Bitmap Indices for Speeding Up High-Dimensional Data Analysis

Kurt Stockinger

CERN, European Organization for Nuclear Research
CH-1211 Geneva, Switzerland
Institute for Computer Science and Business Informatics
University of Vienna, A-1010 Vienna, Austria
Kurt.Stockinger@cern.ch

Abstract. Bitmap indices have gained wide acceptance in data warehouse applications and are an efficient access method for querying large amounts of read-only data. The main trend in bitmap index research focuses on typical business applications based on discrete attribute values. However, scientific data that is mostly characterised by non-discrete attributes cannot be queried efficiently by currently supported access methods.

In our previous work [13] we introduced a novel bitmap algorithm called `GenericRangeEval` for efficiently querying scientific data. We evaluated our approach based primarily on uniformly distributed and independent data. In this paper we analyse the behaviour of our bitmap index algorithm against various queries based on different data distributions.

We have implemented an improved version of one of the most cited bitmap compression algorithms called `Byte Aligned Bitmap Compression` and adapted it to our bitmap indices. To prove the efficiency of our access method, we carried out high-dimensional queries against real data taken from two different scientific applications, namely High Energy Physics and Astronomy. The results clearly show that depending on the underlying data distribution and the query access patterns, our proposed bitmap indices can significantly improve the response time of high-dimensional queries when compared to conventional access methods.

1 Introduction

Over the last three decades many different index data structures were proposed to optimise the access to database management systems. Some of these are optimised for one-dimensional queries such as the B+tree whereas others are optimised for multi-dimensional queries such as the Pyramid-tree [3] or bitmap indices [8]. Recently, bitmap indices have gained wide acceptance in data warehouse applications and are an efficient access method for evaluating complex, multi-dimensional ad-hoc queries against read-only data.

The main trend in bitmap index research focuses on typical business applications based on discrete attribute values. However, scientific data that is mostly characterised by non-discrete attributes cannot be queried efficiently by currently supported access methods.

In our previous work we introduced a novel bitmap algorithm called `GenericRangeEval` [13]. In addition, we proposed a cost model for predicting the performance of bitmap indices for scientific data and based our analysis primarily on uniformly distributed and independent data. In this paper we analyse the query performance of our bitmap indices based on different data distributions. We have implemented an improved version of one of the most cited bitmap compression algorithms called `Byte Aligned Bitmap Compression` and compare the performance to uncompressed bitmap indices. The main contribution of the paper is to demonstrate that bitmap indices can significantly improve the performance of high-dimensional queries against real data from different scientific disciplines.

The paper is organised as follows. In Section 2 we briefly review the related work on conventional bitmap indices and in Section 3 we discuss bitmap indices for scientific data. Section 4 is dedicated to an experimental evaluation of our bitmap compression algorithm. In Section 5 we present our results of applying bitmap indices to High Energy Physics and Astronomy. Conclusions and future work are given in Section 6.

2 Related Work

The simplest form of bitmap indices called *equality encoding* [8, 4] encodes each distinct attribute value as one bitmap (or bit slice). This encoding technique is optimised for so-called *exact-match queries* [12] of the form $Q_e : v = a_i$. [4, 5] proposed *range encoding* and *interval encoding* which are optimised for *one-sided range queries* $Q_{1r} : v_1\ op\ a_i$ where $op \in \{<, \leq, >, \geq\}$ and *two-sided range queries* $Q_{2r} : v_1\ op\ a_i\ op\ v_2$ respectively. *Interval encoding* is the most storage efficient encoding technique since it requires only half of the number of bitmaps when compared to *equality encoding* or *range encoding*.

[16] represented attribute values in binary form that yields indices with only $\lceil log_2 |A| \rceil$ bitmaps, where $|A|$ is the attribute cardinality. The advantage of this encoding scheme is that the storage overhead is even smaller than for *interval encoding*. However, query processing is in most cases more efficient with *interval encoding* since in the worst case only two bitmaps need to be read whereas binary encoding always forces all bitmaps to be read.

