A polynomial quantum computing algorithm for solving the dualization problem for positive boolean functions

Mauro Mezzini[1,*,†], Fernando Cuartero Gomez[2,†], Fernando Lopez Pelayo[2,†], Jose Javier Paulet Gonzalez[3,†], Hernan Indibil de la Cruz Calvo[2,†] and Vicente Pascual[2,†]

[1]*Department of Education, Roma Tre University, 00185 Rome, Italy*

[2]*Computing Systems Department, Faculty of Computer Science Engineering, University of Castilla-La Mancha, 02071 Albacete, Spain*

[3]*Quantum Computing Department, Qsimov Quantum Computing S.L, 45600 Talavera de la Reina, Toledo, Spain*

Abstract

Given two positive Boolean functions $f : \{0,1\}^n \to \{0,1\}$ and $g : \{0,1\}^n \to \{0,1\}$ expressed in their positive irredundant DNF Boolean formulas, the dualization problem consists in determining if g is the dual of f, that is if $f(x_1, \ldots, x_n) = \overline{g}(\overline{x}_1, \ldots \overline{x}_n)$ for all $(x_1, \ldots x_n) \in \{0,1\}^n$. In this paper we present a quantum computing algorithm that solves the dualization problem in polynomial time with respect to the dimensions of the DNF expressions.

Keywords

Quantum Algorithm, Dualization, Boolean Functions, Computational complexity,

1. Introduction

We deal in this paper with Boolean functions $f : \{0,1\}^n \to \{0,1\}$ on n Boolean or binary variables. A literal is a variable x or its negation $\overline{x}$. Given two Boolean functions $f : \{0,1\}^n \to \{0,1\}$ and $g : \{0,1\}^n \to \{0,1\}$ we say that $g \leq f$ if $g(x) \leq f(x)$ for all $x \in \{0,1\}^n$. Given two Boolean vectors $v = (v_1, \ldots, v_n)$ and $w = (w_1, \ldots, w_n)$, we write $v \leq w$ if $v_i \leq w_i$ for all $i \in \{1, 2, \ldots, n\}$. A Boolean function is *positive* (or elsewhere called *monotone*) if $v \leq w$ implies $f(v) \leq f(w)$ [1]. A conjunction $C = \wedge_{i \in I} \ell_i$, $I \subseteq \{1, \ldots, n\}$, of literals is an *implicant* of a Boolean function f if $C \leq f$. An implicant C of f is called *prime* if there is no implicant $D \neq C$ of f such that $C \leq D$. If there exists an implicant D of f such that $C \leq D$ then we say that D *absorbs* C. In other words C is prime if it is not absorbed by any other implicant of f distinct from C.

AIxIA 2023: International Workshop on AI for Quantum and Quantum for AI, November 06–11, 2023, Rome, IT

*Corresponding author.

†These authors contributed equally.

mauro.mezzini@uniroma3.it (M. Mezzini); Fernando.Cuartero@uclm.es (F. C. Gomez); fernandol.pelayo@uclm.es (F. L. Pelayo); jose.paulet@uclm.es (J. J. P. Gonzalez); hernanindibil.cruz@uclm.es (H. I. d. l. C. Calvo); vpfuniversity@gmail.com (V. Pascual)

0000-0002-5308-0097 (M. Mezzini); 0000-0001-6285-8860 (F. C. Gomez); 0000-0001-7849-087X (F. L. Pelayo); 0000-0001-7849-087X (J. J. P. Gonzalez); 0000-0001-6445-5256 (H. I. d. l. C. Calvo)

It is known [1] that a positive Boolean function f always can be expressed in a disjunctive normal form (DNF) containing no negated literals. We will call it a *positive* DNF expression of f. In the following we will denote a positive DNF expression of a positive Boolean function f as

$$\varphi = \bigvee_{I \in F} \bigwedge_{i \in I} x_i \tag{1}$$

where F is a family of subsets of $\{1, 2, \ldots, n\}$. For any $I \in F$ the implicant $\bigwedge_{i \in I} x_i$ is called *term* of the DNF φ. A positive DNF expression of a boolean function is *prime* if all its terms are prime implicants of f; furthermore it is said *irredundant* if there is no $J \in F$ such that

$$\psi = \bigvee_{I \in F, I \neq J} \bigwedge_{i \in I} x_i$$

is another positive DNF representation of f. The following theorem characterizes positive DNF expressions

Theorem 1 ([1] Theorem 1.24 pag.37)**.** *Let φ be a positive DNF expression of a positive Boolean function f. Then φ contains all of the prime implicants of f and it is irredundant if and only if no term of φ is absorbed by any other term of φ.*

