the goal of the subsequent evaluation.

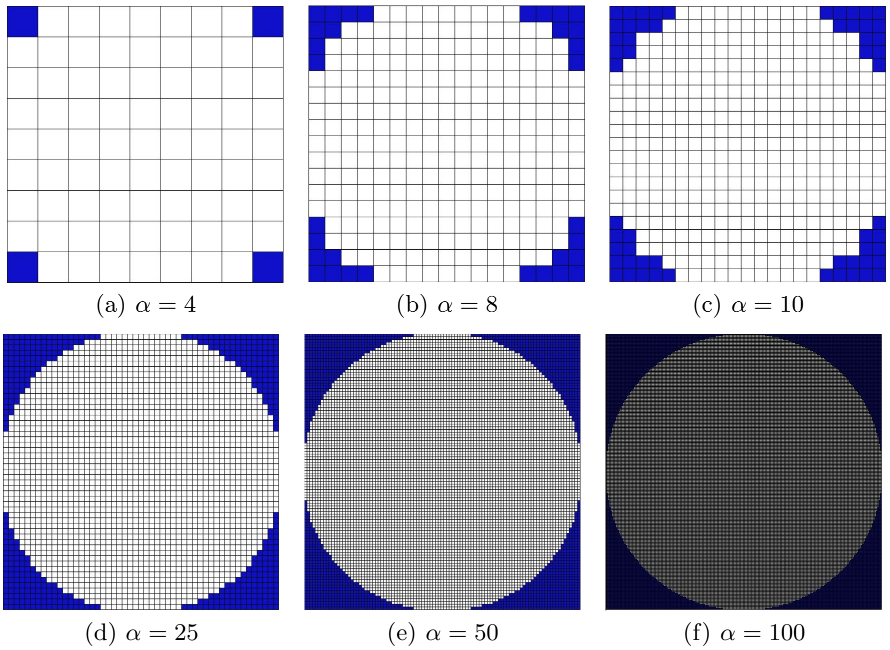


**Fig. 4.** Approximation generated by $\mathcal{HR}^3$ for different values of $\alpha$. The white squares are selected whilst the colored ones are discarded.

## 5 Evaluation

### 5.1 Experimental Setup

We carried out four experiments to compare $\mathcal{HR}^3$ with LIMES 0.5's HYPPO and SILK 2.5.1. In the first and second experiments, we aimed to deduplicate DBpedia places by comparing their names (`rdfs:label`), minimum elevation, elevation and maximum elevation. We retrieved 2988 entities that possessed all four properties. We use the Euclidean metric on the last three values with the thresholds 49 meters resp. 99 meters for the first resp. second experiment. The third and fourth experiments aimed to discover links between Geonames and LinkedGeoData. Here, we compared the labels (`rdfs:label`), longitude and latitude of the instances. This experiment was of considerably larger scale than the first one, as we compared 74458 entities in Geonames with 50031 entities from LinkedGeoData. Again, we measured the runtime necessary to compare the numeric values when comparing them by using the Euclidean metric. We set the distance thresholds to 1 resp. 9° in experiment 3 resp. 4. We ran all experiments on the same Windows 7 Enterprise 64-bit computer with a 2.8GHz i7 processor with 8GB RAM. The JVM was allocated 7GB RAM to ensure that the runtimes were not influenced by swapping. Only one of the kernels of the processors was used. Furthermore, we ran each of the experiments three times and report the best runtimes in the following.

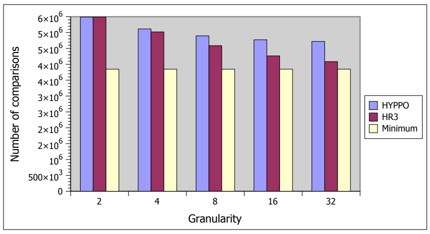
**Algorithm 1.** The $\mathcal{HR}^3$ algorithm

```
Require: Source knowledge base $S$, target knowledge base $T$, distance threshold $\theta$,
  Minkowski distance $\delta$ of order $p$, granularity factor $\alpha$
  Mapping $M := \emptyset$
  $\Delta = \theta/\alpha$
  for $\omega \in S \cup T$ do
    $C(\lfloor\omega_1/\Delta\rfloor, ..., \lfloor\omega_n/\Delta\rfloor) := C(\lfloor\omega_1/\Delta\rfloor, ..., \lfloor\omega_n/\Delta\rfloor) \cup \{\omega\}$
  end for
  for $s \in S$ do
    for $C \in \mathcal{H}(s, \alpha)$ do
      for $t \in C \cap T$ do
        if $\delta(s,t) \leq \theta$ then
          $M := M \cup (s, t, \delta(s,t))$
        end if
      end for
    end for
  end for
  return $M$
```

