Journal of Mathematical Sociology, 39: 92–108, 2015

ISSN: 0022-250X print/1545-5874 online
DOI: 10.1080/0022250X.2014.897949

Routledge
Taylor & Francis Group

ENHANCING THE MINIMIZATION OF BOOLEAN AND MULTIVALUE OUTPUT FUNCTIONS WITH *e*QMC

Adrian Duşa
Department of Sociology, University of Bucharest, Bucharest, Romania

Alrik Thiem
Department of Philosophy, University of Geneva, Geneva, Switzerland

Configurational comparative methods have gained in popularity among sociologists and political scientists. In particular, Qualitative Comparative Analysis (QCA) has attracted considerable attention in recent years. The process of Boolean minimization by means of the Quine-McCluskey algorithm (QMC) is the central procedure in QCA, but QMC's exactitude renders it memory intensive and slow in processing complex output functions. In this article, we introduce the* enhanced *QMC algorithm (*e*QMC) to alleviate these problems.* e*QMC is equally exact but, unlike QMC, capable of processing multivalent condition and outcome factors. Instead of replacing QMC, however,* e*QMC acts as an optimizing complement in contexts of limited empirical diversity. We demonstrate its speed and computer memory performance through simulations.

Keywords: Boolean minimization, enhanced Quine-McCluskey algorithm, Qualitative Comparative Analysis (QCA), Quine-McCluskey algorithm, set theory

1. INTRODUCTION

Configurational comparative methods have gained in popularity among social scientists. In particular, Qualitative Comparative Analysis (QCA) has allowed sociologists and political scientists to present a number of compelling findings.[1] Influential contributions have been made to social movement theory (e.g., Amenta, Caren, & Olasky, 2005; Cress & Snow, 2000), welfare state research (Hicks, Misra, & Ng, 1995; Vis, 2009), trade and labor union studies (Brueggemann & Boswell, 1998; Dixon, Roscigno, & Hodson, 2004), collective action and social systems theory (Lam & Ostrom, 2010; Schlager & Heikkila, 2009), economic reform and development studies (Cornell & Kalt, 2000; Weyland, 1998) and democracy and democratization research (Berg-Schlosser & De Meur, 1994). All of these works have argued, in one way or another, that QCA would do more methodological justice to the implicational asymmetry of their

Both authors contributed equally to this work.

[1]According to the COMPASSS database, about 37% of all applied QCA publications have appeared in sociology and comparative politics. See http://www.compasss.org/bibdata.htm for more details (accessed March 1, 2015).

Address correspondence to Alrik Thiem, University of Geneva, Department of Philosophy, Rue de Candolle 2/Bât. Landolt, 1211 Geneva, Switzerland. E-mail: alrik.thiem@unige.ch

research hypotheses than linear algebra-based techniques of regression analysis. The logic of equifinal conjunctural causation, not that of additive explained variation, has motivated these authors in their choice of method (Amenta & Poulsen, 1994).

The basic variant of QCA, usually called *crisp-set* QCA, firmly rests on Boolean algebra—a theory of mathematical structures that has long been integral to many areas of the natural and technical sciences, including genetic biology, information technology, and electrical engineering (e.g., Brayton, Hachtel, McMullen, & Sangiovanni-Vicentelli, 1984; Edwards, 1973; Thomas, 1973).[2] In social sciences, however, Boolean algebra has led a niche existence until the pioneering article on employment discrimination by Ragin, Mayer, and Drass (1984) and the textbook by Ragin (1987) introduced its basic principles to a wider community of scholars. In essence, the core mechanism of QCA consists in the use of a few Boolean-algebraic theorems to reduce an exhaustive output function which represents a formal description of regularity patterns to a minimally complex equivalent that still preserves all the essential properties of the original representation of the function.[3]

This process of reduction is referred to as *Boolean minimization*, for which different algorithms have been introduced. Among them, *Quine-McCluskey* (QMC) is most well-known (McCluskey, 1956; Quine, 1952). It is exact and suitable for processing moderately complex Boolean functions. For example, the popular QMC-based QCA software `fs/QCA` 2.5 (Ragin & Davey, 2012) can handle about 12 condition factors.[4] But exactitude comes at a cost. For many industrial applications, the memory-intensity of QMC makes its use "totally impractical even for medium sized problems" (Brayton et al., 1984, p. 8). Approximate methods developed in response can process considerably more factors at much higher speeds, but they cannot guarantee exact solutions. Conversely, this orientation renders them unsuitable for research in the social sciences, where very few problems are of a magnitude similar to those found in very large-scale circuit integration. In consequence, beyond a certain number of condition factors, there exists a trade-off for problems of Boolean minimization between result exactitude and model complexity.

In this article, we present a new algorithm for minimizing Boolean and multivalue output functions, which we call *enhanced* Quine-McCluskey (*e*QMC) because it complements QMC instead of replacing it entirely.[5] The mathematical approach of *e*QMC

[2]Two generalized variants of crisp-set QCA exist. The first extension is multivalue QCA (Cronqvist & Berg-Schlosser, 2009), the second fuzzy-set QCA (Ragin, 2008). Although the former rests on multivalue logic and the latter on fuzzy logic, both of which generalize Boolean algebra in different directions, we refer to Boolean algebra throughout this article for simplicity because the principle behind the minimization procedure to be introduced remains the same. A fourth variant called generalized-set QCA has recently joined the fold (Thiem, 2013, 2014).

[3]For an introduction to Boolean algebra, see Hohn (1966). The standard terms in QCA for Boolean input and output variables are *condition* and *outcome factors*. We use these terms and the nomenclature of set theory as one branch of Boolean algebra to distinguish between linear-algebraic and Boolean-algebraic operations.

[4]Ragin (2000) has also introduced an *inclusion algorithm* which parted in many ways with QMC, but abandoned it again later to replace it with the QMC-based *truth table algorithm* (Ragin, 2008).

[5]The *e*QMC algorithm has been implemented in the `QCA` package for the R environment (Duşa & Thiem, 2014; Thiem & Duşa, 2013b, 2013c; R Development Core Team, 2014), currently in version 1.1-4, and is thus available to applied users and open to review. For a review of current software and their specific minimization algorithms, see Thiem and Duşa (2013a).

guarantees exact solutions, but it generates them at a fraction of the memory and time resources consumed by QMC. Moreover, *e*QMC is not confined to bivalent factors but generalizes to multivalent ones (Duşa, 2011).[6] As a result, it creates new possibilities for conducting configurational comparative research by pushing back the conceptual and computational boundaries that have set certain limits.[7] Although these limits have been rarely approached in applied research with QCA, we argue that if nothing is lost for the majority of applications at the lower to intermediate ranges of complexity, but much gained for a minority at the upper ranges, the introduction of *e*QMC makes a significant net contribution to the field of configurational comparative modeling.

