

Calculating Entropy Estimate Using Binary Decision Diagrams

Stanislav Stanković

Jaakko Astola

Tampere International Center for Signal Processing Tampere International Center for Signal Processing
Tampere University of Technology Tampere University of Technology
P.O. Box 527, FI-33101 Tampere, Finland P.O. Box 527, FI-33101 Tampere, Finland
Email: stanislav.stankovic@tut.fi Email: jaakko.astola@tut.fi

Abstract

Information entropy is one of crucial concepts in modern information theory and related fields. Over the years a large number of methods for reliable estimation of information entropy have been invented. In this paper we introduce a method for the estimation entropy of binary streams based on binary decision diagrams. Binary decision diagrams are an efficient method of representation of discrete functions. By exploiting the useful properties of binary decision diagrams we may be able, in some cases, to significantly reduce the time and complexity of calculation of entropy estimate.

I. INTRODUCTION

The notion of entropy as introduced by Shannon in [1], [4], [9], represents one of the fundamental concepts in modern information theory and signal processing. In essence, entropy describes the amount of information carried by a signal. The applicability of this concept greatly transcends the field of information theory in narrow sense. Although the mathematical definition of information entropy is well known, the task of calculation of entropy of a particular signal is by no means trivial.

Formula introduced by Astola and Ryabko in [2] provides a good estimate of the entropy of a given signal. The calculation required for this method can be time consuming. Furthermore the complexity of this calculation depends strongly on the length of the given sequence. In this paper we will present a method to improve speed and reduce complexity of this calculation by exploiting properties of binary decision diagrams.

If characters of the alphabet are represented as discrete numerical values, the signal, output of the source can be seen as a discrete function. Decision diagrams are an efficient method of representation of discrete functions, [3]. As demonstrated in [10], reduced ordered binary decision diagrams are a canonic representation of discrete functions. In this paper we assume that a given function, binary vector, is already represented in the binary decision diagram form. We assume that the reader is familiar with the basic notions about decision diagrams, and do not go into details of construction of decision diagrams and the complexity of necessary algorithms. Representation of functions in this way is a standard approach in various fields, and this method goes well in pair with other similar decision diagram based calculation methods, [5], [7], [11]. Calculation of various spectral transforms, differential operators, autocorrelation functions, etc., can be performed in a unified manner by processing nodes in decision diagrams.

II. CALCULATION OF ENTROPY ESTIMATES USING BDDS

In [2], Ryabko and Astola introduce the following method for calculating the estimate of the entropy of a given vector. Consider an alphabet A and let $A^* = \bigcup_{i=1}^{\infty} A^i$ be a set of all finite words over A . Let $f = f_1 f_2 \dots f_t$ be a given vector of length t , and $v = v_1 \dots v_k$ be a possible substring of f , $f, v \in A^*$. Denote the rate of a substring v occurring in the vector x as $\nu_f(v)$. For example, if $f = 000100$ and $v = 00$, then $\nu_f(00) = 3$, since, we calculate as follows $(00)0100 \rightarrow 0(00)100 \rightarrow 00(01)00 \rightarrow 000(10)0 \rightarrow 0001(00)$, and the sequence $v = 00$ appears three times.

For any $0 \leq k \leq t$ the empirical Shannon entropy of order k is defined as follows:

$$h_k^* = - \sum_{v \in A^k} \frac{\bar{\nu}_f(v)}{(t-k)} \sum_{a \in A} \frac{\nu_f(va)}{\bar{\nu}_f(v)} \log \frac{\nu_f(va)}{\bar{\nu}_f(v)}. \quad (1)$$

In order to calculate the entropy estimate by (1) we need to determine the number of occurrences of all possible substrings v of the length k in the given vector. This is the most computationally intensive part of the process. It can be implemented using a straightforward method, by moving a window of length k over a given vector and increasing the appropriate counter for each encountered substring. In a way this is a brute force method. It does not take into account the individual properties of a particular string. Assume, for example, that the second half of a string is equal to the first half of the string. If we take into account this information, we could reduce the amount of needed calculation by one half. The application of decision diagrams permit us to do this. In this paper we focus only on calculation of the entropy estimate for binary strings. For the sake of simplicity, we demonstrate the proposed method first for the simplest case of $k = 2$. We also assume that given vectors are of the length $t = 2^n$, so that a binary decision diagrams can be constructed.