One of the main problems of bitmap indices is the large space overhead for high cardinality attributes. Thus, [19] proposed *range-based* indices. In this approach a bitmap is used to present an attribute range rather than a distinct attribute value. However, a detailed analysis and a possible solution to the problem of the additional overhead for retrieving data from disk ("sieving out" the matching attribute values), was still left an open issue.

Another possibility to overcome the storage overhead for high cardinality attributes is to compress the bitmaps. Various bitmap compression algorithms are discussed in [1, 6, 2, 18].

3 Bitmap Indices for Scientific Data

3.1 Bitmap Evaluation Algorithm

Conventional bitmap indices encode each distinct attribute value with one bit slice. However, for non-discrete attribute values, attribute ranges rather than attribute values are encoded [9, 12]. In our previous work we introduced a novel bitmap evaluation algorithm called `GenericRangeEval` [13]. An extended version can be found in [15]. Since we base all our analyses throughout this paper on this algorithm, we briefly review the main characteristics and outline the *candidate check problem*.

Assume a conventional range encoded bitmap index [4] with 6 physical bins which represents an attribute with cardinality 7. In our case, 7 attribute ranges are represented. Assuming that our encoded search space is in the range of [0;140), each bin represents a range that is a multiple of 20, as depicted in Figure 1.

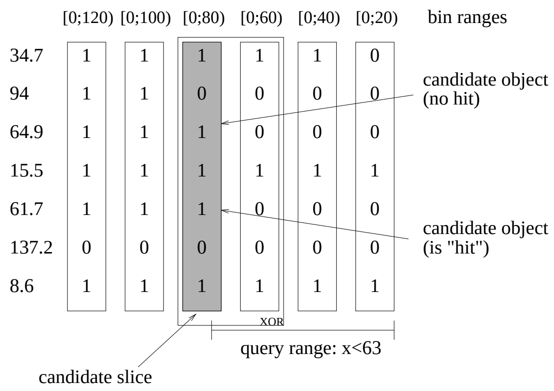

Fig. 1. One-sided range query on a range encoded bitmap index. The query $x < 63$ is evaluated via `GenericRangeEval`.

When we issue the following query $x < 63$, the preliminary foundset would be bin 3 (high-lighted in grey). Since this bin represents values in the range of [0;80), the foundset contains not only so-called *hits* (values 34.7, 15.5 and 8.6) but also *candidates* (values 64.9 and 61.7) that need to be checked against the query constraint. Candidates are all those attribute values in the current bin B_i that are not represented in bin B_{i-1}. We assume that the first bin is denoted by B_0. By XOR-ing B_i with B_{i-1} (see Figure 1) we yield two candidate objects with the values 64.9 and 61.7. These candidate objects need to be checked against the query constraint $x < 63$. Finally, the candidate object with the value 64.9 needs to be discarded since it does not fulfill the query constraint.

In this simple example we only outlined the behaviour of `GenericRangeEval` in a one-dimensional search space. For a detailed discussion about the behaviour in a multi-dimensional search space, we refer to [15].

3.2 Binning Strategies

An important tuning parameter of the bitmap indices for scientific data is the binning. In particular, we can distinguish between *equi-width* and *equi-depth* bins.

With *equi-width* bins, each bin has the same width and thus the distribution of the data values is also reflected in the number of entries for each bin. On the other hand, *equi-depth* binning guarantees that each bin has the same number of entries. Both binning strategies have advantages and disadvantages.

In short, the binning strategy depends on two factors, namely the *data distribution* and the *query access patterns* (or the query distribution). Equi-depth binning guarantees nearly constant access time for all kind of queries independent of the data distribution. One would chose this kind of binning when no query access patterns are available.

Since equi-width bins reflect the data distribution, this kind of binning is preferable if the query access patterns are such that those bins queried most have the least number of entries. Optimal binning strategies are discussed in [7].

4 Bitmap Compression

One of the disadvantages of bitmap indices for high cardinality attributes is the large space complexity, i.e. the high storage overhead for keeping a large number of bins persistently on disk. One way to reduce the space complexity is to compress the index. However, an important requirement for a bitmap compression algorithm is not only to have a good compression ratio but also to show good performance for the logical operations between the bit slices.