The article is structured around three sections. In Section 2, we revisit some elementary concepts in Boolean algebra and explain the logic behind QMC for minimizing Boolean output functions. To prepare the ground for the following sections and also to facilitate later generalization, we do not strictly adhere to Boolean algebra but extend the exposition to multivalent factors. Subsequently, Section 3 introduces the solution to QMC's shortcomings offered by *e*QMC. After its fundamental steps will have been expounded on in a first subsection, two following subsection on the role of logical remainders in *e*QMC's minimization procedure and the identification of prime implicants, respectively, complete this part of the article. In Section 4, we demonstrate the superiority of *e*QMC over QMC with respect to speed and memory consumption through a set of simulations. In the conclusions, practical implications of our results are broached and future avenues of research identified.

2. BOOLEAN MINIMIZATION AND QMC

The essential device in applications of Boolean minimization such as QCA is the *function table*.[8] A function table $\mathbf{F}_{d\times(k+w)}$ is a two-dimensional array of d rows and $k+w$ columns. Any row c_i, with $i=1, 2,\ldots, d$, over the first k columns in $\mathbf{F}$ is called a *configuration*. Every configuration c_i represents a unique product over all p_j possible combinations of levels $v_{h,j}$ of condition factors $\mathbf{X}_j$, with $h=1, 2, \ldots, p$ and $j=1, 2,\ldots, k$. Every configuration c_i is thus associated with a unique k-way Boolean product as given by Eq. (1):

$$c_i = \mathbf{X}_1\{v_h\} \cap \mathbf{X}_2\{v_h\} \cap \ldots \cap \mathbf{X}_k\{v_h\} = \bigcap_{j=1}^{k} \mathbf{X}_j\{v_h\}. \tag{1}$$

The set of all configurations $\mathcal{C}^{\mathbf{F}} = \{c_1, c_2,\ldots, c_d\}$ constitutes the *configuration matrix* of $\mathbf{F}$, with d being equal to the arithmetic product of the total number of levels of all condition factors as given by Eq. (2):

$$d = p_1 \cdot p_2 \cdot \ldots \cdot p_j \cdot \ldots \cdot p_k = \prod_{j=1}^{k} p_j. \tag{2}$$

[6]We use the terms *bivalent, trivalent, multivalent*, etc. to indicate the number of levels a factor comprises.

[7]We explicitly speak of "possibilities" since the advantages of *e*QMC over QMC do not materialize under all research designs. We elaborate on this point in Section 4.

[8]Function tables are also commonly known in QCA as *truth tables* or *tables of combinations*.

Based on the number of level sequences $seq_j^{\mathbf{F}}$ within j and the number of repetitions $rep_j^{\mathbf{F}}$ of each level $v_{h,j}$ within each such sequence, every configuration matrix can be constructed in a specific way.[9] Let the number of level sequences for each condition factor $j = \mathrm{z}$, $seq_{\mathrm{z}}^{\mathbf{F}}$, be determined by Eq. (3):

$$seq_{\mathrm{z}}^{\mathbf{F}} = \frac{d}{\prod_{j=\mathrm{z}}^{k} p_j} = \frac{\prod_{j=1}^{k} p_j}{\prod_{j=\mathrm{z}}^{k} p_j} = \prod_{j=1}^{\mathrm{z}-1} p_j. \tag{3}$$

Furthermore, let the number of repetitions $rep_{\mathrm{z}}^{\mathbf{F}}$ for any level $v_{h,j}$ with $j = \mathrm{z}$ be given by Eq. (4):

$$rep_{\mathrm{z}}^{\mathbf{F}} = \frac{d}{\prod_{j=1}^{\mathrm{z}} p_j} = \frac{\prod_{j=1}^{k} p_j}{\prod_{j=1}^{\mathrm{z}} p_j} = \prod_{j=\mathrm{z}+1}^{k} p_j. \tag{4}$$

Eqs. (3) and (4) imply that $rep_k^{\mathbf{F}} = seq_1^{\mathbf{F}} = 1$ will always be true. By Eqs. (1) to (4), the generic function table shown in Table 1 results. For $k \geq 3$, and any condition factors $j = 1$ and $j = k - 1$, it must be true that $rep_1^{\mathbf{F}} \geq 4$ and $rep_{k-1}^{\mathbf{F}} \geq 2$. For example, three condition factors $\mathbf{X}_1$, $\mathbf{X}_2$ and $\mathbf{X}_3$, with $p_1 = 2$, $p_2 = 3$ and $p_3 = 2$, generate a table of functions $\mathbf{F}_{12\times(3+1)}$ with $seq_1^{\mathbf{F}} = 1$ and $rep_1^{\mathbf{F}} = 6$, $seq_2^{\mathbf{F}} = 2$ and $rep_2^{\mathbf{F}} = 2$, and $seq_3^{\mathbf{F}} = 6$ and $rep_3^{\mathbf{F}} = 1$.

In order to complete $\mathbf{F}$, let the $(k+1)$th to $(k+w)$th columns of Table 1 be denoted by $\mathcal{Z}_i(\mathbf{O}_1\{v_h\})$, $\mathcal{Z}_i(\mathbf{O}_2\{v_h\})$, …, $\mathcal{Z}_i(\mathbf{O}_w\{v_h\})$, which represent binary *function values* of an operation on all levels of the condition factors and exactly one level each of a set of outcome factors $\mathbf{O}_1$, $\mathbf{O}_2$, …, $\mathbf{O}_w$. In QCA, this operation is usually called *consistency* (Ragin, 2006) or *inclusion* (Thiem & Duşa, 2013b, 2013c), and posits a subset relation between each configuration and an outcome. Note that the outcome itself, that is, the level of the outcome factor under consideration, does not show up in the function table because there can be two or more instances of any specific configuration that coincide with different levels of any specific outcome factor. For the sake of simplicity and without loss of generality, we limit ourselves to designs with single outcomes $\mathcal{Z}(\mathbf{O}\{v_h\})$ in the remainder of the article.

