Multidimensional Orthogonal Range Search Using Tries

Lingke Bu [*] Bradford G. Nickerson [†]

Abstract

We present a novel k-dimensional range search algorithm for reporting all k-d rectangles from a set D of size n intersecting a query rectangle. Our algorithm uses $2k$-d tries to solve the orthogonal range search problem in k dimensions, requires linear space, and supports dynamic operations. An expected time analysis of the algorithm indicates it is competitive with the best known k-d range search algorithms when k is large (i.e. $k \approx \log n$).

1 Introduction

Range search represents an important class of problems that occur in computational geometry. Information retrieval problems can also formulated as range search in k dimensions (e.g. [1], [13]). Given a collection D of records, each containing several attributes or keys, an orthogonal range query asks for all records in D with key values each inside specified ranges. The range search problem can be interpreted geometrically by considering the record attributes as coordinates and the k values for each record as a point in a k-d coordinate space. Our definition for orthogonal range search is as follows:

Definition 1 For a data space R^k, where $k =$ the number of dimensions, orthogonal range search is defined as finding and reporting the set $HR, (|HR| = A, HR \subseteq D, D =$ the set of axis-aligned orthogonal data objects represented as rectangles, $|D| = n)$ of data intersecting a query rectangle $W = \{[L_1,H_1], [L_2,H_2], \cdots, [L_k,H_k]\}$, where $[L_j,H_j]$ represents a range for dimension j of the query rectangle, and $L_j < H_j$.

Lower bounds for range search were studied by Chazelle [7] [8], who showed that a sequence of n operations for insertion, deletion, and reporting points in a given range costs $\Omega(n(\log n)^k)$. Edelsbrunner introduced the d-fold rectangle tree to support orthogonal range search on k-d rectangles with time complexity $O(\log^{2k-1} n + A)$ [9], close to the lower bound, with storage $S(n, k) = O(nlog^{k-1}n)$. The time complexity analysis of range search for balanced k-d trees [4] shows that the search cost is $O(sn^{1-1/k} + A)$ for s of the k coordinates restricted to a subrange, and $(k - s)$ for the unspecified coordinates [12]. As pointed out in Flajolet and Puech [11], 1-d tries tend to be better balanced compared to 1-d search trees. For k-d search, this improved balance can lead to asymptotically smaller search times.

2 The data structure

Without loss of generality, we consider our problem defined on real $[0, 1]^k$ space and the following discussions are all based on unit space. We assume the coordinate value on each dimension in unit space can be represented in B bits.

Binary tries are data structures that use a binary representation of a key to store the key as a path in a tree [3]. Binary k-d tries use the principle of bit interleaving. Child nodes in a k-d trie cover $\frac{1}{2}$ the search space of their parent. We represent a rectangle as four coordinate values $(x^{\min}, x^{\max}, y^{\min}, y^{\max})$. The bit string for a rectangle is formed as follows: $b_0^{x^{\min}} b_0^{x^{\max}} b_0^{y^{\min}} b_0^{y^{\max}} b_1^{x^{\min}} b_1^{x^{\max}} \cdots b_{B-1}^{x^{\min}} b_{B-1}^{x^{\max}} b_{B-1}^{y^{\min}} b_{B-1}^{y^{\max}}$.

Extending the bit interleaving principle to k dimensions, we represent a k-d rectangle as $(x_j^{\min}, x_j^{\max})^k$, $\forall j \in \{1 \cdots k\}$, so the resultant bit string will be

$$b_0^{x_1^{\min}}\ b_0^{x_1^{\max}}\ b_0^{x_2^{\min}}\ b_0^{x_2^{\max}}\ b_0^{x_3^{\min}}\ b_0^{x_3^{\max}}\ \cdots\ b_0^{x_k^{\min}}\ b_0^{x_k^{\max}}$$
$$\ddots$$
$$b_{B-1}^{x_1^{\min}}\, b_{B-1}^{x_1^{\max}}\, b_{B-1}^{x_2^{\min}}\, b_{B-1}^{x_2^{\max}}\, b_{B-1}^{x_3^{\min}}\, b_{B-1}^{x_3^{\max}}\ \cdots\ b_{B-1}^{x_k^{\min}}\, b_{B-1}^{x_k^{\max}}.$$

Thus, a k-d rectangle can be represented as a $2k$-d point in a binary trie of height $2k$B.

