Orthogonal Range Search using a Distributed Computing Model

Pouya Bisadi* Bradford G. Nickerson†

Abstract

We present a novel approach for distributed orthogonal range search on a set of N points stored on n nodes. The non-redundant rainbow skip graph [Goodrich et al [12] is used to coordinate message passing among nodes. We show that the maximum number of levels L in such a graph is $L = W(n \ln 2)/\ln 2$, where W is the lambertW function. Experimental validation is performed using 24 nodes, with $N = 2.4 \times 10^7$ points distributed in a uniform random fashion in a $[0, 1]^2$ space. Each node stores an equal number of points, with the distribution of points among nodes controlled by point x coordinates. The experiments were implemented using the Message Passing Interface (MPI) communication model running on a high performance computer cluster. Our results show that the expected number of messages required to answer a point query originating from any node matches the theoretical bound of $\Theta(\log n)$ messages.

1 Introduction

We wish to preprocess a set S of N points into a data structure, so that for an axis aligned rectangle range query γ, the points in $S \cap \gamma$ can be reported or counted efficiently [3]. Increased reliability arises if multiple copies of the data are stored in multiple locations. In addition, increased flexibility (e.g. for access control) can arise for data maintenance by different organizations at each of the different locations. Distributed data structures are useful in these settings.

The performance measure of a data structure is related to the model of computation in which it is defined. Range search complexity is the number of memory accesses in the *RAM* and *pointer-machine* models, and the number of I/Os in the *I/O* model [3]. In the *distributed computing* model it is assumed that the cost of sending a message is higher than the cost of an I/O, so the number of messages exchanged when answering a query becomes the complexity measure.

For the last four decades, orthogonal range search was and continues to be one of the most important problems in data structures, and many people worked on it [10]. The most efficient data structure for worst case 2-dimensional range queries in the *RAM* model is a modified version of the layered range tree, described by Chazelle [6]. Chazelle succeeded in improving the storage to $O(\frac{N \log N}{\log \log N})$ with query time of $O(\log N + k)$, for k points reported in range. Chazelle [7, 8] also proved that this time and space bound are optimal in the worst case. Arge et al [4] provided a two dimensional I/O-efficient structure for general range searching which occupies $O(\frac{N}{B} \frac{\log(N/B)}{\log \log_B N})$ disk blocks and answers queries in $O(\log_B N + \frac{T}{B})$ I/Os, which are optimal in the worst case. Afshani et al [2] presented a space optimal pointer machine data structure for 3-d orthogonal range reporting that answers queries in $O(\log N + k)$ time.

None of these optimum solutions considers a distributed model where reducing node congestion and improving fault tolerance are important. Sridhar et. al. [18] presented a parallel algorithm to report the in-range set of points in a rectangular range-search in $O(\log N)$ time, with $O(\log^2 N)$ processors on an EREW-PRAM model (Exclusive-Read-Exclusive-Write Parallel Random Access Model). A shared memory model presented by Sridhar et. al. [18] can be used in a network, but they did not consider fault tolerance and reliability because their model is for a single machine with multiple processors. Hash functions do not preserve the order of keys and methods like Chord [19] and CAN [17] which use distributed hash tables (DHT) are good for lookup (single point) queries. Aspnes and G. Shah [5] have presented a distributed data structure which supports range queries along single atribute 1-D. However, the Skip Graph stores $\log n$ pointers for each node and asigning one point to each node requires $n \log n$ space which is not practical.

For range search using a distributed model, the basic idea is to divide S into n subsets (maybe with overlap) and to distribute them among n nodes. Each one of these nodes can be a representative of a host in a physical network. Generally, in a distributed data structure each node has a key (or name) m and an address a (like an IP address), so a pointer to a node is a pair (m, a). A lookup query in these data structures can be interpreted as "What is the address of the node that has the key m?". In the case of range search, the question is a bit different; i.e. "What are the addresses of the nodes storing points intersecting with the range query γ?". We would like to find the answer to this question by sending the minimum number of messages. The query can

*Faculty of Computer Science, University of New Brunswick, j3ngr@unb.ca

†Faculty of Computer Science, University of New Brunswick, bgn@unb.ca

be issued from any node u among the n nodes. Once the destination nodes are found, within-node search can be performed using e.g. an I/O-efficient data structure supporting range search.