Once $\mathcal{Z}(\mathbf{O}\{v_h\})$ has been determined for all $c_i \in \mathcal{C}^{\mathbf{F}}$, the information contained in $\mathbf{F}$ can be expressed by means of an output function f as given by Eq. (5):

$$f(\mathbf{O}\{v_h\}) = \mathcal{Z}_1 c_1 \cup \mathcal{Z}_2 c_2 \cup \ldots \cup \mathcal{Z}_d c_d = \bigcup_{i=1}^{d} \mathcal{Z}_i c_i\,. \tag{5}$$

Every product $\mathcal{Z}_i c_i$ in f is a *complete product* ($\mathcal{F}$). Complete products which correspond to configurations for which $\mathcal{Z}(\mathbf{O}\{v_h\}) = 1$ is true are called *positive* and will be denoted by $\mathcal{F}^1$. In contrast, complete products for which $\mathcal{Z}(\mathbf{O}\{v_h\}) = 0$ is true

[9]Configuration matrices can also be constructed by representing its rows as numbers from 0 to $d-1$ using a binary, tertiary, etc., number system. For example, with four Boolean condition factors, the last row (15) in the configuration matrix would contain the combination $\mathbf{X}_1\{1\}\mathbf{X}_2\{1\}\mathbf{X}_3\{1\}\mathbf{X}_4\{1\}$ because $1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 15$. In order not to have to switch between different number systems, we have opted for an arithmetic approach to presentation.

TABLE 1 Generic Function Table $\mathbf{F}_{d\times(k+w)}$

						$\mathcal{Z}_i$			
c_i	X_1	...	X_j	...	X_k	$O_1\{v_h\}$	$O_2\{v_h\}$	...	$O_w\{v_h\}$
1	v_1	...	v_1	...	v_1	...	...	...	...
2	v_1	⋮	v_1	⋮	v_2	⋮	⋮	⋮	⋮
⋮	v_1	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	v_1	⋮	v_2	⋮	v_p	⋮	⋮	⋮	⋮
⋮	⋮	⋮	v_2	⋮	v_1	⋮	⋮	⋮	⋮
⋮	v_2	⋮	⋮	⋮	v_2	⋮	⋮	⋮	⋮
⋮	v_2	⋮	v_p	⋮	⋮	⋮	⋮	⋮	⋮
⋮	v_2	⋮	v_p	⋮	v_p	⋮	⋮	⋮	⋮
⋮	v_2	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	v_p	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	v_p	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	v_p	⋮	v_p	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
d	v_p	...	v_p	...	v_p	...	...	...	...

are called *negative* and will be denoted by $\mathcal{F}^0$. Positive-complete products in QCA result if, and only if, $\mathcal{F}\subseteq\mathbf{O}\{v_h\}$ according to the inclusion score determining $\mathcal{Z}$, whereas negative-complete products result if, and only if, $\mathcal{F}\not\subseteq\mathbf{O}\{v_h\}$.[10]

Output functions derived from function tables are often unnecessarily complex because they include irrelevant information with regards to $\mathbf{O}\{v_h\}$, where "irrelevant" is taken to mean information whose elimination leaves the function value unchanged. Several methods of reducing output functions have been devised over decades of research in propositional logic and switching circuit theory. For example, in problems of low to moderate complexity with $k\leq 6$, Veitch-Karnaugh maps provide a useful graphical means (Hohn, 1966, pp. 170–201).[11] However, a more sophisticated device has been proposed in the form of the QMC algorithm (McCluskey, 1956; Quine, 1952).

QMC's core procedure consists in the repeated use of the general Boolean-algebraic theorem $\Phi\{1\}\Psi\{1\}+\Phi\{1\}\Psi\{0\}=\Phi\{1\}(\Psi\{1\}+\Psi\{0\})=\Phi\{1\}(1)=\Phi\{1\}$, where Φ and Ψ denote Boolean expressions of unspecified complexity, in order to transform an unreduced output function containing only positive-complete products to a minimal yet equivalent representation of the same function by progression through a limited number of transitory states. The first of these states is reached

[10]We disregard cases of *contradictions* (not to be understood in the logical sense). Contradictions describe an indeterminate state between $\mathcal{Z}(\mathbf{O}\{v_h\})=1$ and $\mathcal{Z}(\mathbf{O}\{v_h\})=0$ for which analysts are reluctant to assign binary function values.

[11]For an application of such maps in multivalue QCA, see Thiem (2015).

by combining all pairs $\{\mathcal{F}^1_{i=z}, \mathcal{F}^1_{i\neq z}\}$ that differ by level on one and the same condition factor only across all levels of this factor to successively form less complex equivalents of f. Any product resulting from a paired simplification of two positive-complete products is then called a *proper implicant* ($\mathcal{I}^1$).

More precisely, a proper implicant $\mathcal{I}^1$ of f with respect to $\mathcal{F}^1$ is a product of condition factor levels $\bigcap_j^m \mathbf{X}_j\{v_h\}$ with $m < k$, which is included in f, and which has the property that, if any level $\mathbf{X}_j\{v_h\}$ is eliminated from this product, then the remaining product is still included in f. More generally, a function $g_1(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_k)$ is said to *include* another function $g_2(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_k)$, written $g_1 \supseteq g_2$ or $g_2 \subseteq g_1$, if, for each combination of levels for which $g_2 = 1$ is true, $g_1 = 1$ is also true.

Once proper implicants have been derived after the process of minimizing all positive-complete products, they can usually be simplified further. QMC is an iterative algorithm involving the systematic and exhaustive comparison of all positive-complete products in the initiating pass, and over the following iterations the comparison of all proper implicants until no more condition factors can be absorbed. The surviving proper implicants at the end of this chain are then called *prime implicants* ($\mathcal{P}^1$). A prime implicant $\mathcal{P}^1$ of f with respect to $\mathcal{F}^1$ is a product of conditon factor levels $\bigcap_j^m \mathbf{X}_j\{v_h\}$ with $m < k$, which is included in f, and which has the property that, if any level $\mathbf{X}_j\{v_h\}$ is eliminated from this product, then the remaining product is not included anymore in f.[12] Once the final iteration is completed and the set of prime implicants has been generated, a prime implicant chart is constructed and solved according to the constraints specified by the analyst.[13]

Between two and four condition factors, QMC can still be performed by pen-and-paper procedures without much effort, but the geometric growth rate in the number of paired comparisons dictates the use of tailored software beyond this range. Four bivalent condition factors generate a function table of only 16 rows, but 10 such factors already create no fewer than 1024 configurations. In the `fs/QCA` software, for example, QMC imposes a computational ceiling on model complexity at about 12 condition factors. In the remainder of this article, we introduce an alternative approach that circumvents the brake put on QMC through the iterative comparison of implicants within high-dimensional matrices.