A collection of rectangles in k-d space is denoted by $D = \{R_1, R_2, \cdots, R_n\}$, where n is the number of rectangles in the set. For the i^{th} rectangle $R_i \in D$, let $(x_{ij}^{\min}, x_{ij}^{\max})$ denote the j^{th} side of rectangle R_i, $1 \leq j \leq k$ and $1 \leq i \leq n$. We denote by T the $2k$-d trie constructed by inserting all the rectangles in D into an initially empty trie. Given a node u in T, we denote by T_u the subtree of T rooted at u. There are altogether n leaves in T. Every leaf is associated with one rectangle. The height of the trie (i.e. the length of the key) is $2k$B. At each node in the trie corresponding to a bit b, we traverse the left branch of the trie if $b = 0$ and we traverse the right branch if $b = 1$. After preprocessing all n rectangles in D, we obtain the trie T, which allows us to carry out an orthogonal range search.

[*] CARIS, 264 Rookwood Avenue, Fredericton, N.B., Canada, E3B 2M2. Email: lingke.bu@caris.com

[†] Department of Computer Science, University of New Brunswick, Fredericton, N.B., Canada, E3B 5A3. Email: bgn@unb.ca

3 Range search

A query rectangle $W = [L_1, H_1] \times [L_2, H_2] \times \cdots \times [L_k, H_k]$ is abbreviated as $[L_j, H_j]^k$. For a rectangle $R_i \in D$, the set of k rectangle sides is defined as $\{(x_{ij}^{\min}, x_{ij}^{\max}),$ $0 \le x_{ij}^{\min} \le x_{ij}^{\max} \le 1, \forall j \in \{1 \cdots k\}, i \in \{1 \cdots n\}$.

Remark Two rectangles R_1 and R_2 intersect if and only if their sides on every dimension in the data space intersect, i.e. $R_1 \cap R_2$ is true iff $\forall j \in \{1, \cdots, k\}$, $(x_{1j}^{\min}, x_{1j}^{\max}) \cap (x_{2j}^{\min}, x_{2j}^{\max})$ is true, which happens when $x_{1j}^{\min} \in [0, x_{2j}^{\max})$ and $x_{1j}^{\max} \in (x_{2j}^{\min}, 1]$.

This defines intersection strictly as an overlap in the sense of Allen [2] and Egenhofer [10]. Therefore, rectangle R_i intersects W iff $x_{ij}^{\min} \in [0, H_j)$ and $x_{ij}^{\max} \in (L_j, 1]$, $\forall j \in \{1 \cdots k\}$.

k-d orthogonal range search is performed using our $2k$-d trie for a query W. We use j as the index of the data space, $j \in \{1 \cdots k\}$, and we use p as the index for our problem space, $p \in \{1 \cdots 2k\}$. They are related as $j = \lceil p/2 \rceil$.

Definition 2 Each node in the trie T covers part of the $2k$-d space; that is, every node has a cover space defined as $NC^{2k} = [\mathcal{L}_p, \mathcal{U}_p]^{2k}$, $1 \le p \le 2k$. For a given query rectangle $W = [L_j, H_j]^k$, we obtain the query rectangle's cover space WC^{2k} and define it to be $WC^{2k} = [\mathcal{L}_p, \mathcal{U}_p]^{2k}$,$\mathcal{L}_p = 0$, $\mathcal{U}_p = H_j - \epsilon$, when $p \bmod 2 = 1$; $\mathcal{L}_p = L_j + \epsilon$, $\mathcal{U}_p = 1$, when $p \bmod 2 = 0$, where $1 \le p \le 2k$, $j = \lceil p/2 \rceil$, and ϵ is a small value to guarantee open intervals.