We begin with the following observation. Let $f_{s1} = f_1 f_2 \dots f_{\frac{t}{2}}$ and $f_{s2} = f_{\frac{t}{2}} f_{\frac{t}{2}+1} \dots f_t$ be the first and second half of the given vector. It naturally follows that: $\nu_f(v) = \nu_{f_{s1}}(v) + \nu_{f_{s2}}(v) + c$, which is the number of occurrences of a certain substring of length $k = 2$ in a given binary vector, is equal to the number of occurrences of this substring in the first half of the vector, plus the number of occurrences of this substring in the second half of the given vector. Minor correction of plus one should be made if the given substring occurs exactly at the split of two halves of the given vector so that its first character remains in the first half of the string while the last ends up in the second half.

$$c = 1 \Leftrightarrow v_1 = f_{\frac{t}{2}}, v_2 = f_{\frac{t}{2}+1}, c = 0, \text{ otherwise.}$$

We can recursively apply this observation to each half of the string, splitting them further in shorter and shorter segments, until we reach the segments of the length 2. At this point the calculation of occurrence of a substring becomes trivial. If the given substring v is identical to a particular segment f_{sn} , then $\nu_f(v)$ needs to be incremented by one. We must keep in mind that the modifications to the calculated number every time the substring occurs at the split between the examined segments of the original vector, potentially at every step of the recursion.

This recursive procedures resembles the structure of a binary decision tree. If we mark the levels of a binary decision diagram from 0 for the topmost level containing the root node to $n + 1$ for the level of terminal nodes, then the nodes of the second lowest level n in the diagram correspond to substrings of length 2 in the underlying binary vector. From the properties of decision diagrams, it follows that the exact number a certain substring v occurs in a given vector f is equivalent to the number of paths that that point to the node that is the root of the subdiagram representing the substring v . It can be shown that this number is equal to the weighted sum of incoming edges to the node. The weight associated with particular edge equals $w_i = 2^l$, where l is the difference of the level between the parent and then node in question.

Let $f = [0110101010101111]$ be a given binary vector of order $n = 4$. In Fig. 1 we present this vector in binary decision diagram form and the results of the calculation of number of occurrences of individual substrings.

To calculate the number of occurrences of all possible substrings of length $k = 2$ in a binary vector, we need to iterate through all the nodes at n -th level of decision diagram and calculate the number of the corresponding paths as a described weighted sum. Before we can perform any calculation, we need to extend the diagram with virtual nodes at each place where an edge intersects with n -th level. These

TABLE I

LUT of indices for $k = 2$.

	00	01	10	11
00	00	00	01	01
01	10	10	11	11
10	00	00	01	01
11	10	10	11	11

virtual nodes will correspond to substrings representing pairs of identical binary values $v = 00, v = 11$, which possibly exist in the original vector f .

However, this represents only a partial calculation of the entropy estimate, since it does not take into account the substrings that occur on the splits between the segments of the vector. In total there is $2^n - 1$ of these subvectors, for a vector of the length 2^n . We can show that a binary decision diagram contains all the information necessary for complete calculation of occurrence rate of subvectors of order $k = 2$. For a given binary vector, a subvector of length $k = 2$ starting at position i is determined by the last character of preceding vector at position $i - 1$, and the first character of the following vector at position $i + 1$. Consider an example ...1100..., let $v_{i-1} = 11$ and $v_{i+1} = 00$, it clearly follows that $v_i = 10$. The complete set of these relations can be expressed in tabular form, see Table I. This look-up-table is identical for all binary decision diagrams and needs to be generated only once at the beginning of the process.

Entries of this table can be replaced by the integers using encoding $00 = 0, 01 = 1, 10 = 2$, and $11 = 3$. Due to that, the Table 1 can be concisely expressed by the function $f = \bar{x}_2x_3 + 2x_2\bar{x}_3 + 3x_2x_3$, where x_2 , and x_3 are Boolean variables 0 and 1 interpreted as integers 0 and 1. By choosing between this analytical expression and tabular representation we are premitted to trade between temporal complexity and mamory requirements. If subvectors v_{i-1} and v_{i+1} are represented by adjacent nodes at $n - 1$ level in the decision diagram, substring v_i will correspond to their common parent node. In Fig. 2 we demonstrate the correspondence between subvectors of the previous example and nodes of the full decision tree. This decision tree can be reduced to the decision diagram in Fig. 1 as shown in Fig. 3.

Fig. 4 shows the order in which subvectors for $k = 2$ case are read during the traversal through the decision diagram. For the sake of clarity we present this over a full decision tree.