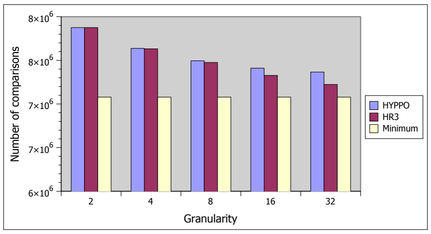
### 5.2 Results

We first measured the number of comparisons required by HYPPO and $\mathcal{HR}^3$ to complete the tasks at hand (see Figure 5). Note that we could not carry out this section of the evaluation for SILK2.5.1 as it would have required altering the code of the framework. In the experiments 1, 3 and 4, $\mathcal{HR}^3$ can reduce the overhead in comparisons (i.e., the number of unnecessary comparisons divided by the number of necessary comparisons) from approximately 24% for HYPPO to approximately 6% (granularity = 32). In experiment 2, the overhead is reduced from 4.1% to 2%. This difference in overhead reduction is mainly due to the data clustering around certain values and the clusters having a radius between 49 meters and 99 meters. Thus, running the algorithms with a threshold of 99 meters led to only a small a-priori overhead and HYPPO performing remarkably well. Still, even on such data distributions, $\mathcal{HR}^3$ was able to discard even more data and to reduce the number of unnecessary computations by more than 50% relative. In the best case (Exp. 4, $\alpha = 32$, see Figure 5(d)), $\mathcal{HR}^3$ required approximately $4.13 \times 10^6$ less comparisons than HYPPO for $\alpha = 32$. Even for the smallest setting (Exp. 1, see Figure 5(a)), $\mathcal{HR}^3$ still required $0.64 \times 10^6$ less comparisons.

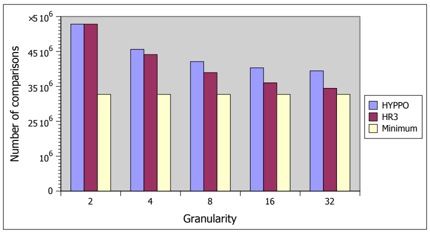
We also measured the runtimes of SILK, HYPPO and $\mathcal{HR}^3$. The best runtimes of the three algorithms for each of the tasks is reported in Figure 6. Note that SILK's runtimes were measured without the indexing time, as the data fetching and indexing are merged to one process in SILK. Also note that in the second experiment, SILK did not terminate due to higher memory requirements. We approximated SILK's runtime by extrapolating approximately 11 min it required for 8.6% of the computation before the RAM was filled. Again, we did not consider the indexing time.



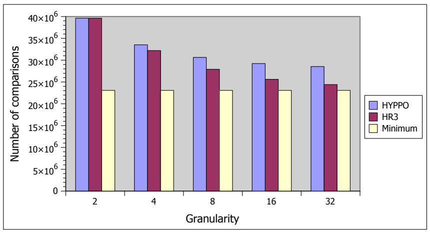
(a) Experiment 1



(b) Experiment 2



(c) Experiment 3



(d) Experiment 4

**Fig. 5.** Number of comparisons for $\mathcal{HR}^3$ and HYPPO

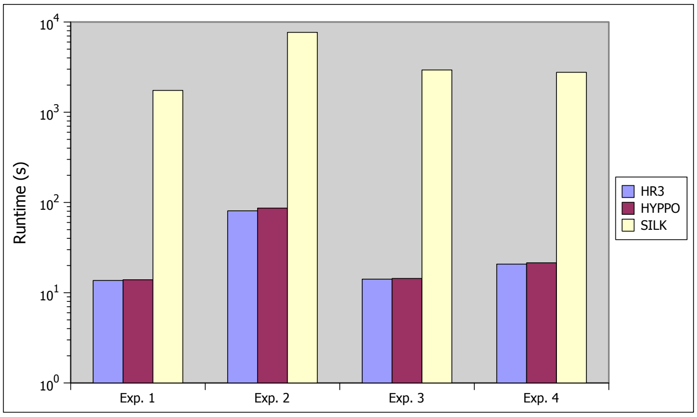
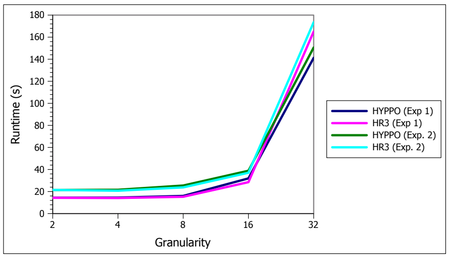