On the p^{th} dimension, there are three types of relationship of WC^p with NC^p, which we call BLACK, GREY, and WHITE. Figure 1 illustrates the three colors for a node's cover on dimension p. Dashed lines are used for WC^p and solid lines for NC^p. WHITE indicates when the trie can be pruned. BLACK relationships occurring $2k$ times contiguously indicates that all rectangles in the subtree intersect W. GREY indicates the trie must be searched further.

Definition 3 If, on all $2k$ dimensions, the cover space relationship satisfies $WC^p \cap NC^p$ = BLACK, $\forall p \in$ $\{1,2,\cdots,2k\}$, then the node in the trie is black. If the cover space relationship satisfies $\exists p \in \{1, 2, \cdots, 2k\}$, such that $WC^p \cap NC^p$ =WHITE, then the node in the trie is white. All other nodes are grey nodes.

The range search algorithm (see Figure 2) traverses from the root of trie T down to its leaves. Arrays $\mathcal{L}$ and $\mathcal{U}$ store the lower and upper bounds of node T's cover space on $2k$ dimensions. At the root, level $\ell = 0$. For the root, the cover space NC^{2k} has $\mathcal{L}_p = 0$ and $\mathcal{U}_p = 1$, $\forall p \in \{1, 2, \cdots, 2k\}$. The cover space is split on the p^{th} dimension as we move down, $p = \ell \, mod \, 2k$, $\forall \, \ell \in \{0, 1, \cdots, (2k\mathrm{B}{-}1)\}$. If on the p^{th} dimension, a

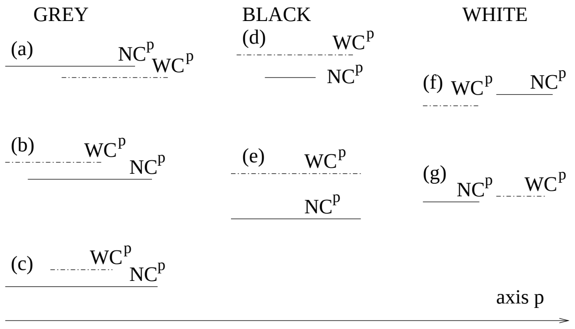

Figure 1: GREY ((a), (b), and (c)), BLACK ((d) and (e)), WHITE ((f) and (g)) relationships of a trie node cover space NC^p to a query rectangle cover space WC^p in dimension p of the $2k$-d problem space.

```
RANGESEARCH($T, \ell, \mathcal{L}, \mathcal{U}, RI, W, List$)
if $T$ = NIL or $\ell > 2kB$
   then return
$p \leftarrow (\ell - 1) \bmod (2k)$
$RI[p] \leftarrow$ INRANGE($\mathcal{L}[p], \mathcal{U}[p], \ell, W$)
if $RI[p]$ is grey
   then $\ell \leftarrow \ell + 1$
        $p \leftarrow \ell \bmod (2k)$
        if $left[T] \ne$ NIL
           then $\mathcal{U}[p] \leftarrow (\mathcal{L}[p] + \mathcal{U}[p])/2$
                RANGESEARCH($left[T], \ell, \mathcal{L}, \mathcal{U},$
                $RI, W, List$)
        if $right[T] \ne$ NIL
           then $\mathcal{L}[p] \leftarrow (\mathcal{L}[p] + \mathcal{U}[p])/2 + \epsilon$
                RANGESEARCH($right[T], \ell, \mathcal{L}, \mathcal{U},$
                $RI, W, List$)
   else if $RI[p]$ is black and COLOR($RI$) is black
           then COLLECT($T, List$)
```

Figure 2: Pseudo-code for the k-d orthogonal range search algorithm. INRANGE($\mathcal{L}[p], \mathcal{U}[p], \ell, W$) is a function to decide the color of NC^{2k} and WC^{2k} relationships for node T. ϵ is a small value to guarantee T's left and right children's cover spaces do not share any points.

