Discrete Applied Mathematics 16 (1987) 79-85
North-Holland

COMMUNICATION

DUALIZATION OF REGULAR BOOLEAN FUNCTIONS

Y. CRAMA
RUTCOR, Rutgers University, New Brunswick, NJ 08903, USA

Communicated by P.L. Hammer
Received 9 June 1986

A monotonic Boolean function is regular if its variables are naturally ordered by decreasing 'strength', so that shifting to the right the non-zero entries of any binary false point always yields another false point. Peled and Simeone recently published a polynomial algorithm to generate the maximal false points (MFP's) of a regular function from a list of its minimal true points (MTP's). Another efficient algorithm for this problem is presented here, based on a characterization of the MFP's of a regular function in terms of its MTP's. This result is also used to derive a new upper bound on the number of MFP's of a regular function.

1. Introduction

A *monotonic Boolean function of n variables* (or, for short, a function) is a mapping $f: \{0,1\}^n \to \{0,1\}$ such that: $x \leq y$ implies $f(x) \leq f(y)$. A function f is called *regular* if it satisfies the following condition at every point x:

$$\text{if } i<j,\ x_i=1,\ x_j=0 \text{ and } f(x)=0, \text{ then } f(x+e_j-e_i)=0,$$

where e_k denotes as usual the k-th unit vector of appropriate dimension.

Regularity and related concepts have been studied under various names in such areas of applied mathematics as threshold logic (Hu [4], Muroga [6]), game theory (Maschler and Peleg [5], Einy [2]) or graph and hypergraph theory (Chvátal and Hammer [1], Reiterman et al. [8]). The interest in regular functions usually stems from their close relationship with *threshold* functions: for our purpose, a Boolean function $f(x)$ is called threshold if there exist $n+1$ integers $c_1 \geq c_2 \geq \cdots \geq c_n \geq 0$ and $b \geq 0$ such that:

$$f(x)=0 \quad \text{if and only if} \quad \sum_{i=1}^{n} c_i x_i \leq b.$$

Clearly, every threshold function is also regular, and thus the class of regular functions provides a (proper) generalization of the class of threshold functions.

Now, define a *true point* (or winning coalition, or dependent set) of a Boolean function f as a point x such that $f(x)=1$. Similarly, x is a *false point* (or losing coalition, or independent set) of f if $f(x)=0$. Clearly, the list of *minimal true points* (MTP) or of *maximal false points* (MFP) of a (monotonic) function f is sufficient to completely specify f. With this terminology, the *threshold synthesis* problem can

be stated as follows: given the list of MTP's of a Boolean function f, decide whether f is threshold, and if yes, produce a linear inequality defining it (see e.g. [4], [6]). Equivalently, this amounts to deciding whether a simple game defined by its minimal winning coalitions is a weighted majority game, or to recognizing threshold hypergraphs (see [8]).

This classical problem of threshold logic was open for a number of years, until the recent publication by Peled and Simeone of a polynomial-time algorithm for its solution [7]. In fact, the existence of such an algorithm is an easy corollary of the following theorem, which can therefore be seen as the main result of [7]:

Theorem 1 [7]. *There is an algorithm that accepts as input the list of MTP's of a regular function f, and that outputs the list of MFP's of f in* $O(n^3m)$ *time, where m is the number of MTP's of f. In particular, f has no more than* $nm+m+n$ *MFP's.*

An algorithm that produces the list of MPF's of a Boolean function from its list of MTP's will henceforth be called a *dualization algorithm*. The dualization algorithm of Peled and Simeone is an improved version of a procedure originally suggested by Hammer, Peled and Pollatschek [3]. Its time complexity is low, but its description and the proof of its validity are quite involved. The bound on the number of MFP's follows directly from a careful analysis of the algorithm.

The main result of this paper is a simple, combinatorially insightful, characterization of the MFP's of a regular function in terms of its MTP's. This result is proved in Section 2, where it is used to derive an $O(n^2m)$ dualization algorithm for regular functions, as well as an improved upper-bound on the number of MFP's of such functions. In Section 3, we introduce the notion of 'shelters', and discuss some further refinements of the results presented in Section 2.

2. MTP's, MFP's, and a dualization algorithm

The key result of this paper states:

Proposition 1. *For a regular function f of n variables, x is an MFP of f such that* $x_n=0$ *if and only if* $x+e_n$ *is an MTP of f.*

Proof. (Only if) If x is an MFP of f and $x_n=0$, then $x+e_n$ is a true point of f. Moreover, x being a false point of f, $x+e_n-e_i$ is a false point too, by regularity, for all i such that $x_i=1$. Hence, $x+e_n$ is an MTP of f.