Many distributed data structures have been presented for general applications in a distributed model. The skip graph [5], family tree [20] and rainbow skip graph [12] are a few of them. Zatloukal and Harvey [20] use a modified SkipNet [15] to construct a structure they call the family tree, achieving $O(\log n)$ expected messages for search and update, while restricting required space (number of stored pointers) for each node to be $O(1)$, which is optimal.

Goodrich et al [12] presented a peer-to-peer data structure called the rainbow skip graph that achieves high fault-tolerance, constant-sized nodes, and fast update and query times for ordered data. In this paper, a non-redundant rainbow skip graph [12] is used for routing purposes.

2 Our Results

We utilized the non-redundant rainbow skip graph to implement an orthogonal range search structure. To our knowledge, this is the first implementation of this routing data structure for range search on spatial data. In this data structure, the cost of search is independent of the query issuer. Our experimental results support this statement. We prove that the maximum number of levels in a rainbow skip graph is $L = \frac{W(n \ln 2)}{\ln 2}$ where W is the lambertW [9] function and n is the number of nodes in the non-redundant rainbow skip graph [12].

3 Data Structure and Search Algorithm

3.1 Non-Redundant Rainbow Skip Graphs

A skip graph [5] is a distributed data structure which consists of skip lists [16]. It has all the functionality of a balanced tree in a distributed system and its algorithms for insertion and deletion are the same as a skip list (see Figure 1). The search algorithm in a skip graph is almost the same as searching a skip list. The main difference is that every node is in every level of a skip graph.

To implement a distributed orthogonal range search data structure, a non-redundant rainbow skip graph is used because it provides all the features necessary for a general purpose peer–to–peer data structure. Based on the Goodrich et al [12] definition, a non-redundant rainbow skip graph on n nodes consists of a skip graph [5] on $\Theta(\frac{n}{\log n})$ supernodes, where a supernode consists of $\Theta(\log n)$ nodes that are maintained in a doubly-linked list called the core list of the supernode. As explained in the next section, $\Theta(\log n)$ is not the optimum size of supernodes. The keys of the nodes of each supernode are

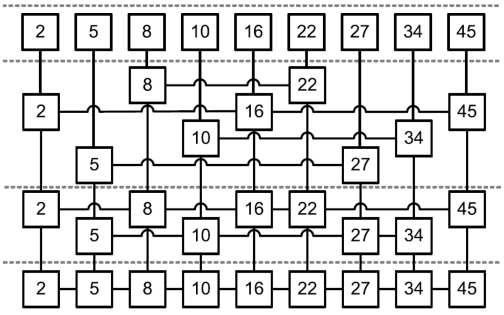

Figure 1: An example of skip graph from [11]. The levels are separated by dashed lines.

a contiguous subsequence of the ordered sequence of all keys. The smallest key of each supernode V is referred to as the key of V, and the skip graph is defined over these keys. For each supernode V, a different member of V is associated with each level i of the skip graph, and this member is the level i *representative of* V, which is denoted as V_i. The level i list to which V belongs contains V_i. These lists of the skip graph are called the level lists. V_i, which can be chosen arbitrarily from among the elements of V, is connected to V_{i+1} and V_{i-1}, which respectively are called the parent and child of V_i. These vertical connections form another linked list associated with supernode V that is referred as the *tower list* of V. Each supernode has one tower list. Each element of a supernode is a member of at most three lists; the core list, the tower list, and one level list. Figure 2 shows a rainbow skip graph created over the same data shown in Figure 1. For example, to search for the key 19 from node 2, nodes 2, 5, 22, 16, 27, 16, 22 are visited to find that key 19 is not in the graph.