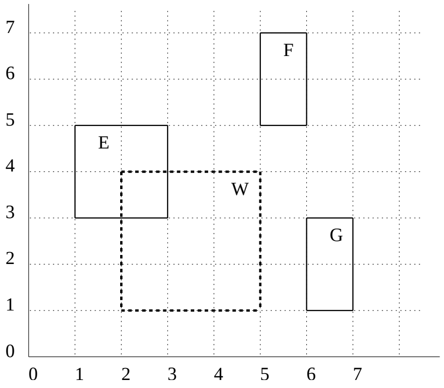

Figure 3: Example of three rectangles E, F, and G with a query rectangle W and number of data bits B = 3.

parent node T has cover space $[\mathcal{L}_p, \mathcal{U}_p]$, then T's left child's cover space is $[\mathcal{L}_p, (\mathcal{L}_p + \mathcal{U}_p)/2]$ and T's right child's cover space is $((\mathcal{L}_p + \mathcal{U}_p)/2, \mathcal{U}_p]$. Comparing a node's cover space NC^{2k} (stored in $\mathcal{L}$ and $\mathcal{U}$) with $[0, H_j)$ and $(L_j, 1]$, if one of the $2k$ ranges falls outside (as determined by the InRange function), we encounter a white node and the search need not check any subtrees of T. All nodes in the subtree of a white node are white nodes.

If all $2k$ ranges fall within NC^{2k} at some node T_u, then all rectangles in the subtree attached to T_u intersect W and are collected into a *List* for reporting. Array RI of size $2k$ (initialized to store all grey values) keeps track of the color of the NC^p to WC^p relationship for T and ancestors of T. If the NC^p to WC^p relationship for node T is BLACK, and the same is true for all $2k - 1$ ancestors of T, then node T is black as defined in Definition 3. Function Color(RI) checks that all $2k$ relationships are BLACK. All nodes in the subtree of a black node are black nodes. Collect(T, *List*) traverses the subtree of T adding each found rectangle to *List*.

Figures 3 and 4 depict an example of the range search algorithm, illustrated on an integer domain for clarity. Figure 4 is the trie for the data shown in Figure 3. In Figure 4, empty circles represent white nodes. A black-filled circle represents a black node, which means all rectangles represented by leaves inside the subtree attached to the black node intersect W. A grey-filled circle represents a grey node. For query rectangle $W = [2, 5] \times [1, 4]$, $k = 2$, the query rectangle's cover space is $WC^4 = [0, 5)(2, 7][0, 4)(1, 7]$. The rectangle denoted as G= $[6, 7] \times [1, 3]$ has its cover space NC^4 listed along the right side of Figure 4. The small letters b, w and g to the right of a node indicate the result of the InRange call giving the color of the NC^p to WC^p relationship. The trie is divided into B= 3 bands, each of height $2k = 4$ (see Figure 4). For rectangle G, traversal of T during the 2-d range search for rectangles intersecting W stopped at Band 1 when the first white node was encountered. For rectangle F, a white node is found in Band 0, and for rectangle E, the intersection with W is determined in Band 2.