In the following section we discuss the impact of bitmap compression on range encoded bitmap indices based on `GenericRangeEval`. Together with Theodore Johnson from AT&T Labs-Research we implemented an improved version [6] of the original Byte-Aligned Bitmap Compression (BBC) algorithm by [1]. In addition, we also implemented the Boolean operations for two-sided BBC which was still left as an open issue in [6] due to the complexity of the algorithm.

We will now analyse the performance of compressed bitmap indices based on data following a typical exponential distribution of the form e^x (see Figure 2 a). We have chosen this kind of distribution since it is very common in High Energy Physics applications (see Section 5.1) which we mainly use for verifying our results. Our benchmark operates on 10^6 objects with 25 attributes (dimensions). Each range encoded bitmap index consists of 100 equi-width bins per indexed attribute. The benchmark was carried out on a Pentium II 400 under Linux Red Hat 6.1. The bitmap index is implemented with especially developed, persistent data structures on top of Objectivity/DB [14].

In Figure 2 b) we can see the compression ratio for each bin using two-sided Byte-Aligned Bitmap Compression (BBC2). We can observe that especially bins on the right side, i.e. bins with a high ordinal number, show good compressibility whereas bins with a low ordinal number show poor compressibility. The reason for the good compressibility is that due to the cumulative effect of range encoding [15], the bins with a high ordinal number have most of the bits set to 1. Similarily, bins with a low ordinal number have most of the bits set to 0 and can thus also be compressed. However, the compression ratio highly depends on the underlying data distribution.

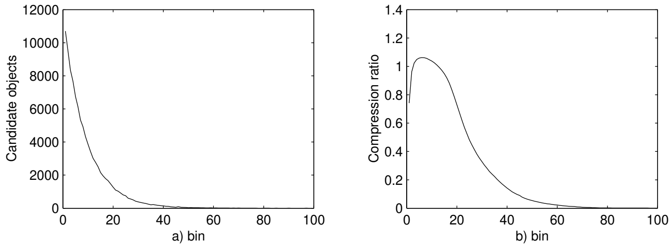

Fig. 2. Range encoded bitmap index based on data following an exponential distribution.

In the following we will compare the query response time of the compressed bitmap index (BBC2) with the response time of the uncompressed (verbatim) bitmap index. In Figure 3 a) and c) we plotted the query response time for the index I/O, i.e. not including the I/O operations for the candidate check. The graphs show the response times for one-sided 10-dimensional and 25-dimensional queries with various *query boundaries* ranging from bin 0 to bin 100. Let us explain this with an example.

Assume, for instance, an exponential distribution with values in the range of [0;10) and a bitmap index with 100 equi-width bins (as depicted in Figure 3). Thus, the boundary of the query $a_1 < 5$ is bin 50.

To sum up our results of Figures 3 a) and c), in the best case the compressed bitmap index is a factor of two faster than the verbatim bitmap index whereas in the worst case the compressed bitmap index is about a factor of two slower than the verbatim one. As we can see, the compressed bitmaps are faster for queries with high bin boundaries, i.e. for bins which show good compressibility. Thus, reading the compressed bit slice requires a smaller number of page I/Os. In addition, the logical operations can be performed faster when compared with verbatim bit slices. The total query response time, including the I/O operations for the candidate check, are given in Figures 3 b) and d).

Finally, we carried out the same benchmarks on a more powerful machine called "tier2" with higher CPU and disk I/O rates. The results are summarised in Figure 4. We can observe that the impact of bitmap compression yields only