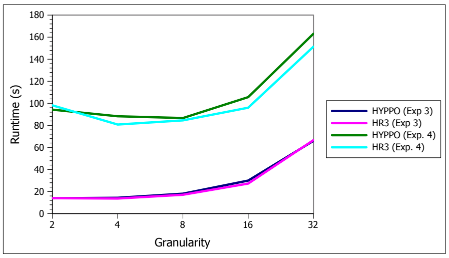
**Fig. 6.** Comparison of the runtimes of $\mathcal{HR}^3$, HYPPO and SILK2.5.1

Due to the considerable difference in runtime (approximately 2 orders of magnitude) between HYPPO and SILK, we report solely HYPPO and $\mathcal{HR}^3$'s runtimes in the detailed runtimes figures 7(a) and 7(b). Overall, $\mathcal{HR}^3$ outperformed the other two approaches in all experiments, especially for $\alpha = 4$. It is important to note that the improvement in runtime increases with the complexity of the experiment. For example, while $\mathcal{HR}^3$ outperforms HYPPO by 3% in the second experiment, the different grows to more than 7% in the fourth experiment. In addition, the improvement in runtime augments with the threshold. This can be seen in the third and fourth experiments. While $\mathcal{HR}^3$ is less than 2% faster in the third experiment, it is more than 7% faster when $\theta = 4$ the fourth experiment . As expected, $\mathcal{HR}^3$ is slower than HYPPO for $\alpha < 4$ as it carries out exactly the same comparisons but still has the overhead of computing the index. Yet, given that we know that $\mathcal{HR}^3$ is only better when $n(\alpha-1)^p > \alpha^p$, our implementation only carries out the indexing when this inequality holds. By these means, we can ensure that $\mathcal{HR}^3$ is only used when it is able to discard hypercubes that HYPPO would not discard, therewith reaching superior runtimes both with small and large values $\alpha$. Note that the difference between the improvement of the number of comparisons necessitated by $\mathcal{HR}^3$ and the improvement in runtime over all experiments is due to the supplementary indexing step required by $\mathcal{HR}^3$.

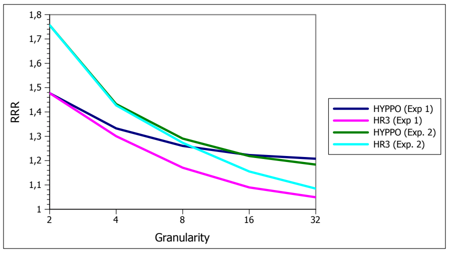
Finally, we measured the RRR of both $\mathcal{HR}^3$ and HYPPO (see Figures 7(c) and 7(d)). In the two-dimensional experiments 3 and 4, HYPPO achieves a RRR close to 1. Yet, it is still outperformed by $\mathcal{HR}^3$ as expected. A larger difference between the RRR of $\mathcal{HR}^3$ and HYPPO can be seen in the three-dimensional experiments, where the RRR of both algorithms diverge significantly. Note that the RRR difference grows not only with the number of dimensions but also with the size of the problem. The difference in RRR between HYPPO and $\mathcal{HR}^3$ does not always reflect the difference in runtime due to the indexing overhead of $\mathcal{HR}^3$. Still, for $\alpha = 4$, $\mathcal{HR}^3$ generates a sufficient balance of indexing runtime and comparison runtime (i.e., RRR) to outperform HYPPO in all experiments.