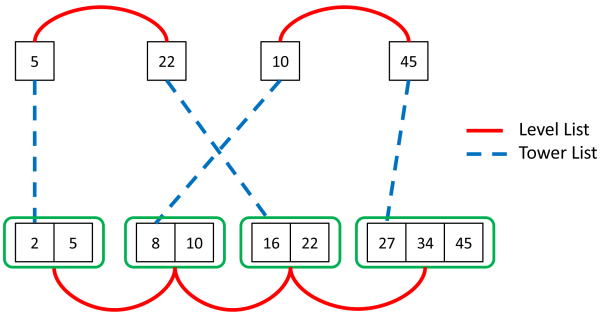

Figure 2: Nine nodes are grouped into four supernodes to create a rainbow skip graph with two levels and four tower lists.

3.2 Supernode Size

Lemma 1 *Considering that a node cannot be in multiple level lists, the maximum number of levels for a non-*

redundant rainbow skip graph with n nodes is when the size of all supernodes are equal to the number of levels.

Proof. A non-redundant rainbow skip graph is the result of creating a skip graph from supernodes (groups of nodes) with each supernode belongs to many level lists. There are two constraints: First, at each level i a different member of a supernode is the supernode representative V_i in that level, and a supernode cannot be a member of level $i+1$ if it is not a member of level i. If the number of nodes in a supernode is less than the number of levels in the skip graph, this supernode cannot be in all the levels.

As the second constraint, the number of supernodes is $\frac{n}{|V|}$ and the number of levels is related to the number of supernodes. Therefore the number of levels $\log_2 \frac{n}{|V|}$ is related to size of the supernodes. If $|V|$ goes up, the number of levels goes down.

Therefore, the number of levels is always the minumim of $|V|$ and $\log_2 \frac{n}{|V|}$ (see Figure 3). □

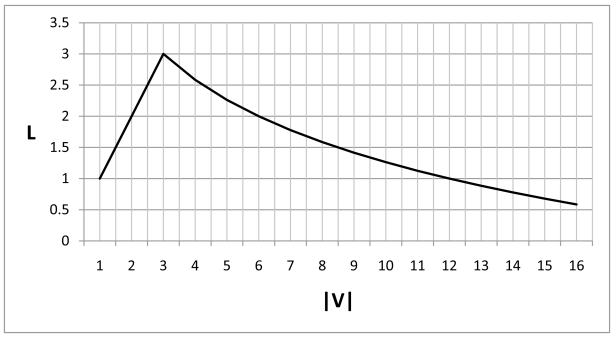

Figure 3: Relationship of the size of supernodes $|V|$ and number of levels L when $n = 24$.

Theorem 2 *The maximum number of levels L for a non-redundant rainbow skip graph is*

$$L = \frac{W(n \ln 2)}{\ln 2} \tag{1}$$

where n is the number of nodes and W is the lambertW function which is the inverse function of $z = We^W$.

Proof. When every node is a member of a level list, there is no node that is just in the core list of its supernode. In other words, to maximize the value of L from Lemma 1, the size of supernode V and the number of levels L should be equal. Such a rainbow skip graph has L levels and each of its supernodes has L members as their level representative. Consequently, there are 2^L supernodes with a size of L. As a result, the number of all nodes is $L \times 2^L = n$. Solving this equation for L gives the value for n which is based on the *lambertW* [9] function shown in equation (1). □

3.3 Data Distribution

Routing in the non-redundant rainbow skip graph requires a set of keys having a total order relation. Having a total order relation [13] on the set of keys makes it possible for each node to determine whether the requested node of a query is one of the successor or predecessor nodes without knowledge about the whole data structure. Therefore, each node can route the received query message in the correct direction.

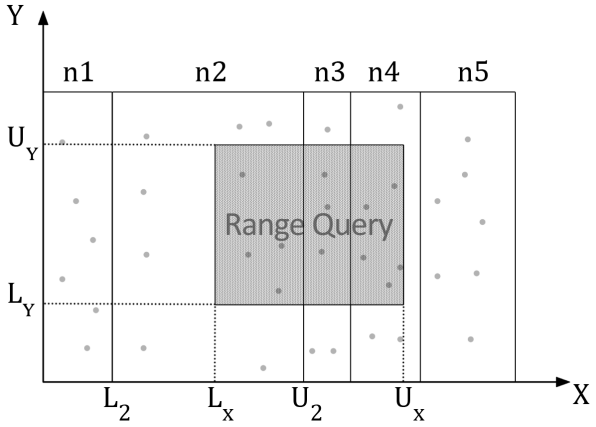

Figure 4: A distribution of a 2D space among 5 nodes. The hatched area is a rectangular query. Also, the lower and upper bounds L_2 and U_2 of the second node region are labelled.