It is evident that in order to completely determine the occurrence rates of all subvectors of a given vector we need to visit all the nodes in the diagram. The complete algorithm therefore represents an inorder traversal of the decision diagram, with the following additional steps performed at each node:

If the level of the current node is $< n + 1$,

- 1) determine to which subvector the node corresponds,
- 2) increase the value of appropriate counter by the weight of the corresponding incoming edge,
- 3) store the index of the counter.

If the level of the current node is $> n + 1$,

- 1) based on indices stored in children nodes, determine the preceding and following substring,
- 2) increase the appropriate counter.

After the complete traversal of the tree we obtain the final results, $\nu(v_1) = \nu(01) = 5 + 1 = 6$, $\nu(v_2) = \nu(10) = 0 + 5 = 5$, $\nu(v_1) = \nu(11) = 2 + 2 = 4$.

Finally we can compare these values with values obtained by application of original method. For original vector $f = [0110101010101111]$ after the application of sliding window we obtain 01, 11, 10, 01, 10, 01, 10, 01, 10, 01, 10, 01, 11, 11, 11, that is, $\nu(v_1) = \nu(01) = 6$, $\nu(v_2) = \nu(10) = 5$, $\nu(v_1) = \nu(11) = 4$. It is evident that both methods have produced identical results. The rates of occurrence of subvectors can be then directly used in (1) to calculate the final entropy estimate. The complexity of this algorithm is proportional to the number of nodes in decision diagram, versus the length of the given vector which

TABLE II

Number of steps needed to calculate the occurrence rate of substrings using a BDD, and the standard approach.

function	BDD	Standard Approach
BW 0	10	31
5XP 10	12	127
MISEX1 0	66	255
RD84 0	4	255
APEX4 0	258	511
APEX4 10	6	511
EX1010 0	10	1023
MISEX3 0	8194	16383
MISEX3 3	514	16383
MISEX3 4	4	16383

was the complexity of standard approach. The same argument about efficiency of decision diagrams in general can be applied to this algorithm. For further details please refer to [3], [5], [12].

A. Generalization

We have demonstrated the method of calculating entropy estimate using binary decision diagrams on the most simple example for $k = 2$ length of subvectors. However, this method can be easily generalized for arbitrary length of subvectors of form $k = 2^j$. Similar as in previous example, for the case $k = 4$, $j = 2$, consider a subvector $v_i = 1101$, and its neighboring subvector $v_{i+4} = 1001$. After a simple observation we determine that there are additional subvectors spanning a split between v_i and v_{i+4} , namely $v_{i+1} = 1011$, $v_{i+1} = 0110$, $v_{i+1} = 1100$. This fact will be reflected in the structure and size of look up table of indices needed for the process.

For the general case $k = 2^j$, LUT will have $2^j x 2^j$ cells in which we need to store $k - 1$ index values, thus adding to memory complexity of the overall algorithm. As this table needs to be stored only once for all calculations of entropy estimates for the same k , in real application for reasonable length of subvectors this overhead will not be significant.

III. EXPERIMENTAL RESULTS

To demonstrate the validity of the proposed method, we have conducted a set of experiments. We have counted the number of steps needed to calculate the rate of occurrence of substrings in a given vector using both the standard and BDD based approach. In these experiments mathematical center of North Carolina set of benchmark functions was used, a standard set of benchmark functions for logic design and BDD related problems. The Table II presents the selected results. First column in the table represents number of steps needed for BDD based method, and the second results for the standard approach.

The number of steps in standard approach is determined by the length of original vector, while in BDD based approach it is determined by the number of nodes. The example of function misex3, and its particular outputs 0, 3, and 4, illustrates that the approach using BDDs, depends on the number of nodes, unlike the standard approach, where the complexity remains the same for all the outputs. We did not take into account the number of steps needed for construction of decision diagram, since the basic assumption was that functions were already represented in the decision diagram form. The efficient method for construction of decision diagrams is a complex problem out of the scope of this paper. Details on that can be found in [6], [8], [10].

IV. CONCLUSION

In this paper we have demonstrated how binary decision diagrams can be used for efficient computation on entropy estimate for a binary source. The proposed method was designed according to same lines as similar binary decision diagram based calculation methods. We have used, as a basis, intuitive approach

introduced by Ryabko and Astola [2] and introduced modifications using the properties of binary decision diagrams. In this manner we were able to make significant reductions in complexity of the calculation and needed calculation time.