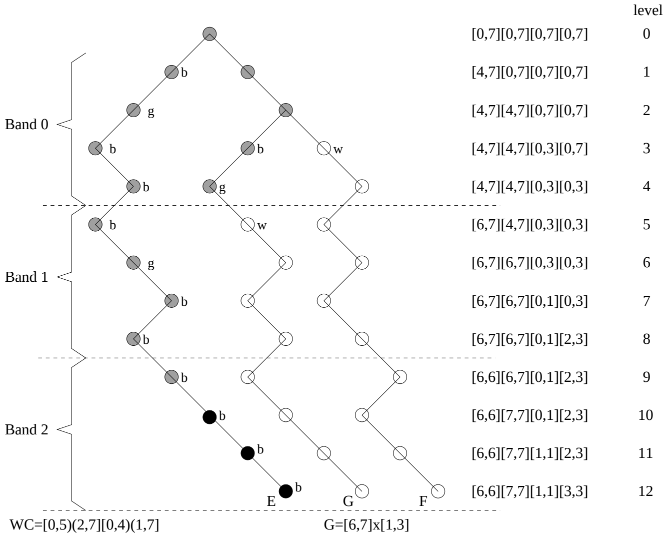

Figure 4: Example of a binary 4-d trie for the 2-d data of Figure 3. The list of 8-tuples near the right hand side is the cover space NC^4 of each node on the trie path representing rectangle G.

4 Analysis

We adapt the approach used in [6] which in turn, uses Theorem 2 of [11].

Proposition 1 *Given a binary trie T of τ nodes containing a set of k-d input rectangles $D = \{R_1, \cdots, R_n\}$, assuming input data set D and query rectangle q satisfy the uniform probabilistic model, $q= (q_1, q_2, \cdots, q_{2k})$, $S \subset \{1, 2, \cdots, 2k\}$, the cost of partial match retrieval $Q_S(n, k)$ measured by the number of nodes traversed in trie T is*

$$Q_S(n, k) = E\{\Sigma_{t=1}^{\tau} \prod_{p \in S} |NC_t^p|\}.$$

Proof The probability that a node in trie T will be visited is determined by the volume of every node's cover space in the space $[0, 1]$. ∎

The time complexity can be determined by computing the number of grey and black nodes in the trie built from input data D. We have the following equation:

$$Q(n, k) = \Sigma_{t=1}^{\tau} 1_{[node_t \in GN \cup BN]}$$

where we use $1_{[A]}$ as the characteristic function of the event A.

Lemma 1 $E\{\Sigma_{t=1}^{\tau} \prod_{p=1}^{2k} |NC_t^p|\} = O(\log_2 n)$.

Theorem 1 *Given a binary trie T containing a set of k-d input rectangles $D = \{R_1, R_2, \cdots, R_n\}$, R_i with i.i.d. random variable center c_i on $[0,1]^k$, and with i.i.d. random variable side length d_i distributed on $[0,1]^k$, consider a random orthogonal range search with query rectangle W with center at Z which is uniformly distributed on $[0,1]^k$, and independent of the centers of D, and with size $\Delta_1 \times \Delta_2 \times \cdots \times \Delta_k$ which are also i.i.d. random variables on $[0,1]^k$. The expected orthogonal range search time $E\{Q(n,k)\} \leq$*

$$\Sigma_{S\subset\{1,\cdots,2k\}}(\textstyle\prod_{p\notin S}|WC^p|)(\gamma(\frac{1}{2k}\log_2 n)n^{1-\frac{s}{2k}} + O(1)) + O(\log_2 n),$$

where γ_u is a periodic function of u with period 1, small amplitude, and mean value

$$\gamma_0 = -\frac{s}{4k^2\log 2}\Gamma(\frac{s}{2k}-1)\,\Sigma_{\ell=0}^{2k-1}(\delta_1\,\delta_2\,\cdots\,\delta_\ell)2^{-\ell(1-s/2k)}$$

with $\delta_\ell = 1$, if the ℓ^{th} attribute of the query is specified, and $\delta_\ell = 2$ if it is unspecified.