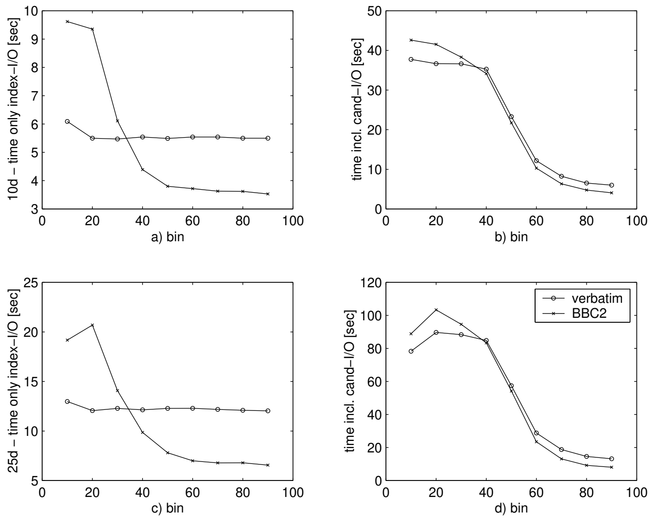

Fig. 3. Response time for verbatim vs. compressed bitmap indices.

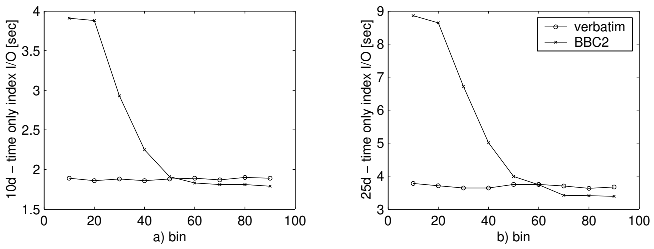

Fig. 4. Response time for verbatim vs. compressed bitmap indices on "tier2".

a slight performance improvement for queries with high bin boundaries when compared to the verbatim bitmap index. However, as we expect that in the future the CPU speed will increase faster than the I/O rate, the performance improvement due to bitmap compression will be even more significant.

5 Applications

5.1 High Energy Physics

We will now evaluate the performance of the bitmap index with queries against real data taken from High Energy Physics. The data consists of 1,401,020 objects with 37 attributes each. The size of this data is 262 MB.

Similar to our previous benchmark, we created a bitmap index for each attribute consisting of 100 equi-width bins. The size of the whole bitmap index for all attributes is 790 MB. Out of these 37 attributes we have chosen 10 attributes which we study more carefully. The distributions of some of these attribute values are reflected by the distribution of the bins (see Figure 5).

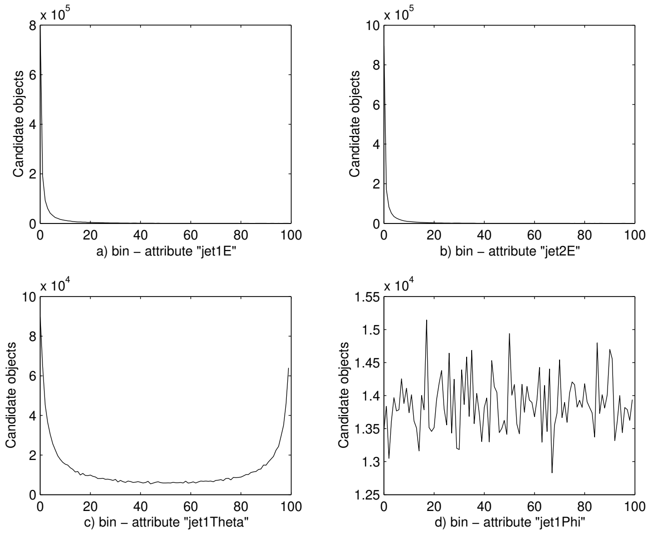

Fig. 5. Effect of equi-width binning for the attributes `jet1E`, `jet2E`, `jet1Theta` and `jet1Phi`.

All benchmarks are carried out on a machine with the following configuration: Dual 933 MHz Pentium III Linux server, 900 MB RAM, using a 600 GB 3ware RAID 0 array. Note, that currently all the data fits in main memory but in a few years the data of the physics experiments will be up to one Terabyte and thus will most likely not fit into main memory any more. However, for all our benchmarks we assured that the disk cache was trashed, i.e. the queries were run against a cold cache.