To use the non-redundant rainbow skip graph, the entire space must be split into regions in such a way that a total order binary relation [13] (here denoted by $\leq$) is definable on this set of regions. The simplest distribution is splitting points based on one of their coordinates (e.g. x) which is shown in Figure 4. For example, a suitable key for the presented distribution in Figure 4 can be $m(L_i, U_i)$ where i is the node number, and L_i and U_i are the lower and upper x coordinate bounds of node region, respectively. In this way, the key set has a total order based on the x coordinate.

In orthogonal range search, if a point is not in the query range in one dimension, its coordinate value (in that dimension) is either greater or lower than all the points in the query range. As the distribution of points is based on one dimension, if the query range has intersection with nodes $i-1$ and $i+1$, it has intersection with node i for sure. Consequently, having the address of node i whose region intersects the query $\gamma([L_x, U_x], [L_y, U_y])$, node $i-1$ should be checked too unless $L_i \leq L_x \leq U_i$ and node $i+1$ should be checked unless $L_i \leq U_x \leq U_i$.

3.4 Range Search

We created a non-redundant rainbow skip graph using the lower bound of each node region as its key. The rainbow skip graph normal routing algorithm is used to find the node whose region covers L_x. In order to answer a query $\gamma([L_x, U_x], [L_y, U_y])$ from node u, the rainbow skip graph search algorithm [12] is used to find the node that stores the lower bound of γ. First we find the top level representative of the supernode of node u. Then, by a standard skip graph [5] search, we find the supernode whose key (x lower bound) is the maximum key lower than L_x. The next step is performing a linear scan through the core list to find the first node that stores points in range query γ. Then, this node reports the points to u (the query issuer) and passes the query to its successor node if the upper bound of the query (U_x) is outside its region; the next node does the same. Each of these steps (except the reporting part) requires $O(\log n)$ messages, then the complexity of point search using the non-redundant rainbow skip graph is $O(\log n)$ messages.

In the distributed computing model, the notion of failed nodes is important. If no messages are received in response to a query, we assume the nodes intersecting the query range have failed. In the worst case, a query intersects all n regions, but finds no points in range. A message indicating this empty set is required at each queried node. This leads to $O(n)$ messages for range search on a set of N points distributed on n nodes of a non-redundant rainbow skip graph. The same worst case search complexity holds if $k > 0$ and we assume $O(1)$ messages can hold the k points reported in range.

4 Experimental Validation

To test this data structure, 2.4×10^7 two dimensional points drawn from a uniform random distribution $\in [0, 1]^2$ are distributed based on their x coordinate among 24 nodes. Therefore each node covered around 4% of the whole area. Around 2,400 queries (see Figure 6) were randomly generated such that:

- query center $(x_i, y_i) \in [0.1, 0.9]^2$
- lower bound $(x_L, y_L) = (x_i, y_i) - (\Delta x_i, \Delta y_i)$
- upper bound $(x_U, y_U) = (x_i, y_i) + (\Delta x_i, \Delta y_i)$
- Δx_i and Δy_i are uniform random $\in [0.0, 0.1]$

The experiments used the Message Passing Interface [14] (MPI) on the Atlantic Computational Excellence Network [1](ACEnet). Figure 5 shows the implemented data structure. This structure consists of three levels and each supernode has 3 nodes which are its representatives in different levels. Notice that in this figure, levels are shown by different dashed lines for their links.