By Theorem 1, a positive DNF which contains only and all the implicants of a positive Boolean function f is unique and irredundant. We will call it *positive irredundant DNF* (PIDNF). The *dualization* problem [2, 3, 4, 5], given a positive Boolean function $f : \{0,1\}^n \to \{0,1\}$ expressed in its PIDNF, consists in finding the PIDNF of a positive Boolean function g such that $f(x) = \overline{g}(\overline{x})$ for all $x \in \{0,1\}^n$. The decision version of the dualization problem, is defined as follows: given two positive Boolean functions f and g expressed in their PIDNF, is g the dual of f? The dualization problem and its associated decision version, are prominent problems in several research areas such as machine learning and data mining [6, 7, 8, 9] artificial intelligence [10, 11, 12], database systems and others (see [2] and the references within). The best deterministic classical computing algorithm for solving the dual problem has complexity $O(N^{o(\log N)})$ where $N = |F| + |G|$ and $|F|$ and $|G|$ are the number of terms of the PIDNF expression of f and g [5]. Determining the complexity status of the dualization problem and its associated decision version is a prominent open problem [13]. Equally interesting is the self-dualization problem, that is, the problem of determining if a positive Boolean function, expressed in its PIDNF, is self-dual. Obviously, if we set g equal to f, the self-dualization can be cast as a dualization problem. Conversely, given two distinct Boolean functions f and g, the dualization problem can be reduced to the self-dualization of the function $h = yf \vee zg \vee yz$ where y and z are two additional Boolean variables [5]. Therefore, in the following, we treat the self-dualization and the dualization problems as equivalent problems having the same complexity. In this paper we develop quantum computing algorithm for the self-dualization problem whose complexity is polynomial in the number of term of the PIDNF expression of f.

2. Methods

In the following the variable x is interpreted sometimes as a Boolean (or binary) n-dimensional vector and sometimes as the decimal expression of the binary vector. In particular if x is the

decimal value of the binary vector $(x_1, \ldots, x_n)$ then the decimal value of the binary vector $(\overline{x}_1, \ldots, \overline{x}_n)$ is $\overline{x} = 2^n - x - 1$. We start with the following proposition which will be often used later in the paper.

Proposition 2 ([5]). *Necessary condition for two positive Boolean functions $f = \bigvee_{I \in F} \bigwedge_{i \in I} x_i$ and $g = \bigvee_{J \in G} \bigwedge_{j \in J} x_j$ expressed in their PIDNF to be mutually dual is that*

$$I \cap J \neq \emptyset \text{ for every } I \in F \text{ and } J \in G \tag{2}$$

Proof. If, by contradiction, there exist implicants $I \in F$ and $J \in G$ such that $I \cap J = \emptyset$, let $x = (x_1, \ldots, x_n)$ such that $x_i = 1$ if $i \in I$ and $x_i = 0$ if $i \notin I$. Clearly $f(x) = 1 = g(\overline{x})$ and f and g could not be mutually dual. □

By Proposition 2, if f is self-dual then every implicant of F must intersect every other implicant.

Lemma 3. *Suppose f is self-dual. Then f is balanced, that is, for half of x values is 0 and for the other half is 1.*

Proof. Let $0 \leq x < 2^n$, then $\overline{x} = 2^n - x - 1$. Furthermore since f is self-dual we have that $f(x) \neq f(\overline{x})$ for all $0 \leq x < 2^n$. Therefore

$$\begin{aligned}
&\sum_{x=0}^{2^{n-1}-1} f(x) + \sum_{x=2^{n-1}}^{2^n-1} f(x) = \\
&\sum_{x=0}^{2^{n-1}-1} f(x) + \sum_{x=0}^{2^{n-1}-1} f(2^n - x - 1) = \\
&\sum_{x=0}^{2^{n-1}-1} [f(x) + f(\overline{x})] = 2^{n-1}
\end{aligned}$$

□

Lemma 4. *Let f be a positive Boolean function expressed in its PIDNF which satisfies also (2). Then f is self-dual if and only if $\sum_{x=0}^{2^n-1} f(x) = 2^{n-1}$.*