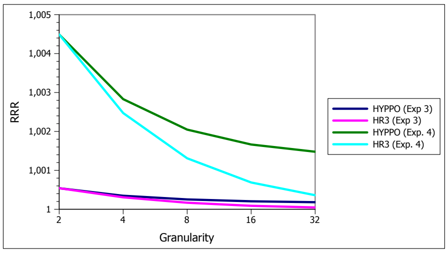
(a) Runtimes for experiments 1 and 2



(b) Runtimes for experiments 3 and 4



(c) RRR for experiments 1 and 2



(d) RRR for experiments 3 and 4

**Fig. 7.** Comparison of runtimes and RRR of $\mathcal{HR}^3$ and HYPPO

## 6 Related Work

The growing size and number of knowledge bases available in the Linked Data Cloud makes Link Discovery intrinsically complex with respect to its runtime. To address this issue, manifold time-efficient frameworks have been developed. LIMES [14] offers a complex grammar for link specifications that can be translated into a combination of time-efficient atomic mappers that are combined via a hybrid approach. For example, LIMES implements a dedicated approach for numeric values called HYPPO. SILK [9] implements a different Link Discovery paradigm and aims to place all instances that are to be compared in a multi-dimensional space. It then uses MultiBlock to discard unnecessary comparisons efficiently. In contrast to LIMES and SILK, which implement lossless approaches, the approach presented in [21] uses a candidate selection approach based on discriminative properties to compute links very efficiently but potentially loses links while doing so. Other frameworks and approaches include those described in [18,8,19].

Albeit Link Discovery is closely related with record linkage [7] and deduplication [4], it is important to notice that Link Discovery goes beyond these two tasks as Link Discovery aims to provide the means to link entities via arbitrary relations. Different blocking techniques such as standard blocking, sorted-neighborhood, bi-gram indexing, canopy clustering and adaptive blocking have been developed by the database community to address the problem of the quadratic time complexity of brute force comparison [11]. In addition, very time-efficient approaches have been proposed to compute string similarities for record linkage, including All-Pairs [2], PPJoin and PPJoin+ [24]. However, these approaches alone cannot deal with the diversity of property values found on the Web of Data as they cannot deal with numeric values. In addition, most time-efficient string matching algorithms can only deal with simple link specifications, which are mostly insufficient when computing links between large knowledge bases.

In recent work, the discovery of adequate link specifications has been addressed mainly by using machine learning approaches. For example, [21] detect discriminative properties by using string concatenations. RAVEN [16] combines stable marriage algorithms and a perceptron-based learning algorithm with the frame of active learning to compute boolean and linear classifiers. SILK [10] employs genetic programming to learn link configurations from positive and negative examples. [13] go a step further and combine genetic programming with active learning to discover high-accuracy link specificity with a small number of annotations. Another approach based on genetic programming is presented in [17]. Here, the authors show how link specifications can be learned without any input from the user. To the best of our knowledge, none of the approaches presented previously provide formal guarantees w.r.t. their performance. $\mathcal{HR}^3$ is the first matching approach that it guaranteed not to lose links while converging to the small possible reduction ratio. Note that while $\mathcal{HR}^3$ was designed for numeric values, it can be used in any space with Minkowski distances, for example for comparing indexes in multi-dimensional spaces. Thus, it can be used for any datatype mapped to a metric space.

## 7 Conclusion and Future Work

In this paper, we presented $\mathcal{HR}^3$, a time-efficient approach for the discovery of links in spaces with Minkowski distances. We proved that our approach can achieve is optimal w.r.t its reduction ration by showing that its $RRR$ converges towards 1 when $\alpha$ converges towards $\infty$. It is important to note that an optimal $RRR(\mathcal{A})$ does not necessarily mean that $\mathcal{A}$ outperforms algorithms with a poorer $RRR$ with respect to runtime as achieving a good $RRR$ score usually requires better preprocessing (usually in form of indexing), which might be more time-demanding than the combination of a rougher preprocessing and a run with a poorer $RRR$. Thus, in addition to proving formally that we can guarantee a $RRR$ that converges towards 1, we implemented our approach and compared it with the state-of-the-art algorithms HYPPO implemented in LIMES and SILK. We showed that we outperform both frameworks w.r.t. to their runtime and that we reach $RRR$ close to 1 for $\alpha$ as small as 32. Our experiments also showed that $\alpha = 4$ is a good setup for $\mathcal{HR}^3$. Our approach aims to be the first of a novel type of Link Discovery approaches, i.e., of approaches which can guarantee theoretical optimality while also being empirically usable. In future work, we will thus aim to develop more of such approaches and to make use of their theoretical characteristics for memory and space management. With respect to $\mathcal{HR}^3$, we will mainly improve the implementation of its indexing to ensure even better runtimes.