3. THE SOLUTION TO QMC'S SHORTCOMINGS

We propose a novel solution to the problem of minimizing output functions. Since this approach complements QMC rather than replacing it entirely, we call it the *enhanced* Quine-McCluskey (*e*QMC) algorithm. The first subsection presents the fundamentals of *e*QMC before the second subsection elaborates on the way in which the algorithm deals with logical remainders. Finally, the third subsection explains how *e*QMC identifies prime implicants.

[12] In the extreme case where no Boolean minimization is possible, $\mathcal{F}^1 = \mathcal{I}^1 = \mathcal{P}^1$.

[13] For more detailed explanations of prime implicant charts, see Edwards (1973, p. 98ff.), Lewin and Protheroe (1992, p. 76ff.), and McCluskey (1965, p. 140ff.).

3.1. Fundamentals of *e*QMC

The approach of *e*QMC is based on two seemingly trivial observations. First, every positive-complete product $\mathcal{F}_i^1$ is a (proper) subset of a least one prime implicant $\mathcal{P}_i^1$. Second, no negative-complete product $\mathcal{F}_i^0$ is a subset of any prime implicant (Duşa, 2007). On the basis of these two interconnected facts, *e*QMC performs an exhaustive procedure that relies on index vectors instead of complex matrices. These index vectors are generated from an auxiliary device called *implicant matrix*.

Implicant matrices and function tables look similar and partly overlap in the information they contain, but they consist of different representations of this information. While each cell in a function table usually records a condition factor level, either 0 or 1 in the case of Boolean tables, cells in implicant matrices collect factor level indices.[14] For each condition factor, this index is augmented by a degenerate zero-value indicating the elimination of this factor. Generally defined, an implicant matrix $\Omega_{q\times(1+k+1)}$ is a two-dimensional array of q rows and $1+k+1$ columns. The rows not only contain level index representations of all complete products $\mathcal{F}$, but also of all proper implicants $\mathcal{I}$ and prime implicants $\mathcal{P}$. Since $\Omega_{q\times(1+k+1)}$ includes configurations and prime implicants as limiting constructs, it is called an implicant matrix.

Similar to the principles of setting up function tables introduced in Eqs. (1) to (4) above, let the number of implicants q in Ω be determined by the adjusted product of the total number of levels of all condition factors as given in Eq. (6):

$$q = (p_1 + 1) \cdot (p_2 + 1) \cdot \ldots \cdot (p_j + 1) \cdot \ldots \cdot (p_k + 1) = \prod_{j=1}^{k} (p_j + 1). \tag{6}$$

Based on the number of level index sequences seq_j^{Ω} within j and the number of repetitions rep_j^{Ω} of each level index value h within each such sequence, every implicant matrix can be constructed in a specific way. Let the number of level index sequences for each condition factor $j = \mathrm{z}$, $seq_{\mathrm{z}}^{\Omega}$, be given by Eq. (7):

$$seq_{\mathrm{z}}^{\Omega} = \frac{q}{\prod_{j=1}^{k} (p_j + 1)} = \frac{\prod_{j=1}^{k} (p_j + 1)}{\prod_{j=1}^{k} (p_j + 1)} = \prod_{j=1}^{\mathrm{z}-1} (p_j + 1). \tag{7}$$

Furthermore, let the number of repetitions rep_j^{Ω} for any level index value h_j with $j = \mathrm{z}$ be given by Eq. (8):

$$rep_{\mathrm{z}}^{\Omega} = \frac{q}{\prod_{j=1}^{\mathrm{z}} (p_j + 1)} = \frac{\prod_{j=1}^{k} (p_j + 1)}{\prod_{j=1}^{\mathrm{z}} (p_j + 1)} = \prod_{j=\mathrm{z}+1}^{k} (p_j + 1). \tag{8}$$

By Eqs. (6) to (8), the generic implicant matrix in Table 2 results. The leftmost column represents the implicant r_i with $i = 1, 2, \ldots, q$. Thus, implicant matrices are comprised of all possible supersets q that can be formed from a given set of k

[14]Potentially, this could also apply to function tables, but as this solution is not generally considered, we draw an explicit distinction between factors and their level indices.

TABLE 2 Generic Implicant Matrix $\Omega_{q\times(1+k+1)}$

r_i	h_1	...	h_j	...	h_k	$\mathcal{F}$	Implicant
1	0	⋮	0	⋮	0	⋮	$\emptyset$
2	0	⋮	0	⋮	1	⋮	$\ldots\cap\ldots\cap\mathbf{X}_k\{v_1\}$
⋮	0	⋮	0	⋮	2	⋮	$\ldots\cap\ldots\cap\mathbf{X}_k\{v_2\}$
⋮	0	⋮	⋮	⋮	⋮	⋮	$\ldots\cap\mathbf{X}_j\{\cdot\}\cap\ldots\cap\mathbf{X}_k\{\cdot\}$
⋮	0	⋮	1	⋮	p_k	⋮	$\ldots\cap\mathbf{X}_j\{v_1\}\cap\ldots\cap\mathbf{X}_k\{v_p\}$
⋮	0	⋮	1	⋮	0	⋮	$\ldots\cap\mathbf{X}_j\{v_1\}\cap\ldots$
⋮	0	⋮	1	⋮	1	⋮	$\ldots\cap\mathbf{X}_j\{v_1\}\cap\ldots\cap\mathbf{X}_k\{v_1\}$
⋮	0	⋮	⋮	⋮	2	⋮	$\ldots\cap\mathbf{X}_j\{\cdot\}\cap\ldots\cap\mathbf{X}_k\{v_2\}$
⋮	0	⋮	2	⋮	⋮	⋮	$\ldots\cap\mathbf{X}_j\{v_2\}\cap\ldots\cap\mathbf{X}_k\{\cdot\}$
⋮	⋮	⋮	2	⋮	p_k	⋮	$\mathbf{X}_1\{\cdot\}\cap\ldots\cap\mathbf{X}_j\{v_2\}\cap\ldots\cap\mathbf{X}_k\{v_p\}$
⋮	1	⋮	2	⋮	⋮	⋮	$\mathbf{X}_1\{v_1\}\cap\ldots\cap\mathbf{X}_j\{v_2\}\cap\ldots\cap\mathbf{X}_k\{\cdot\}$
⋮	⋮	⋮	⋮	⋮	⋮	⋮	$\mathbf{X}_1\{\cdot\}\cap\ldots\cap\mathbf{X}_j\{\cdot\}\cap\ldots\cap\mathbf{X}_k\{\cdot\}$
⋮	2	⋮	p_j	⋮	⋮	⋮	$\mathbf{X}_1\{v_2\}\cap\ldots\cap\mathbf{X}_j\{v_p\}\cap\ldots\cap\mathbf{X}_k\{\cdot\}$
⋮	⋮	⋮	⋮	⋮	⋮	⋮	$\mathbf{X}_1\{\cdot\}\cap\ldots\cap\mathbf{X}_j\{\cdot\}\cap\ldots\cap\mathbf{X}_k\{\cdot\}$
q	p_1	...	p_j	...	p_k	⋮	$\mathbf{X}_1\{v_p\}\cap\ldots\cap\mathbf{X}_j\{v_p\}\cap\ldots\cap\mathbf{X}_k\{v_p\}$