Proof. The necessity is given by Lemma 3. As for the sufficiency, suppose that $\sum_{x=0}^{2^n-1} f(x) = 2^{n-1}$ and suppose by contradiction that $f(x) = f(\overline{x})$ for some $0 \leq x < 2^n$. Since (2) holds, when $f(x) = 1$ there exists an implicant I such that $x_i = 1$ for all $i \in I$. But then $f(\overline{x}) = 0$ since I intersects all other implicants of F. In other words $f(x) + f(\overline{x}) \leq 1$ for all $0 \leq x < 2^n$. Therefore we must have that $f(z) = f(\overline{z}) = 0$ for some $0 \leq z < 2^n$. But since

$$2^{n-1} = \sum_{x=0}^{2^n-1} f(x) = \sum_{x=0}^{2^{n-1}-1} [f(x) + f(\overline{x})] \leq 2^{n-1}$$

we must have, for every $0 \leq x < 2^{n-1}$, that $f(x) + f(\overline{x}) = 1$, and this is a contradiction. □

Now we state he following Remark which will be useful for the rest of the paper (see Theorem 1.11 of reference [1]).

Remark 5. *A term $C = \bigwedge_{i\in I} x_i$ of a positive DNF expression $\varphi = \bigvee_{I\in F} \bigwedge_{i\in I}$ of a positive Boolean function f is absorbed by a term $D = \bigwedge_{j\in J} x_j$ of φ if and only if $J \subseteq I$.*

We define $w(x)$ the *Hamming weight* of the integer $0 \leq x < 2^n$, as the number of ones in the binary representation of x, or, equivalently, if $x = (x_1, \ldots, x_n)$ is a binary vector, then $w(x) = \sum_{i=1}^{n} x_i$.

We said that the complexity of the dualization problem is measured with respect to the combined size of the PIDNF representation of f and g, that is, with respect to $N = |F| + |G|$. Furthermore as stated in [5], the number n of variables of the Boolean functions is always less than $|F||G|$. However there exists instances of the self-dual function in which $N = O(2^n)$ as in the following example.

Choose $n > 4$ odd and consider the following Boolean function whose positive DNF expression φ has as a set F of implicants, the set of all subsets of $\{1, \ldots, n\}$ of cardinality $\lceil n/2 \rceil$ where $\lceil a \rceil$ is the least integer greater or equal than a.

Lemma 6. *The function f expressed by φ is self-dual. Moreover φ is the PIDNF representation of f, and has a number of terms equal to $\binom{n}{\lceil n/2 \rceil}$.*

Proof. Trivially, by definition, $|F| = \binom{n}{\lceil n/2 \rceil}$. If there exist two implicants I and J such that $I \cap J = \emptyset$ then $|I \cup J| = |I| + |J| = 2\lceil n/2 \rceil > n$ a contradiction to the fact that the number of variables is n. So we have that (2) holds for φ.

For every x such that $w(x) < \lceil n/2 \rceil$ we have that $f(x) = 0$ since every implicant I of φ has cardinality $|I| = \lceil n/2 \rceil$. On the other hand for every x such that $w(x) \geq \lceil n/2 \rceil$ then $f(x) = 1$ since, if we consider x as a binary vector, we will always find an implicant I such that $x_i = 1$ for all $i \in I$. Now it is immediate to check that $|\{x : 0 \leq x < 2^n, w(x) \geq \lceil n/2 \rceil\}| = 2^{n-1}$. By Lemma 4, f is self-dual. It remains to show that φ is irredundant. By Theorem 1, φ is not irredundant if there is some term $C = \bigwedge_{i\in I} x_i$ of φ which is absorbed by some other term $D = \bigwedge_{i\in J} x_i$ of φ. By Remark 5, this happens if and only if $J \subseteq I$ and $J \neq I$. But this is impossible because $|J| = |I|$ for all pairs of $I, J \in F$. □

2.1. The quantum computing approach

In the following we give several quantum computing algorithms for the dual and self-dual problems.

2.1.1. Deutsch-Jozsa approach

Given two Boolean functions f and g we build the function $h(x) = f(x) \oplus \overline{g}(\overline{x})$ where $\oplus$ is the sum modulo two.

Note that h can be obtained from f and g by using a polynomial number of logic gates with respect to the number of terms in their PIDNF expressions. If $f(x) = \overline{g}(\overline{x})$ for all x then $h(x) = 0$ for all x; that is h is a constant function. We prepare a black box U_h which performs

the transformation $|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus h(x)\rangle$, for $0 \leq x < 2^n$. We use the blackbox U_h in the Deutsch-Jozsa [14] algorithm. We have that the measurements of first n qubits will be

$$\frac{1}{2^n} \sum_{z=0}^{2^n-1} \sum_{x=0}^{2^n-1} (-1)^{x \cdot z + h(x)} |z\rangle$$

and the probability of measuring for $|z\rangle = |0\rangle$ is, when $h(x) = 0$ for all x, equal to 1 since

$$\frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{h(x)} |0\rangle = |0\rangle$$

so we have the following remark.