Based on some previous studies of access patterns in High Energy Physics [15], we have chosen a set of 13 example queries over one to ten dimensions. Due to space restrictions we only list one query and refer to [15] for more details:

```
Q8: Bitmaps ‘‘jet1E > 1000 && jet2E > 1000 && part1E > 1000 &&
jet1Phi > 0 && part1Phi > 0’’
```

Table 1 shows the response time for each of the 13 queries over various dimensions. Similar to [3], we also compare the query response time to the sequential scan. The results show that the bitmap index shows better performance than the sequential scan for high-dimensional queries. It is important to note that with a higher number of bit slices and with higher-dimensional queries, the performance improvement can be even more significant.

Query	#dims	time [sec]	time seq. scan [sec]
Q1	1	6.9	3.6
Q2	1	5.8	3.6
Q3	2	6.3	5.5
Q4	3	6.6	8.0
Q5	4	6.8	10.1
Q6	4	6.8	10.1
Q7	4	6.8	10.1
Q8	5	7.0	12.3
Q9	5	15.1	12.3
Q10	5	7.0	12.3
Q11	8	20.6	20.6
Q12	10	20.8	23.2
Q13	10	13.9	23.2

Table 1. Response time for 13 multi-dimensional sample queries - bitmap index vs. sequential scan.

5.2 Astronomy

In this sub-section we will summarise the experience we gained when we evaluated the performance of bitmap indices for typical Astronomy queries of the Sloan Digital Sky Survey (SDSS) database [10]. All SDSS data is stored in Objectivity/DB and can be retrieved by a special SQL-like query interface developed by the astronomers. Our main challenge was to show that bitmap indices can even further improve the already optimised astronomy queries against the database [11].

The astronomy data consists of 6,182,527 objects with 65 attributes each. The total size is 1.9 GB. We created 65 bitmap indices with 100 equi-width bins each. The total size of all indices is 5.8 GB.

We evaluated the performance of the bitmap index against queries taken from the SDSS query server logs. In particular, the logs contained 357 queries of 41 different users; 49 out of these queries are against the data set called "sxGalaxy" which we studied in more detail. We took three representative range queries and compared the performance of the Sloan Sky Server against the performance of the bitmap index. One typical query looks as follows:

```
SELECT g,r,i FROM sxGalaxy
WHERE ((RA() between 180 and 185) && (DEC() between 1. and 1.2)
&& (r between 10 and 18) && (i between 10 and 18)
&& (g between 10 and 18))
```

Our results show a significant performance improvement of a factor of 10 to 20 when compared to the current implementation. The reason for the signifcant performance improvement is as follows. First, we clustered the data attribute-wise [14] rather than object-wise (as you would do for typical object oriented database management systems). Second, the queries against the highly specialised bitmap indices caused less page I/O than the current indices of SDSS. We have thus successfully demonstrated that with object oriented database management systems we can also get significant improvements with access methods and clustering techniques which are mainly used in relational database management systems. For further details of this study we again refer to [15].

6 Conclusions

In this paper we evaluated the performance of our bitmap index called `GenericRangeEval` against various queries based on different data distributions. We have implemented an extended version of one of the most cited bitmap compression algorithms called `Byte-Aligned Bitmap Compresion` and adopted it for our bitmap indices in order to cope with typical scientific data. We have demonstrated that the query response times of compressed bitmap indices can be significantly lower than for uncompressed bitmap indices. However, the speedup factor strongly depends on the ratio of CPU versus I/O speed. Since the CPU speed is expected to increase faster than the I/O rate, the performance improvements gained with bitmap compression will be even more significant in the future.

To prove the efficiency of our access method, we analysed the performance of bitmap indices for high-dimensional queries from two different scientific applications, namely High Energy Physics and Astronomy. In both cases the results clearly show that bitmap indices can lead to significant performance improvements when compared to conventional access methods. However, the performance improvement strongly depends on the underlying data distribution and the query access patterns. As part of our future work we will continue our investigation on a cost model for predicting the performance of bitmap indices based on various data distributions, in order to automate the choice between conventional access methods and bitmap indices for high-dimensional queries.

7 Acknowledgments

Our special thanks go to Erich Schikuta who always showed great enthusiasm during the write-up of this paper. We would also like to thank Jim Gray for the very fruitful discussions concerning the Astronomy data.