condition factors, including the empty set in r_1, and all configurations c_1 to c_d. The full condition factor level representation of each implicant is provided in the rightmost column for illustrative purposes, but it forms no essential part of Ω.

Each configuration in Ω thus possesses a finite number of nontrivial implicants s_k, which is determined by the sum of binomial coefficients defined in Eq. (9):

$$s_k = \binom{k}{1} + \binom{k}{2} + \cdots + \binom{k}{j} + \cdots + \binom{k}{k} = \sum_{j=1}^{k} \frac{k!}{j!(k-j)!} = 2^k - 1\,. \quad (9)$$

Note that s_k only depends on k, not on p_j. For example, every generic configuration $\mathbf{X}_1\{\cdot\}\cap\mathbf{X}_2\cap\{\cdot\}\mathbf{X}_3\{\cdot\}$ has exactly seven implicants: $\mathbf{X}_1\{\cdot\}$, $\mathbf{X}_2\{\cdot\}$, $\mathbf{X}_3\{\cdot\}$, $\mathbf{X}_1\{\cdot\}\cap\mathbf{X}_2\{\cdot\}$, $\mathbf{X}_1\{\cdot\}\cap\mathbf{X}_3\{\cdot\}$, $\mathbf{X}_2\{\cdot\}\cap\mathbf{X}_3\{\cdot\}$ and $\mathbf{X}_1\{\cdot\}\cap\mathbf{X}_2\{\cdot\}\cap\mathbf{X}_3\{\cdot\}$. As illustrated in the Venn diagram in Figure 1, these implicants also form the following set relations among themselves: $\mathbf{X}_1\{\cdot\}\supset\mathbf{X}_1\{\cdot\}\cap\mathbf{X}_2\{\cdot\}\cup\mathbf{X}_1\{\cdot\}\cap\mathbf{X}_3\{\cdot\}$, $\mathbf{X}_2\{\cdot\}\supset\mathbf{X}_1\{\cdot\}\cap\mathbf{X}_2\{\cdot\}\cup\mathbf{X}_2\{\cdot\}\cap\mathbf{X}_3\{\cdot\}$, $\mathbf{X}_3\{\cdot\}\supset\mathbf{X}_1\{\cdot\}\cap\mathbf{X}_3\{\cdot\}\cup\mathbf{X}_2\{\cdot\}\cap\mathbf{X}_3\{\cdot\}$.

The index values of the rows in Ω that correspond to the implicants of a configuration can be computed by the following technique. A multiplication vector $\mathbf{m}$ is formed (Dušа, 2011), each of whose k elements equals the number of level index repetitions rep_j^{Ω} for the corresponding condition factor as given in Eq. (10):

$$\mathbf{m} = \left(\prod_{j=2}^{k}(p_j+1), \prod_{j=3}^{k}(p_j+1), \ldots, \prod_{j=k}^{k}(p_j+1), 1\right) = (m_1, m_2, \ldots, m_k). \quad (10)$$

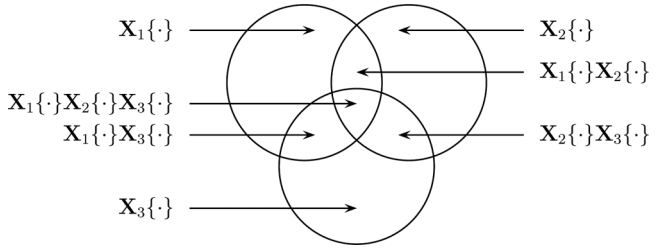

FIGURE 1 Set relations among configuration implicants.

The multiplication vector $\mathbf{m}$ is then used to form the Hadarmard product with each row vector of condition factor level indices $r_i(\mathcal{F}^1)$ that represents a positive-complete product. This product is defined by Eq. (11):

$$r_i\left(\mathcal{F}^1\right) \bullet \mathbf{m} = (h_{i,1}m_1, h_{i,2}m_2, \ldots, h_{i,k}m_k). \tag{11}$$

All $2^k - 1$ unique sums that can be formed from this product then yield the row index vector $\mathbf{r}_1^1 = (r_{1,1}^1, r_{1,2}^1, \ldots, r_{1,2^{k-1}}^1)$ for the first of all x $\mathcal{F}^1$ after a unit vector $\mathbf{u}$ has been added.[15] For example, if $\mathbf{X}_1$ is bivalent, and both $\mathbf{X}_2$ and $\mathbf{X}_3$ are trivalent condition factors, then $\mathbf{m} = (16, 4, 1)$. If the configuration $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{2\} \cap \mathbf{X}_3\{2\}$ in c_{18} of $\mathbf{F}_{18\times4}$ is positive-complete as $\mathcal{F}_{18}^1$, then the Hadarmard product yields $r_{48}\bullet\mathbf{m} = (2 \cdot 16, 3 \cdot 4, 3 \cdot 1) = (32, 12, 3)$ because $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{2\} \cap \mathbf{X}_3\{2\}$ is always found in $r_{48} = (2, 3, 3)$ of $\Omega_{48\times5}$. All seven possible sums that can be formed from this product then generate the row index vector $\mathbf{r}_1^1 = (33, 13, 4, 45, 36, 16, 48)$ after $\mathbf{u}$ has been added. This procedure is repeated for all x positive-complete products. The set union of all row index vectors $\mathbf{r}_1^1, \mathbf{r}_2^1, \ldots, \mathbf{r}_{x_1}^1$ yields the *positive implicant vector* $\mathbf{r}^1$ because all its elements are associated implicants of positive-complete products.