Proof $E\{Q(n,k)\} = E\{\,\Sigma_{t=1}^{\tau}\, 1_{[node_t \in GN \cup BN]}\}$. This calculation includes the reporting time for collection of the subtree of black nodes which arises during the traversal. The probability that a node is black or grey is given as: $Pr(node_i \in GN \cup BN) \leq \prod_{p=1}^{2k}(|NC^p| + |WC^p|)$. The probability for query rectangle W's cover space WC to intersect a node's cover space NC is the probability that Z_j, the center of W, is within distance $\frac{\Delta_j}{2}$ of NC^j. This probability is bounded by the volume of NC expanded by Δ_j in the j^{th} dimension, $\forall j \in \{1,\cdots,k\}$. There are two cases. On the left side of the j^{th} dimension, $|WC_j^{\min}| = |WC^p| = |\,[0,H_j)\,| = H_j = Z_j + \frac{\Delta_j}{2}$, and $p \bmod 2 = 1$. On the right side of the j^{th} dimension, $|WC_j^{\max}| = |WC^p| = |\,(L_j,1]\,| = 1 - L_j = 1-(Z_j - \frac{\Delta_j}{2}) = 1 - Z_j + \frac{\Delta_j}{2}$, and $p \bmod 2 = 0$. We have

$$\begin{aligned}
&E\{Q(n,k)\} \leq E\{\Sigma_{t=1}^{\tau}\textstyle\prod_{p=1}^{2k}(|WC^p| + |NC_t^p|)\\
&= \Sigma_{S\subseteq\{1,\cdots,2k\}}(\textstyle\prod_{p\notin S}|WC^p|)E\{\Sigma_{t=1}^{\tau}\prod_{p\in S}|NC_t^p|\}\\
&= \Sigma_{S\subset\{1,\cdots,2k\}}(\textstyle\prod_{p\notin S}|WC^p|)E\{\Sigma_{t=1}^{\tau}\prod_{p\in S}|NC_t^p|\}\\
&\quad +\Sigma_{S=\{1,\cdots,2k\}}(\textstyle\prod_{p\notin S}|WC^p|)E\{\Sigma_{t=1}^{\tau}\prod_{p=1}^{2k}|NC_t^p|\}\\
&= \Sigma_{S\subset\{1,\cdots,2k\}}(\textstyle\prod_{p\notin S}|WC^p|)E\{\Sigma_{t=1}^{\tau}\prod_{p\in S}|NC_t^p|\}\\
&\quad +E\{\Sigma_{t=1}^{\tau}\textstyle\prod_{p=1}^{2k}|NC_t^p|\}
\end{aligned}$$

Using Theorem 2 from [11] for our $2k$-d trie and Proposition 1, we obtain $E\{Q(n,k)\} \leq$

$$\Sigma_{S\subset\{1,\cdots,2k\}}(\textstyle\prod_{p\notin S}|WC^p|)(\gamma(\frac{1}{2k}\log_2 n)n^{1-\frac{s}{2k}} + O(1)\,) + E\{\Sigma_{t=1}^{\tau}\prod_{p=1}^{2k}|NC_t^p|\},$$

and by Lemma 1, we obtain $E\{Q(n,k)\} \leq$

$$\Sigma_{S\subset\{1,\cdots,2k\}}(\textstyle\prod_{p\notin S}|WC^p|)(\gamma(\frac{1}{2k}\log_2 n)n^{1-\frac{s}{2k}} + O(1)\,) + O(\log_2 n). \quad \blacksquare$$

A similar approach yields a lower bound of $E\{Q(n,k)\} \geq$

$$\Sigma_{S\subset\{1,\cdots,2k\}}(\textstyle\prod_{p\notin S}\frac{|WC^p|}{2})(\gamma(\frac{1}{2k}\log_2 n)n^{1-\frac{s}{2k}} + O(1)) - C,$$

where C is constant value determined by k [5].