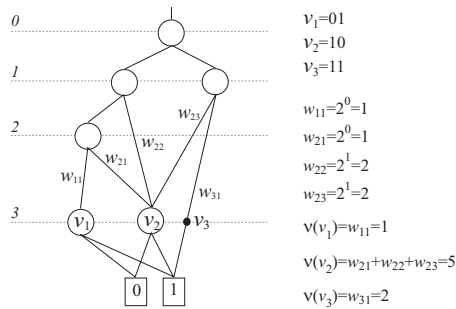


Fig. 1. Example of calculation of entropy estimate via binary decision diagram.

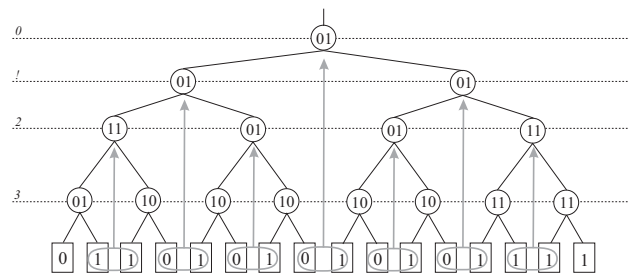


Fig. 2. subvectors of order $k = 2$ over nodes of a decision tree.

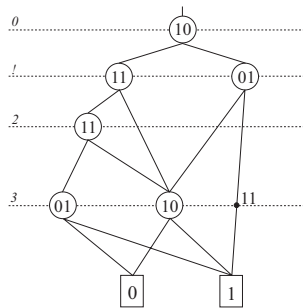


Fig. 3. subvectors of order $k = 2$ over nodes of a reduced ordered decision diagram.

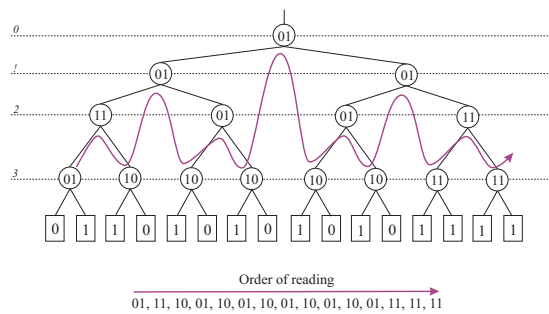


Fig. 4. Order of reading of subvectors for $k = 2$, presented on a full decision tree.

REFERENCES

- [1] Ash, R. B., Information Theory, John Wiley and Sons, 1967.
- [2] Astola, J., Ryabko, B., "Universal Codes as a Basis for Time Series Testing",
- [3] Bryant, R. E., "Graph-based algorithms for Boolean functions manipulation", IEEE Trans. Comput., Vol. C-35, No. 8, 1986, 667 – 691.
- [4] Cover, T. M., and Thomas, J., A., Elements of Information Theory, John Wiley, 1991, New York.
- [5] Clarke, E. M., McMillan, K. L., Zhao, X., Fujita, M., "Spectral transforms for extremely large Boolean functions", in: Keschull, U., Schubert, E., Rosenstiel, W., Eds., Proc. IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design, 16-17.9.1993, Hamburg, Germany, 86-90.
- [6] Drechsler, R., Becker, B., Binary Decision Diagrams - Theory and Implementations, Kluwer Academic Publishers, 1998.
- [7] Malyugin, V. D., Stanković, R. S., Stanković, M. S., "Calculation of the coefficients of polynomial representations of switching functions through binary decision diagrams", Proc. Int. Conf. on Preventive Engineering, PIT'94, Dec. 1994, Niš, Yugoslavia, 10-1 - 10-4.
- [8] Sasao, T., Fujita, M., Representations of Discrete Functions, Kluwer Academic Publishers, 1996.
- [9] Shannon, C. E., "A mathematical Theory of communications", Bell Sys. Tech. J., Vol. 27, 1848.
- [10] Stanković, R. S., Astola, J. T., Spectral Interpretation of Decision Diagrams, Springer, 2003.
- [11] Stanković, R. S., Karpovsky, M. G., "Remarks on calculation of autocorrelation on finite dyadic groups by local transformations of decision diagrams", in R. Moreno-Diaz, F. Pichler, Q. Arencibia, (eds.), Computer Aided Systems Theory - EUROCAST 2005, Las Palmas de Gran Canaria, Spain, February 7-11, 2005, Revised Selected Papers, Series Lecture Notes in Computer Science, Vol. 3643/2005, Springer, Berlin/Heidelberg, Germany, 2005, 301-310.
- [12] Yanushkevich, S.N., Miller, D.M., Shmerko, V.D., Stankovic, R.S., Decision Diagram Techniques for Micro and Nanoelectronics Design Handbook, CRC Press/Taylor & Francis, 2006.