While QMC only processes $\mathcal{F}^1$, *e*QMC also requires row index vectors of $\mathcal{F}^0$. The multiplication vector $\mathbf{m}$ is thus recycled to form the Hadarmard product with each row vector $r_i(\mathcal{F}^0)$. This product is defined as given in Eq. (12).

$$r_i\left(\mathcal{F}^0\right) \bullet \mathbf{m} = (h_{i,1}m_1, h_{i,2}m_2, \ldots, h_{i,k}m_k) \tag{12}$$

All $2^k - 1$ unique sums that can be formed from this product then yield the row index vector $\mathbf{r}_1^0 = (r_{1,1}^0, r_{1,2}^0, \ldots, r_{1,2^{k-1}}^0)$ for the first of all y $\mathcal{F}^0$ after $\mathbf{u}$ has been added. For example, if configuration $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{2\} \cap \mathbf{X}_3\{0\}$ in c_{16} of $\mathbf{F}_{18\times4}$ is negative-complete as $\mathcal{F}_{16}^0$, then $r_{32} \bullet \mathbf{m} = (2 \cdot 16, 3 \cdot 4, 1 \cdot 1) = (32, 12, 1)$ because this configuration is always found in $r_{46} = (2, 3, 1)$ of $\Omega_{48\times5}$. All seven possible sums that can be formed from this product then generate the row index vector $\mathbf{r}_1^0 = (33, 13, 2, 45, 34, 14, 46)$. This procedure is repeated for all y $\mathcal{F}^0$. The union of all row index vectors $\mathbf{r}_1^0, \mathbf{r}_2^0, \ldots, \mathbf{r}_{y_1}^0$ yields the negative implicant vector $\mathbf{r}^0$ because all its elements are associated implicants of negative-complete products.

[15]As Eq. (9) is computed in base 10, which starts with 0, but r_i starts with $i = 1$, the unit vector has to be added.

The relative complement of $\mathbf{r}^0$ in $\mathbf{r}^1$ yields the implicant difference vector $\mathbf{r}^D$. For example, if the two instances of $\mathbf{r}_1^0$ and $\mathbf{r}_1^1$ given above were the only row index vectors such that $\mathbf{r}_1^0 = \mathbf{r}^0$ and $\mathbf{r}_1^1 = \mathbf{r}^1$, then $\mathbf{r}^D = (4, 36, 16, 48)$ because $\mathbf{r}^1 \backslash \mathbf{r}^0 = (33, 13, 4, 45, 36, 16, 48) \backslash (33, 13, 2, 45, 34, 14, 46) = (4, 36, 16, 48)$. Row r_4 corresponds to $\mathbf{X}_3\{2\}$, r_{16} to $\mathbf{X}_2\{2\} \cap \mathbf{X}_3\{2\}$ and r_{36} to $\mathbf{X}_1\{1\} \cap \mathbf{X}_3\{2\}$, all three of which are proper implicants of $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{2\} \cap \mathbf{X}_3\{2\}$, but not of $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{2\} \cap \mathbf{X}_3\{0\}$. The proper implicant $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{2\}$ corresponding to $r_{45} = (2, 3, 0)$ cannot be part of any solution because it is included by both $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{2\} \cap \mathbf{X}_3\{2\}$ as well as $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{2\} \cap \mathbf{X}_3\{0\}$. In the case of $\mathbf{r}^D$ containing only row index values that of positive-complete products, then f is non-minimizable and the output function is free from redundancies.

3.2. Dealing with Logical Remainders

In an empirically ideal world, where "ideal" should be taken to mean "fully saturated," researchers would have perfect information about $\mathcal{Z}_i(\mathbf{O}\{v_h\})$. However, not only is social reality empirically limited, but the very idea of systematic configurational comparison has from the start focused on small numbers of cases, as a result of which the number of configurations often exceeds the number of objects under investigation, sometimes considerably so.[16] It will therefore usually prove impossible to assign binary function values to the vast majority of configurations based on empirics. A configuration without a function value is called a *logical remainder*, or a *don't care*, in switching circuit theory. A *don't care* implies that Boolean minimization does not factor in whether terminal gates are open or not, as long as a less complex gate-lead structure can be obtained. The same principle holds in the social sciences. Even if the function values of logical remainders are unknown, researchers assume, based on counterfactual reasoning, that these values could be established were these configurations to be observed in social reality.

Logical remainders are responsible for the exponential increase in the complexity of the output function f. The *e*QMC algorithm alleviates this problem because only observed configurations and their implicants are used in deriving the implicant difference vector. As a result, the inclusion of logical remainders, irrespective of the criteria on which this inclusion is based, has no effect whatsoever on the amount of information that has to be processed by *e*QMC. Only as the number of logical remainders which are coded as negative-complete products increases does *e*QMC begin to lose its competitive advantage. In this case, *e*QMC incorporates their implicants into the negative implicant vector, which in turn reduces its performance. If all logical remainders are assigned a negative function value (a strategy called the *complex* or *conservative* solution), QMC is clearly superior to *e*QMC in performance, all the more so with increasing numbers of condition factors. This implies that QMC and *e*QMC work best when complemented by each other. QMC's resource consumption increases with the number of remainders made available for minimization, whereas that of *e*QMC increases relative to QMC with the number of remainders that are excluded from it.

[16]The analysis by Cress and Snow (1996) on homeless social movement organizations provides an extreme example of a mismatch between cases and configurations. It features 15 objects which cover 10 configurations, but 14 conditions create a property space of no fewer than 16,384 configurations.

3.3. Identifying Prime Implicants

No exact formula for computing the number of implicants exists, but the number of prime implicants can grow as large as $3^k/k$ (Brayton et al., 1984). The implicant difference vector $\mathbf{r}^{\mathrm{D}}$ is computed using an exhaustive method, but many implicants are not prime implicants. However, Ω is constructed in such a way that for any two implicants $\mathcal{I}_a$ in r_a and $\mathcal{I}_b$ in r_b for which $\mathcal{I}_a \supset \mathcal{I}_b$ is true, $a < b$ will also be true. *e*QMC thus proceeds by sorting $\mathbf{r}^{\mathrm{D}} = (r_1^{\mathrm{D}}, r_2^{\mathrm{D}}, \ldots, r_{x_{\mathrm{D}}}^{\mathrm{D}})$ such that $r_1^{\mathrm{D}} < r_2^{\mathrm{D}} < \ldots < r_{x_{\mathrm{D}}}^{\mathrm{D}}$.