## References

1. Auer, S., Lehmann, J., Ngonga Ngomo, A.-C.: Introduction to Linked Data and Its Lifecycle on the Web. In: Polleres, A., d'Amato, C., Arenas, M., Handschuh, S., Kroner, P., Ossowski, S., Patel-Schneider, P. (eds.) Reasoning Web 2011. LNCS, vol. 6848, pp. 1–75. Springer, Heidelberg (2011)
2. Bayardo, R.J., Ma, Y., Srikant, R.: Scaling up all pairs similarity search. In: WWW, pp. 131–140 (2007)
3. Ben-David, D., Domany, T., Tarem, A.: Enterprise Data Classification Using Semantic Web Technologies. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part II. LNCS, vol. 6497, pp. 66–81. Springer, Heidelberg (2010)
4. Bleiholder, J., Naumann, F.: Data fusion. ACM Comput. Surv. 41(1), 1–41 (2008)
5. Botelho, F.C., Ziviani, N.: External perfect hashing for very large key sets. In: CIKM, pp. 653–662 (2007)
6. Elfeky, M.G., Elmagarmid, A.K., Verykios, V.S.: Tailor: A record linkage tool box. In: ICDE, pp. 17–28 (2002)
7. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering 19, 1–16 (2007)
8. Glaser, H., Millard, I.C., Sung, W.-K., Lee, S., Kim, P., You, B.-J.: Research on linked data and co-reference resolution. Technical report, University of Southampton (2009)
9. Isele, R., Jentzsch, A., Bizer, C.: Efficient Multidimensional Blocking for Link Discovery without losing Recall. In: WebDB (2011)

10. Isele, R., Bizer, C.: Learning Linkage Rules using Genetic Programming. In: Sixth International Ontology Matching Workshop (2011)
11. Köpcke, H., Thor, A., Rahm, E.: Comparative evaluation of entity resolution approaches with fever. Proc. VLDB Endow. 2(2), 1574–1577 (2009)
12. Lopez, V., Uren, V., Sabou, M.R., Motta, E.: Cross ontology query answering on the semantic web: an initial evaluation. In: K-CAP 2009, pp. 17–24. ACM, New York (2009)
13. Ngonga Ngomo, A.-C., Lyko, K.: EAGLE: Efficient Active Learning of Link Specifications Using Genetic Programming. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 149–163. Springer, Heidelberg (2012)
14. Ngonga Ngomo, A.-C.: A Time-Efficient Hybrid Approach to Link Discovery. In: Sixth International Ontology Matching Workshop (2011)
15. Ngonga Ngomo, A.-C., Auer, S.: LIMES - A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data. In: Proceedings of IJCAI (2011)
16. Ngonga Ngomo, A.-C., Lehmann, J., Auer, S., Höffner, K.: RAVEN – Active Learning of Link Specifications. In: Proceedings of OM@ISWC (2011)
17. Nikolov, A., d'Aquin, M., Motta, E.: Unsupervised Learning of Link Discovery Configuration. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 119–133. Springer, Heidelberg (2012)
18. Raimond, Y., Sutton, C., Sandler, M.: Automatic interlinking of music datasets on the semantic web. In: Proceedings of the 1st Workshop about Linked Data on the Web (2008)
19. Scharffe, F., Liu, Y., Zhou, C.: Rdf-ai: an architecture for rdf datasets matching, fusion and interlink. In: Proc. of IJCAI IR-KR Workshop (2009)
20. Schwarte, A., Haase, P., Hose, K., Schenkel, R., Schmidt, M.: FedX: Optimization Techniques for Federated Query Processing on Linked Data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 601–616. Springer, Heidelberg (2011)
21. Song, D., Heflin, J.: Automatically Generating Data Linkages Using a Domain-Independent Candidate Selection Approach. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 649–664. Springer, Heidelberg (2011)
22. Urbani, J., Kotoulas, S., Maassen, J., van Harmelen, F., Bal, H.: OWL Reasoning with WebPIE: Calculating the Closure of 100 Billion Triples. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part I. LNCS, vol. 6088, pp. 213–227. Springer, Heidelberg (2010)
23. Vaquero, L.M., Rodero-Merino, L., Buyya, R.: Dynamically scaling applications in the cloud. SIGCOMM Comput. Commun. Rev. 41, 45–52
24. Xiao, C., Wang, W., Lin, X., Yu, J.X.: Efficient similarity joins for near duplicate detection. In: WWW, pp. 131–140 (2008)