(If) If $x+e_n$ is an MTP of f, then $x_n=0$, and x is a false point of f. Assume that x is not an MFP, i.e., there exists $i<n$ such that $x+e_i$ is a false point. Then, by regularity, $x+e_n$ is also a false point: contradiction. □

Proposition 1 provides a simple characterization of all the MFP's of f with last component 0. Given this result, the following recursive dualization procedure suggests itself: list all the MFP's of f with last component 0, then fix x_n at 1 and iterate. In order to formalize these ideas and to establish the validity of the approach, we introduce now one more definition.

If f is a Boolean function of n variables, then f_n denotes the restriction of f obtained by fixing the n-th variable at 1. Hence, f_n is a function of $n-1$ variables. The following properties are easy to check, and we state them without proof.

Proposition 2. *For a monotonic function f of n variables,*

(i) *x is an MFP of f_n if and only if $(x, 1)$ is an MFP of f.*

(ii) *x is an MTP of f_n if and only if either $(x, 0)$ is an MTP of f, but for no $y<x$ is $(y, 1)$ an MTP of f, or $(x, 1)$ is an MTP of f.*

(iii) *f_n is regular if f is regular.*

The practical implications of Proposition 2 are obvious: given the list of MTP's of a regular function f, and using Proposition 1, we can easily find the MFP's of f with last component 0; by Proposition 2(i), we may then restrict our attention to the MFP's of f_n; the MTP's of f_n are readily available from Proposition 2(ii), and f_n is regular by Proposition 2(iii); thus, the whole problem may be solved iteratively. A straightforward implementation of this procedure yields an $O(n^2m^2)$ dualization algorithm for regular functions with m MTP's. Our next result will allow us to reduce this time complexity by a factor of m.

For a non-zero binary point x, denote by $l(x)$ the largest index k such that $x_k = 1$. If f is regular, the statement of Proposition 2(ii) can be sharpened as follows:

Proposition 3. *For a regular function f of n variables, x is an MTP of f_n if and only if either $(x, 0)$ is an MTP of f, but $(x - e_{l(x)}, 1)$ is not, or $(x, 1)$ is an MTP of f.*

Proof. (Only if) This is an immediate corollary of Proposition 2(ii).

(If) If $(x, 1)$ is an MTP of f, then x is an MTP of f_n by Proposition 2(ii). Now, assume that $(x, 0)$ is an MTP of f, and that x is not an MTP of f_n. We will deduce from these assumptions that $(z, 1) = (x - e_{l(x)}, 1)$ must also be an MTP of f. Indeed, by Proposition 2(ii), there exists $y<x$ such that $(y, 1)$ is an MTP of f. Let j be any index in $\{1, 2, \ldots, n-1\}$ such that $y_j = 0$ and $x_j = 1$. Then, by regularity, $(y + e_j, 0)$ is a true point of f, and, by minimality of $(x, 0)$, it follows that $x = y + e_j$. If $j = l(x)$, then $(z, 1) = (y, 1)$ is an MTP of f, and we are done.

So, assume $j < l(x)$; $(x - e_j, 1)$ being a true point of f, $(z, 1)$ is a true point of f too, by regularity. Moreover, $(z, 0)$ is a false point of f, since $(x, 0)$ is an MTP. Hence, by regularity again, $(z - e_i, 1)$ is a false point of f, for all i such that $z_i = 1$. But this means that $(z, 1)$ is an MTP of f. □

We are now in a position to state formally our dualization algorithm. We assume

that the input to the algorithm is an $m \times n$ matrix A, whose rows a_i $(i = 1, 2, \ldots, m)$ represent the MTP's of f, sorted in increasing lexicographic order:

$$\text{if } i < k \text{ and } s = \min \{j | a_{ij} \neq a_{kj}\}, \text{ then } a_{is} = 0.$$

The condition "$a_s \leq a_i$" in the statement of the procedure is to be understood componentwise.

Dualization algorithm
begin
  for $j = n$ **down to** 1 **do**
    $s \leftarrow 0$;
    for $i = 1$ **to** m **do**
      if row i is labelled 'removed' **then next** i;
      if $a_{ij} = 1$
        then begin
          $s \leftarrow i$;
          output $(a_i - e_j + e_{j+1} + \cdots + e_n)$;
          $a_{ij} \leftarrow 0$;
        end
        else if $s \neq 0$ **and** $a_s \leq a_i$ **then** label row i 'removed';
      end if
    next i;
  next j;
end

Before proving the validity of this procedure, we first illustrate its use on an example.