Remark 7. *Let f and g be two monotone prime Boolean functions which also satisfy (2) and let $h = f \oplus g$. If we measure at the end of the Deutsch-Jozsa algorithm with blackbox function U_h, a value $|x\rangle \neq |0\rangle$ then f is not the dual of g.*

In the same way as above we could build a blackbox U_f and apply the Deutsch-Jozsa algorithm to check if f is balanced. By Lemma 3, if f is self-dual the Deutsch-Jozsa algorithm will output a value $|y\rangle \neq |0\rangle$ with probability one. If we measure a value equal to $|0\rangle$ at the end of the algorithm, we can conclude that f is not self-dual.

We note that if f is not self-dual then the Deutsch-Jozsa algorithm could output a value equal $|y\rangle \neq |0\rangle$ with probability $p < 1$. So if we repeat the algorithm k times and f is not self-dual, the probability of observing $|y\rangle \neq |0\rangle$ for k times will be p^k. We obtained, in this way, a probabilistic algorithm whose running time depend on p.

2.1.2. Quantum counting approach

Another approach to solve the self-dual problem would be to use the quantum counting algorithm [15]. In this approach we build a blackbox U_f. Let M be the number of x such that $f(x) = 1$. By Lemma 4, if f is self-dual and (2) holds, then $M = 2^{n-1}$. The quantum counting algorithm will estimate the phase of the eigenvalues of the Grover operator which are $e^{i\theta}$ or $e^{i(2\pi-\theta)}$ where θ is the angle of rotation of the Grover operator and it satisfies the following equation $\sin\frac{\theta}{2} = \sqrt{\frac{M}{2^n}}$. Thus, if $M = 2^{n-1}$, we have that $\theta = \frac{\pi}{2}$. We assume that the register for measuring the phase is composed by t qubits. At the end of the counting algorithm we will measure the phase φ of the eigenvalue $e^{i2\pi\varphi} = e^{i\theta}$ or $e^{i2\pi\varphi} = e^{i(2\pi-\theta)}$ from which we obtain that $\varphi = 1/4$ or $\varphi = 3/4$. Therefore, if the function f is self-dual, at the end of the counting algorithm, we should measure $|2^{t-2}\rangle$ or $|2^{t-1} + 2^{t-2}\rangle$ with probability one. In other words, if the measurement at the end of the counting algorithm is not equal to $|2^{t-1}\rangle$ and not equal to $|2^{t-1} + 2^{t-2}\rangle$ the function f is not self-dual.

We note that the number of iteration of the counting algorithm, and therefore its complexity, depend on the number of qubits t we use to approximate the phase φ.

2.1.3. Grover algorithm approach

A final approach is to use the Grover algorithm to find an x such that $f(x) = f(\overline{x})$. If such x is found then f is not self-dual.

2.1.4. Merging all the approaches

From Remark 1, Lemma 3 and Lemma 4 we may summarize the discussion above in the following quantum algorithm for checking if a function f is self-dual.

Algorithm Quantum Dual
Input: A PIDNF of a Boolean function f satisfying (2) and a black box U_f which performs the transformation $|x\rangle|y\rangle \to |x\rangle|y \oplus f(x)\rangle$, for $0 \leq x < 2^n$.

Output: *True* if f is self-dual and *False* otherwise.

Procedure:

1. Let $h(x) = f(x) \oplus \overline{f}(\overline{x})$. Use the Deutsch-Jozsa algorithm to check if h is constant. If the output of the Deutsch-Jozsa algorithm is not equal to $|0\rangle$ then output *False* and exit.
2. Use the Deutsch-Jozsa algorithm to check if f is balanced. If the output of the Deutsch-Jozsa algorithm is equal to $|0\rangle$ then output *False* and exit.
3. Use the Quantum Counting algorithm to count the number of x such that $f(x) = 1$ using $t = \lceil n/2 \rceil$ qubits to measure the phase angle. If the measurement at the end of the algorithm is $|y\rangle$ and if $y \neq 2^{t-2}$ and $y \neq 2^{t-1} + 2^{t-2}$ then output *False* and exit.
4. Use the Grover algorithm to find an x such that $f(x) = f(\overline{x})$. If such x is found then output *False* and exit.
5. Output *True*

The complexity of the algorithm Dual is dominated by the complexity of the quantum counting and of the Grover algorithms. Recall that n is the number of variables of the Boolean formula and N is the dimension of the algorithm's input. The blackbox used in the algorithms requires $O(nN)$ gates. Both algorithms achieve a complexity on the number of quantum iterations which is $O(2^{n/2})$ while the best deterministic classical computing algorithm has time complexity of $O(N^{o(\log N)})$ [5]. However, we saw in Lemma 6, that a self-dual function can have a number of terms in its PIDNF equal to $\binom{n}{\lceil n/2 \rceil}$ which is asymptotic to $O(2^n)$. Therefore we have that $N = O(2^n)$ from which we obtain that the complexity of our quantum algorithm for the dualization problem is $O(nN^{3/2}) = O(N^{3/2} \log N)$.