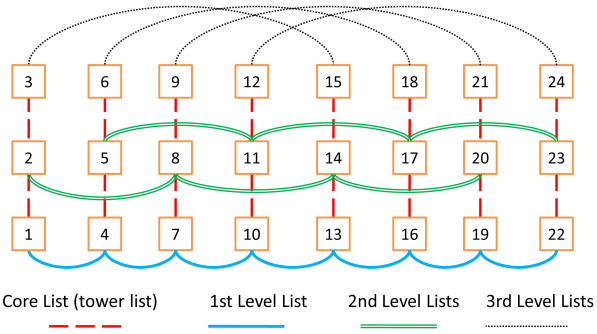

Figure 5: The rainbow skip graph which is used for experimental validation. Level lists are shown by curved lines. There are 8 supernodes, each containing three nodes.

As the tower list is exactly the same as the core list, the maximum number of connections for each node is 4. The program uses 25 slots, the first one (index 0) as the test harness and the rest as the nodes. Node 0 randomly sent commands for issuing queries to nodes and collected the data. We made the simplifying assumption that messages are big enough for nodes to report back all the points in their intersection with γ in one message.

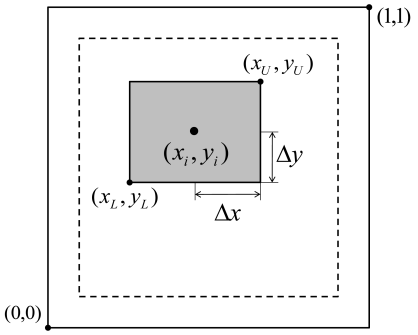

Figure 6: The center and size of query rectangles are generated from a uniform random distribution function.

As shown in Figure 7, each query was answered by passing an average of 13 messages through the network, with the number of passed messages averaging no more than 16 and no less than 11.

Figure 8 and 9 show two graphs representing the relationship between query cost and size of query rectangles. Each dot in figures 8- 11 represents one of the 2,400 range query results. As expected, the number of messages is independent of the height of query rectangles because each node covers the height of the total area. The cost for responding to queries rises linearly with increasing query rectangle width.

As shown in Figure 10, the query cost is from 1 to 24 messages when the area of the query is small (see

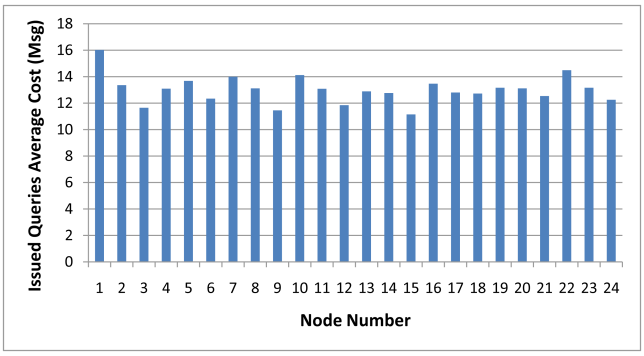

Figure 7: The average number of messages for answering queries issued from each node.

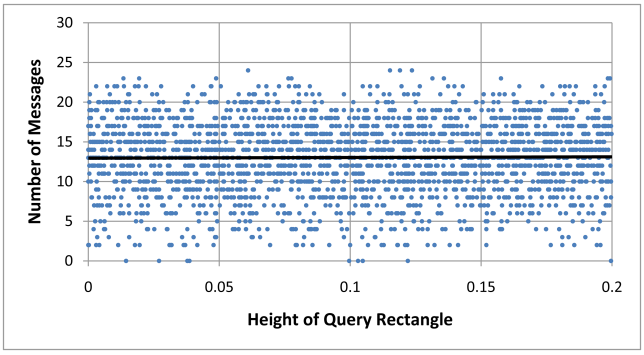

Figure 8: The number of messages for answering queries is not related to the height of query rectangles. The black line is a linear trendline for the number of messages.