All row index values that correspond to proper subsets of an element in $\mathbf{r}^{\mathrm{D}}$ are then eliminated in succession, beginning with r_1^{D}. The procedure first identifies all $h_j = 0$ in $r_{r_1^{\mathrm{D}}}$ before collecting together all row index values that result from the consecutive addition of the corresponding element in the multiplication vector $\mathbf{m}$ over all t columns in Ω for which $h_j = 0$ is true. Formally, this vector $\mathbf{a}_{r_1^{\mathrm{D}}} = (a_1 \cup a_2 \cup \cdots \cup a_t)$ is created as the union of all unique elements resulting from a series of sub-vector Eqs.. If for any $a \in \mathbf{a}$ except r_1^{D}, $\mathcal{I}_{r_1^{\mathrm{D}}} \supset \mathcal{I}_a$ and $a \in \mathbf{r}^{\mathrm{D}}$, then a is eliminated from $\mathbf{r}^{\mathrm{D}}$. With the shortened implicant difference vector $\mathbf{r}_1^{\mathrm{D}}$, the same procedure is repeated starting with r_2^{D}. If for any $a \in \mathbf{a}$ except r_2^{D}, $\mathcal{I}_{r_2^{\mathrm{D}}} \supset \mathcal{I}_a$ and $a \in \mathbf{r}_1^{\mathrm{D}}$, then a is eliminated from $\mathbf{r}_1^{\mathrm{D}}$, yielding $\mathbf{r}_2^{\mathrm{D}}$, and so on.

An example with three bivalent condition factors $\mathbf{X}_1$, $\mathbf{X}_2$ and $\mathbf{X}_3$ again serves to illustrate these steps. Consider an (already ordered) implicant difference vector $\mathbf{r}^{\mathrm{D}} = (2, 5, 11, 14, 19, 21, 22, 24, 25, 27)$ derived from the implicant matrix $\Omega_{27\times5}$ shown in Table 4. As $r_1^{\mathrm{D}} = 2$, *e*QMC starts with $r_2 = (0, 0, 1)$, which corresponds to $\mathbf{X}_3\{0\}$ (bold; dashed cell borders), and first finds all zero-values in that row (underlined). These appear in columns h_1 and h_2. Starting with the rightmost column h_2, its corresponding vector a_1 is given by $a_1 = (2,\ 2+1\cdot3,\ 2+2\cdot3) = (2,\ 5,\ 8)$ because $m_2 = 3$ and $p_2 = 2$. Row r_5 corresponds to $\mathbf{X}_2\{0\} \cap \mathbf{X}_3\{0\}$ and r_8 to $\mathbf{X}_2\{1\} \cap \mathbf{X}_3\{0\}$, both of which are subsets of $\mathbf{X}_3\{0\}$ only involving condition factor $\mathbf{X}_2$ in addition.

The next vector a_2 is associated with h_1, where the second zero-value appears. As the corresponding multiplier from $\mathbf{m}$ is $m_1 = 9$, $a_2 = (2+1\cdot9,\ 5+1\cdot9,\ 8+1\cdot9,\ 2+2\cdot9,\ 5+2\cdot9,\ 8+2\cdot9) = (11, 14, 17, 20, 23, 26)$. The union of a_1 and a_2 yields $a_1 \cup a_2 = \mathbf{a} = (2, 5, 8, 11, 14, 17, 20, 23, 26)$. All of these subsets of $\mathbf{X}_3\{0\}$ are indicated by "2" in the column "$\supset$" of Table 3. Since 5, 11, and 14 are also elements in $\mathbf{r}^{\mathrm{D}}$ and their corresponding implicants $\mathcal{I}_5$, $\mathcal{I}_{11}$ and $\mathcal{I}_{14}$ are all subsets of $\mathcal{I}_2$, $\mathbf{r}^{\mathrm{D}}$ is shortened to $\mathbf{r}_1^{\mathrm{D}} = (2, 19, 21, 22, 24, 27)$. When repeating this procedure for r_{19}—the next element in $\mathbf{r}_1^{\mathrm{D}}$ after 2—the final implicant difference vector $\mathbf{r}_2^{\mathrm{D}} = (2, 19)$ results. Both corresponding prime implicants $\mathbf{X}_3\{0\}$ (r_2) and $\mathbf{X}_1\{1\}$ (r_{19}) are essential because the former covers $\mathbf{X}_1\{0\} \cap \mathbf{X}_2\{0\} \cap \mathbf{X}_3\{0\}$ (r_{14}), whereas the latter covers $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{0\} \cap \mathbf{X}_3\{1\}$ (r_{24}) and $\mathbf{X}_1\{1\} \cap \mathbf{X}_2\{1\} \cap \mathbf{X}_3\{1\}$ (r_{27}). Thus, the solution to the minimization is $\mathbf{X}_1\{1\} \cup \mathbf{X}_3\{0\}$. This approach of eliminating implicants is efficient because $\mathbf{r}^{\mathrm{D}}$ is shortened progressively, reaching its final form in the smallest possible number of iterations. At the end, it contains only the set of prime implicants. In the last step of *e*QMC, a standard linear optimization routine solves the prime implicant chart. Strictly speaking, it is thus no integral part of the algorithm. We briefly summarize the steps of *e*QMC again in Table 4.