3. Conclusions

In this paper we shed a new light on the complexity of the dualization problem, with a perspective from a quantum computing approach. We demonstrate that the dualization problem can be solved by using a mixture of several quantum computing algorithms obtaining an exponential speed up if compared to the best classical computing algorithm proposed in literature when the

input size of the problem is exponential to the number of variables of the Boolean functions. We think that these ideas, far from being conclusive, could be used to develop faster, either classical or quantum computing, dualization algorithms.

References

[1] Y. Crama, P. L. Hammer, Boolean Functions - Theory, Algorithms, and Applications, volume 142 of *Encyclopedia of mathematics and its applications*, Cambridge University Press, 2011. URL: http://www.cambridge.org/gb/knowledge/isbn/item6222210/?site_locale=en_GB.

[2] T. Eiter, G. Gottlob, K. Makino, New results on monotone dualization and generating hypergraph transversals, SIAM Journal on Computing 32 (2003) 514–537. URL: https://doi.org/10.1137/S009753970240639X. doi:10.1137/S009753970240639X. arXiv:https://doi.org/10.1137/S009753970240639X.

[3] T. Eiter, G. Gottlob, Identifying the minimal transversals of a hypergraph and related problems, SIAM Journal on Computing 24 (1995) 1278–1304. URL: https://doi.org/10.1137/S0097539793250299. doi:10.1137/S0097539793250299. arXiv:https://doi.org/10.1137/S0097539793250299.

[4] T. Eiter, K. Makino, G. Gottlob, Computational aspects of monotone dualization: A brief survey, Discrete Appl. Math. 156 (2008) 2035–2049. URL: https://doi.org/10.1016/j.dam.2007.04.017. doi:10.1016/j.dam.2007.04.017.

[5] M. L. Fredman, L. Khachiyan, On the complexity of dualization of monotone disjunctive normal forms., J. Algorithms 21 (1996) 618–628.

[6] D. Gunopulos, R. Khardon, H. Mannila, H. Toivonen, Data mining, hypergraph transversals, and machine learning, in: Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database and Knowledgebase Systems (PODS'97), ACM, United States, 1997, pp. 209–216.

[7] E. Boros, V. Gurvich, L. Khachiyan, K. Makino, Dual-bounded generating problems: Partial and multiple transversals of a hypergraph, SIAM Journal on Computing 30 (2001) 2036–2050. URL: https://doi.org/10.1137/S0097539700370072. doi:10.1137/S0097539700370072. arXiv:https://doi.org/10.1137/S0097539700370072.

[8] E. Boros, V. Gurvich, L. Khachiyan, K. Makino, On the complexity of generating maximal frequent and minimal infrequent sets, in: H. Alt, A. Ferreira (Eds.), STACS 2002, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002, pp. 133–141.

[9] C. Domingo, N. Mishra, L. Pitt, Efficient read-restricted monotone cnf/dnf dualization by learning with membership queries, Mach. Learn. 37 (1999) 89–110. URL: https://doi.org/10.1023/A:1007627028578. doi:10.1023/A:1007627028578.

[10] R. Khardon, Translating between horn representations and their characteristic models, J. Artificial Intelligence Res. 3 (1995) 349–372.

[11] G. Gogic, C. Papadimitriou, M. Sideri, Incremental recompilation of knowledge, J. Artificial Intelligence Res. 8 (1998) 23–37.

[12] R. Reiter, A theory of diagnosis from first principles, Artificial Intelligence 32 (1987) 57–95.

[13] G. Gottlob, E. Malizia, Achieving new upper bounds for the hypergraph duality problem through logic, SIAM Journal on Computing 47 (2018) 456–

492. URL: https://doi.org/10.1137/15M1027267. doi:10.1137/15M1027267. arXiv:https://doi.org/10.1137/15M1027267.

[14] D. Deutsch, R. Jozsa, Rapid solution of problems by quantum computation, Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences 439 (1992) 553 – 558. URL: https://api.semanticscholar.org/CorpusID:121702767.

[15] M. A. Nielsen, I. L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2000.