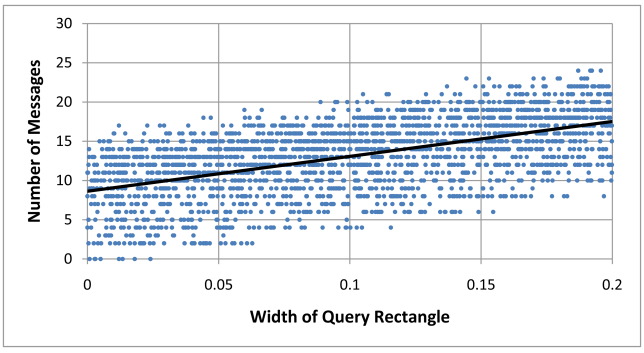

Figure 9: Number of messages increases linearly with increasing query rectangle width. The black line is a linear trendline for the number of messages.

Figure 10). The maximum messages arising on the left of Figure 10 is high as a very thin horizontal query rectangle requires a lot of message passing to answer even if it has a relatively small area.

The maximum cost of queries with aspect ratio less than one is lower than the queries with aspect ratio $>$ 1 (see Figure 11). There is a higher probability for low

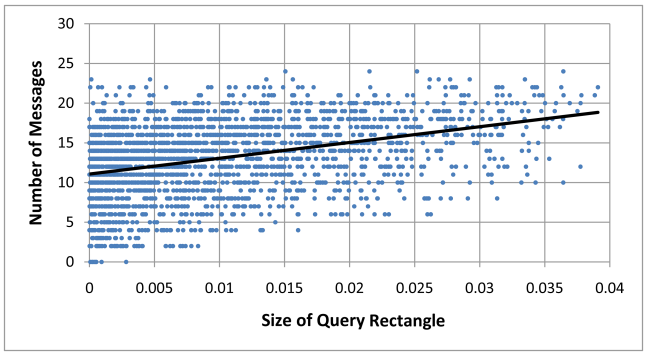

Figure 10: Number of messages increases with growth of the query size since the size is dependent on query width. The black line is a linear trendline for the number of messages.

aspect ratio query rectangles to have a smaller width and thus intersect with fewer node regions.

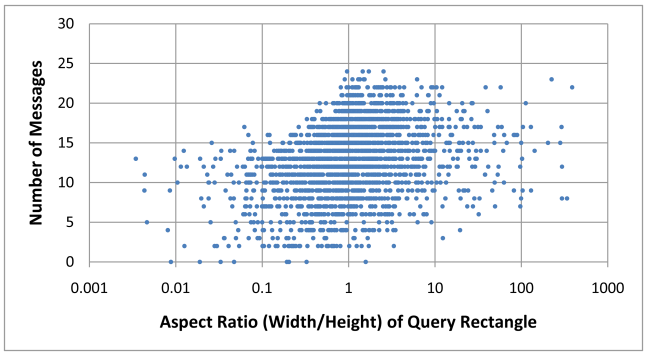

Figure 11: Number of messages rises linearly with increasing query rectangle aspect ratio.

5 Simulation

We performed a simulation of the performance of the non-redundant rainbow skip graph for $n = 24$, 36, 48, 64, 80 and 96. The results are shown in Figure 12. The average cost to find the first node in range corresponds to a point search issued from any node. As Figure 12 shows, a point search costs $< 2 \log_2 n$ messages, which matches the expected number of messages reported in [12]. Figure 12 also shows the average cost that includes messages sent to report the points in range.

6 Conclusion

The aim of using a distributed model for orthogonal range search is to provide reliability, flexibility and robustness to the data structure. In this paper we presented a novel approach for distributed orthogonal range search using the non-redundant rainbow skip graph.

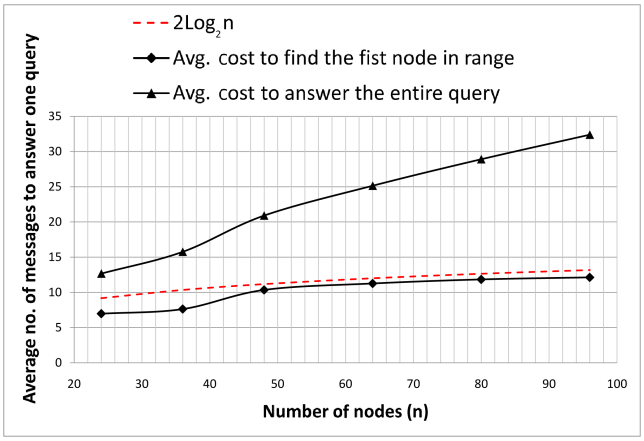

Figure 12: Average number of messages to answer one query based on the number of nodes.