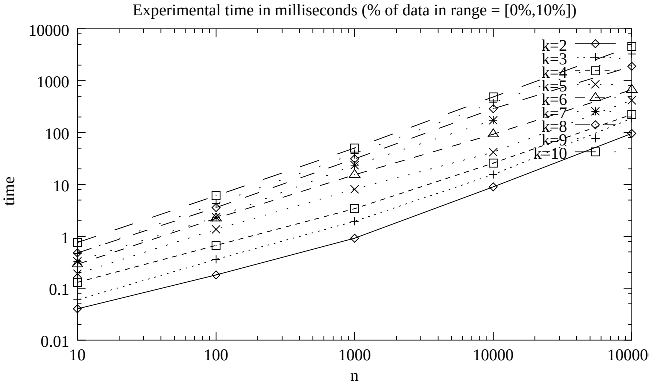

Figure 5: Experimental time for percent of data in range = [0%, 10%].

5 Discussion and Experimental Results

Based on the lower and upper bound shown above, we can write the expected range search time as

$$E\{Q(n,k)\} = c_1\textstyle\prod_{p=1}^{2k}|WC^p|n + c_2\Sigma_{S\subset\{1,\cdots,2k\},0<|S|<2k}(\prod_{p\notin S}|WC^p|)\gamma(\frac{\log_2 n}{2k})n^{1-\frac{s}{2k}} + O(\log_2 n)$$

where c_1 and c_2 are constant values. The first term accounts for the number of rectangles returned by the orthogonal range search. The third term arises from the height of the trie which is unavoidable. The second term dominates, and arises from the number of grey nodes checked to determine intersection with W. For $k = 2$ and $k = 3$, we have determined that $E\{Q(n,k)\}$ behaves as $O(A + n^\alpha)$ for $0.5 \leq \alpha < 1$ [5]. We conjecture that $E\{Q(n,k)\} \approx O(A + n^\alpha)$ with $\alpha < 1$ holds for $k > 3$, but this remains to be shown.

Experimental validation of our approach was performed using randomly generated k-d rectangles for $2 \leq k \leq 10$ and $10 \leq n \leq 100000$. Figure 5 shows the time taken for k-d range search when A $\leq 0.1n$. For $k = 10$, Figure 5 shows time increasing at a rate of approximately $n^{0.9}$, which matches our theoretical findings.

6 Acknowledgement

The Natural Sciences and Engineering Research Council (NSERC) of Canada and the UNB Faculty of Computer Science provided financial support for the research reported here. We are grateful for their support.

References

[1] P. K. Agarwal. Range searching. In J. E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, pages 575–598, Boca Raton, NY, 1997. CRC Press.

[2] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[3] R. A. Baeza-Yates and G. H. Gonnet. Fast text searching for regular expressions or automaton searching on tries. *Jounal of the ACM*, 43(6):915–936, 1996.

[4] J. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, September 1975.

[5] L. Bu. Tries for spatial data range search. Technical Report TR03-160, University of New Brunswick, Fredericton, NB, Canada, February 2003.

[6] P. Chanzy, L. Devroye, and C. Zamora-Cura. Analysis of range search for random k-d trees. *Acta Informatica*, 37(4/5):355–383, 2001.

[7] B. Chazelle. Lower bounds for orthogonal range search: Ii. the arithmatic model. *Journal of the ACM*, 37(3):439–463, July 1990.

[8] B. Chazelle. Lower bounds for orthogonal range searching: I. the reporting case. *Journal of the ACM*, 37(2):200–212, April 1990.

[9] H. Edelsbrunner. A new approach to rectangle intersection part i. *Int. J. Computer Mathematics*, 13:209–219, 1983.

[10] M. Egenhofer. Spatial sql: A query and presentation language. *IEEE Transactions on Knowledge and Data Engineering*, 6(1):86–95, 1994.

[11] P. Flajolet and C. Puech. Partial match retrieval on multidimensional data. *Journal of the Association for Computing Machinery*, 33(2):371–407, April 1986.

[12] D. T. Lee and C. K. Wong. Worst-case analysis for region and partial region searches in multidimensional binary search trees and balanced quad trees. *Acta Informatica*, 9:23–29, 1977.

[13] F. F. Yao. Range search. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A: Algorithms and Complexity, pages 370–374, Amsterdam, 1990. Elsevier.