Example. This example is borrowed from [3].

$$A = \begin{matrix} 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{matrix}$$

In the first 'for j' loop, the algorithm outputs 01010, 01100 and 10010. The third row of A is removed, and the updated matrix is:

$$A = \begin{matrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{matrix}$$

Then, the algorithm outputs 01001, 10001, removes the second, fourth and fifth rows, and A is reduced to:

$$A = \begin{matrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{matrix}$$

Finally, the algorithm outputs 00111, removes the second row of A and terminates.

Theorem 2. *The algorithm above correctly outputs the MFP's of f, in $O(n^2m)$ time. The number of MFP's of f is at most the number of non-zero entries of the input matrix A.*

Proof. (i) First we show that, at the beginning of each 'for j' loop, and for the current value of j, the rows of A not labelled 'removed' are exactly the MTP's of $f_{j+1,\ldots,n}$ (completed by some 0's which fill-up the last columns of A: we will disregard this detail from now on). Indeed, this is certainly true at the beginning of the first 'for j' loop. Now, assume it also holds for some $j \leq n$. Then, during the loop, the MTP's of $f_{j,\ldots,n}$ are computed using Proposition 3 as follows.

If $a_{ij}=1$, then a_{ij} is simply set to 0.

If $a_{ij}=0$, then we only have to check whether or not $z+e_j$ is an MTP of $f_{j+1,\ldots,n}$, where $z=a_i-e_l$ and $l=l(a_i)$. But it is easy to see that, if $z+e_j$ is such an MTP, then it is the last MTP with j-th component equal to 1 encountered before a_i, i.e., a_s+e_j (this is because the rows are ordered lexicographically, and this order is preserved by setting the last components to 0 or by removing rows). Thus, a_i is removed if $a_i-e_l=a_s$, or equivalently if $a_s \leq a_i$.

Now, it is easily seen that, for each $j \in \{1,2,\ldots,n\}$, in decreasing order, the algorithm outputs the MFP's x of f such that $x_j=0$ and $x_{j+1}=\cdots=x_n=1$. Indeed, by Propositions 1 and 2, these are exactly the binary points of the form:

$$a_i-e_j+e_{j+1}+\cdots+e_n,$$

where a_i is an MTP of $f_{j+1,\ldots,n}$ such that $a_{ij}=1$.

This proves the correctness of the algorithm.

(ii) The main 'if block' of the algorithm is iterated at most nm times, and can be implemented to run in $O(n)$ time. Hence, the total running time of the procedure is $O(n^2m)$.

(iii) The bound on the number of MFP's is now trivial, since the algorithm outputs at most one MFP per non-zero entry of A. □

Remark. Notice that, if the rows of A are not originally in lexicographic order, then they can be so sorted in $O(nm \log m)$ time. Since $m \leq 2^n$, the total running time of the dualization algorithm is still $O(n^2m)$.

From Theorem 2, it follows immediately that a regular function with m MTP's has at most nm MFP's: hence, our bound on the number of MFP's is strictly better

than that given by Theorem 1. In the next section, we will show that some preprocessing procedure can be used to reduce the size of the input matrix, thus yielding an improved upper bound on the number of MFP's as well as a theoretically more efficient dualization algorithm.

3. Shelters

An MTP x of a regular function f is called a *shelter* of f if $l<n$ and $x-e_l+e_{l+1}$ is not an MTP of f, or if $l=n$, where $l=l(x)$ (see [3]). So, Proposition 1 can be rephrased as follows:

Proposition 1'. *For a regular function f of n variables, x is an MFP of f such that $x_n=0$ if and only if $x+e_n$ is a shelter of f.*

It is easy to see that a regular function is completely specified by the set of its shelters, and Peled and Simeone heavily exploit that property in [7]. Indeed, the first phase of their dualization algorithm consists in extracting the shelters of f from its list of MTP's: this can be done in $O(nm)$ time if the MTP's are ordered lexicographically. The shelters constitute the input to the second phase of their algorithm, in which the dualization is effectively carried out. Denoting by q the number of shelters of f, Peled and Simeone show that this second phase runs in $O(n^3q)$ time, and that the number of MFP's of f does not exceed $nq+q+n$.