We proved that the maximum number of levels in a non-redundant rainbow skip graph occurs when the size of each supernode is equal to number of levels. The maximum number of levels $L = \frac{W(n \log 2)}{\log 2}$. We also showed experimentally that a point search cost requires $\Theta(\log n)$ messages, which matches the expected results in Goodrich et al [12]. The experimental results showed that distributed range search cost using the non-redundant rainbow skip graph was independent of which node issued the query. In addition we showed that the number of messages required to answer a range query increased linearly with increasing query rectangle width.

It remains to determine the optimum size of a supernode in the non-redundant rainbow skip graph such that the number of messages passed to answer a range query is minimized.

7 Acknowledgements

The authors would like to acknowledge the support of the Natural Sciences and Engineering Research Council (NSERC) of Canada and the UNB Faculty of Computer Science.

References

[1] The atlantic computational excellence network (http://www.ace-net.ca/wiki/acenet).

[2] P. Afshani, L. Arge, and K. Larsen. Orthogonal range reporting: query lower bounds, optimal structures in 3-d, and higher-dimensional improvements. In *Proceedings of the 2010 annual symposium on Computational geometry*, pages 240–246. ACM, 2010.

[3] P. K. Agarwal. *Range searching. CRC Handbook of Discrete and Computational Geometry.* CRC Press, Inc., 2004.

[4] L. Arge, V. Samoladas, and J. Vitter. On two-dimensional indexability and optimal range search indexing. In *Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 346–357. ACM, 1999.

[5] J. Aspnes and G. Shah. Skip graphs. *ACM Transactions on Algorithms*, 3(4), 2007.

[6] B. Chazelle. Filtering search: A new approach to query-answering. *SIAM J. Comput.*, 15(3):703–724, 1986.

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

[8] B. Chazelle. Lower bounds for orthogonal range searching: Ii. the arithmetic model. *J. ACM*, 37(3):439–463, 1990.

[9] R. Corless, G. Gonnet, D. Hare, D. Jeffrey, and D. Knuth. On the lambertw function. *Advances in Computational mathematics*, 5(1):329–359, 1996.

[10] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry.* Springer, 2008.

[11] M. Goodrich, M. Nelson, and J. Sun. The rainbow skip graph: a fault-tolerant constant-degree distributed data structure. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 384–393. ACM, 2006.

[12] M. Goodrich, M. Nelson, and J. Sun. The rainbow skip graph: A fault-tolerant constant-degree p2p relay structure. pages 384–393, New York, NY, USA, 2009. ACM.

[13] G. Gratzer. *Lattice theory.* WH Freeman, 1971.

[14] W. Gropp, E. Lusk, and A. Skjellum. Using MPI: portable parallel programming with the message passing interface. 1999.

[15] N. J. A. Harvey, M. B. Jones, S. Saroiu, M. Theimer, and A. Wolman. Skipnet: a scalable overlay network with practical locality properties. In *USITS'03: Proceedings of the 4th conference on USENIX Symposium on Internet Technologies and Systems*, pages 9–9, Berkeley, CA, USA, 2003. USENIX Association.

[16] W. Pugh. Skip lists: a probabilistic alternative to balanced trees. *Communications of the ACM*, 33(6):668–676, 1990.

[17] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. pages 161–172, 2001.

[18] R. Sridhar, S. Iyengar, and S. Rajanarayanan. Range search in parallel using distributed data structures. *Journal of Parallel And Distributed Computing*, 15(1):70–74, 1992.

[19] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication Review*, 31(4):149–160, 2001.

[20] K. C. Zatloukal and N. J. A. Harvey. Family trees: an ordered dictionary with optimal congestion, locality, degree, and search time. In *SODA '04: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 308–317, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.