References

1. G. Antoshenkov, Byte-Aligned Bitmap Compression, *Technical Report, Oracle Corp.*, 1994.
2. S. Amer-Yahia, T. Johnson, Optimizing Queries On Compressed Bitmaps, *Proceedings of 26th International Conference on Very Large Data Bases*, Cairo, Egypt, Sept. 2000, Morgan Kaufmann.

3. S. Berchtold, C. Boehm, H.-P. Kriegel, The Pyramid-Tree: Breaking the Curse of D imensionality, SIGMOD 1998, *Proceedings ACM SIGMOD International Conference on Management of Data*, Seattle, Washington, USA, June 1998.
4. C. Chan, Y.E. Ioannidis, Bitmap Index Design and Evaluation, *In Proceedings ACM SIGMOD International Conference on Management of Data*, Seattle, Washington, USA, June 1998.
5. C. Chan, Y.E. Ioannidis, An Efficient Bitmap Encoding Scheme for Selection Queries, *Proceedings ACM SIGMOD International Conference on Management of Data*, Philadephia, Pennsylvania, USA, June 1999.
6. T. Johnson, Performance Measurements of Compressed Bitmap Indices, *Proceedings of 25th International Conference on Very Large Data Bases*, Edinburgh, Scotland, UK, September 1999, Morgan Kaufmann.
7. N. Koudas, Space Efficient Bitmap Indexing, *International Conference on Information and Knowledge Management*, McLean, VA, USA, November 2000.
8. P. O'Neil, D. Quass, Improved Query Performance with Variant Indexes, *Proceedings ACM SIGMOD International Conference on Management of Data*, Tucson, Arizona, USA, May 1997.
9. A. Shoshani, L.M. Bernardo, H. Nordberg, D. Rotem, A. Sim, Multidimensional Indexing and Query Coordination for Tertiary Storage Management, *11th International Conference on Scientific and Statistical Database Management*, Cleveland, Ohio, USA, July 1999.
10. Sloan Digital Sky Survey, *http://www.sdss.org*
11. A.Szalay, P. Kunszt, A. Thakar, J. Gray, D. Slutz, Designing and Mining Multi-Terabyte Astronomy Archives: The Sloan Digital Sky Survey, *Proceedings ACM SIGMOD International Conference on Management of Data*, Philadephia, Pennsylvania, USA, June 1999.
12. K. Stockinger, D. Duellmann, W. Hoschek, E. Schikuta. Improving the Performance of High Energy Physics Analysis through Bitmap Indices. *International Conference on Database and Expert Systems Applications*, London - Greenwich, UK, Sept. 2000. Springer-Verlag.
13. K. Stockinger, Design and Implementation of Bitmap Indices for Scientific Data, *International Database Engineering & Applications Symposium*, Grenoble, France, July 2001, IEEE Computer Society Press.
14. K. Stockinger, Performance Analysis of Generic vs. Sliced Tags in HepODBMS, *International Conference on Computing in High Energy and Nuclear Physics*, Beijing, China, September, 2001.
15. K. Stockinger, Multi-Dimensional Bitmap Indices for Optimising Data Access within Object Oriented Databases at CERN, *Ph.D. Thesis*, University of Vienna, Austria, November 2001.
16. M. Wu, A.P. Buchmann, Encoded Bitmap Indexing for Data Warehouses, *Proceedings of the Fourteenth International Conference on Data Engineering*, Orlando, Florida, USA, February 1998.
17. M. Wu, Query Optimization for Selections Using Bitmaps, *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data*, Philadephia, Pennsylvania, USA, June 1999.
18. K. Wu, E. J. Otoo, A. Shoshani, A Performance Comparison of Bitmap Indexes, *International Conference on Information and Knowledge Management*, Atlanta, Georgia, USA, November, 2001.
19. K. Wu, P.S. Yu, Range-Based Bitmap Indexing for High-Cardinality Attributes with Skew, *Technical Report, IBM Watson Research Center*, May 1996.