Now, denote by A^* the $q\times n$ matrix whose rows are the shelters of f sorted in lexicographic order, and assume that A^* constitutes the input to the dualization algorithm described in Section 2. Then, by Proposition 1', the algorithm will correctly output the MFP's of f with last component 0. If we can show that, upon completion of the first 'for j' loop, the updated matrix contains all the shelters of f_n, then it will follow by an easy induction argument that the algorithm eventually outputs all the MFP's of f.

With this in mind, we prove now:

Proposition 4. *If x is a shelter of f_n, then either $(x,0)$ or $(x,1)$ is a shelter of f.*

Proof. (i) Let x be a shelter of f_n. If $(x,0)$ is not an MTP of f, then it follows from Proposition 2(ii) that $(x,1)$ is an MTP, and hence a shelter, of f.

So, we may assume from now on that $(x,0)$ is an MTP of f. We will also assume that $(x,0)$ is not a shelter of f, and show that this leads to a contradiction. Let $l=l(x)$; our assumption means that $x'=(x,0)-e_l+e_{l+1}$ is an MTP of f.

(ii) *Case 1:* $l=n-1$. Hence, $x'=(x,1)-e_{n-1}$ is an MTP of f. But then, by Proposition 2(ii), $x-e_{n-1}$ is an MTP of f_n, and this contradicts the minimality of x.

(iii) *Case 2:* $l<n-1$. So, $x'=(z,0)$, where $z=x-e_l+e_{l+1}$. Since x is a shelter of f_n, z is not an MTP of f_n. But $x'=(z,0)$ is an MTP of f. Hence, by Proposition 2(ii), there exists an MTP $(y,1)$ of f such that $y<z$.

Because $y<z$, $y_l=0$; and because y is an MTP of f_n, $y_{l+1}=1$ (else, $y<x$). So, by regularity, $(y+e_l-e_{l+1}, 1)$ is a true point of f. But:

$$y+e_l-e_{l+1}<z+e_l-e_{l+1}=x,$$

and this contradicts the assumption that x is an MTP of f_n. □

We are now ready to prove:

Theorem 3. *When running on the $q\times n$ input matrix A^*, the algorithm of Section 2 correctly outputs the MFP's of f in $O(n^2q)$ time. The number of MFP's of f is at most the number of non-zero entries of A^*.*

Proof. The proof is similar to the proof of Theorem 2. As discussed above, we only have to show that, at the beginning of the second 'for j' loop, the updated matrix A^* contains all the shelters of f_n (plus possibly some other MTP's of f_n).

So, consider an arbitrary shelter x of f_n. By Proposition 4, $(x,0)$ or $(x,1)$ is a shelter of f.

If $(x,1)$ is a shelter of f, then x is a row of the updated matrix.

If $(x,0)$ is a shelter of f, then $(x,0)-e_l+e_{l+1}$ is a false point of f. Hence, by regularity, $(x,1)-e_l$ is a false point of f, and x is a row of the updated matrix (see Proposition 3). □

Remark. If the list of MTP's of f is not originally sorted in lexicographic order, then the overall time complexity of the dualization procedure is $O(nm\log m+n^2q)$.

Acknowledgements

This research was supported by the Air Force Office of Scientific Research Grant #AFOSR 0271 and by the National Science Foundation Grant #ECS 85-03212 to Rutgers University.

References

[1] V. Chvátal and P.L. Hammer, Aggregation of inequalities in integer programming, Annals of Discrete Math. 1 (1977) 145–162.

[2] E. Einy, The desirability relation of simple games, Math. Social Sciences 10 (1985) 155–168.

[3] P.L. Hammer, U.N. Peled and M.A. Pollatschek, An algorithm to dualize a regular switching function, IEEE Trans. Computers 28 (1979) 238–243.

[4] S.T. Hu, Threshold Logic (University of California Press, Berkeley, 1965).

[5] M. Maschler and B. Peleg, A characterization, existence proof and dimension bounds for the kernel of a game, Pacific J. Math. 18 (1966) 289–328.

[6] S. Muroga, Threshold Logic and its Applications (Wiley-Interscience, New York, 1971).

[7] U.N. Peled and B. Simeone, Polynomial-time algorithms for regular set-covering and threshold synthesis, Discrete Appl. Math. 12 (1985) 57–69.

[8] J. Reiterman, V. Rödl, E. Šiňajová and M. Tůma, Threshold hypergraphs, Discrete Math. 54 (1985) 193–200.