TABLE 3 Hypothetical Implicant Matrix $\Omega_{27\times 5}$

r_i	h_1	h_2	h_3	Implicant	$\mathcal{F}$	$\supset$
1	0	0	0	$\emptyset$		
2	**0**	**0**	**1**	$\mathbf{X}_3\{0\}$		**2**
3	0	0	2	$\mathbf{X}_3\{1\}$		
4	0	1	0	$\mathbf{X}_2\{0\}$		
5	0	1	1	$\mathbf{X}_2\{0\}\mathbf{X}_3\{0\}$		2
6	0	1	2	$\mathbf{X}_2\{0\}\mathbf{X}_3\{1\}$		
7	0	2	0	$\mathbf{X}_2\{1\}$		
8	0	2	1	$\mathbf{X}_2\{1\}\mathbf{X}_3\{0\}$		2
9	0	2	2	$\mathbf{X}_2\{1\}\mathbf{X}_3\{1\}$		
10	1	0	0	$\mathbf{X}_1\{0\}$		
11	1	0	1	$\mathbf{X}_1\{0\}\ \mathbf{X}_3\{0\}$		2
12	1	0	2	$\mathbf{X}_1\{0\}\ \mathbf{X}_3\{1\}$		
13	1	1	0	$\mathbf{X}_1\{0\}\mathbf{X}_2\{0\}$		
14	1	1	1	$\mathbf{X}_1\{0\}\mathbf{X}_2\{0\}\mathbf{X}_3\{0\}$	1	2
15	1	1	2	$\mathbf{X}_1\{0\}\mathbf{X}_2\{0\}\mathbf{X}_3\{1\}$	0	
16	1	2	0	$\mathbf{X}_1\{0\}\mathbf{X}_2\{1\}$		
17	1	2	1	$\mathbf{X}_1\{0\}\mathbf{X}_2\{1\}\mathbf{X}_3\{0\}$		2
18	1	2	2	$\mathbf{X}_1\{0\}\mathbf{X}_2\{1\}\mathbf{X}_3\{1\}$	0	
19	**2**	**0**	**0**	$\mathbf{X}_1\{1\}$		**19**
20	2	0	1	$\mathbf{X}_1\{1\}\ \mathbf{X}_3\{0\}$		2,19
21	2	0	2	$\mathbf{X}_1\{1\}\ \mathbf{X}_3\{1\}$		19
22	2	1	0	$\mathbf{X}_1\{1\}\mathbf{X}_2\{0\}$		19
23	2	1	1	$\mathbf{X}_1\{1\}\mathbf{X}_2\{0\}\mathbf{X}_3\{0\}$		2,19
24	2	1	2	$\mathbf{X}_1\{1\}\mathbf{X}_2\{0\}\mathbf{X}_3\{1\}$	1	19
25	2	2	0	$\mathbf{X}_1\{1\}\mathbf{X}_2\{1\}$		19
26	2	2	1	$\mathbf{X}_1\{1\}\mathbf{X}_2\{1\}\mathbf{X}_3\{0\}$		2,19
27	2	2	2	$\mathbf{X}_1\{1\}\mathbf{X}_2\{1\}\mathbf{X}_3\{1\}$	1	19

TABLE 4 Summary of Steps Performed by *e*QMC

Step	Task
1	Set up implicant matrix $\Omega_{q\times(1+k+1)}$ from function table $\mathbf{F}_{d\times(k+1)}$
2	Compute multiplication vector $\mathbf{m}$ from level index repetitions rep_j^{Ω}
3	Compute positive implicant vector $\mathbf{r}^1$ from row index vectors for all $\mathcal{F}^1$
4	Compute negative implicant vector $\mathbf{r}^0$ from row index vectors for all $\mathcal{F}^0$
5	Compute implicant difference vector $\mathbf{r}^{\mathrm{D}}$ as relative complement $\mathbf{r}^1\backslash\mathbf{r}^0$
6	Sort implicant difference vector $\mathbf{r}^{\mathrm{D}}$ in ascending order
7	Eliminate redundant implicants from $\mathbf{r}^{\mathrm{D}}$
8	Construct and solve prime implicant chart from surviving elements of $\mathbf{r}^{\mathrm{D}}$

4. TESTING THE PERFORMANCE OF *e*QMC

In this section, the computational capabilities of *e*QMC in terms of speed and memory consumption by simulation are demonstrated by comparison with the capabilities of the most efficient implementation of QMC available

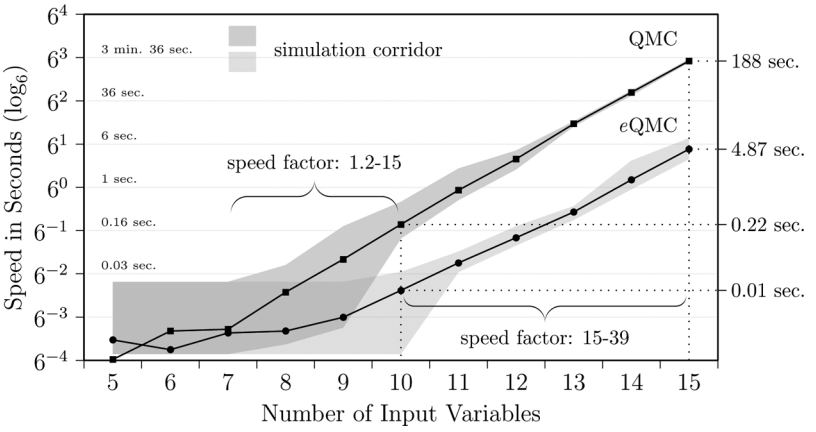

FIGURE 2 Speed comparison between QMC and *e*QMC.

(Duşa, 2010).[17] We let the number of bivalent condition factors k vary between 5 and 15. For each k, 50 random samples of 20 observed configurations are drawn, half of which are coded as positive-complete products, the other half as negative-complete products. This implies that with 5 condition factors, 37.5% of configurations represent logical remainders, whereas this figures increases to 99.9% with 15 condition factors. We assume this to be a reasonable approximation of many case distributions in empirical applications.

For each k, two variables are measured. First, we compute the average speed across all samples as seconds taken from the specification of the output function until the completion of the minimization process, which ends with the decomposition of the PI chart. For reasons of simplicity and comparability, we disregard ambiguities in the data and determine only the most economical solution. Second, we record random access memory (RAM) consumption in megabytes (MB). Figure 2 shows the performance of QMC and *e*QMC in terms of speed. Notice that for reasons of better readability, the scale of the ordinate has been transformed to $\log_6$. Between 7 and 10 condition factors, the speed advantage of *e*QMC over QMC varies between a factor of 1.2 and 15. At 10 condition factors, QMC completes the minimization in 0.2 s, whereas it takes *e*QMC barely 0.01 s. Below 7 condition factors, both QMC and *e*QMC are roughly on a par in speed, with only substantively irrelevant differences. This picture changes dramatically at the higher ends of model complexity. For Boolean functions with 10 to 15 condition factors, the speed advantage of *e*QMC increases to factors of between 15 and 39. These differentials are considerable. At 15 conditions factors, it takes QMC over 3 min to derive the solution, whereas *e*QMC does not even require 5 s for performing the same operation.

The results from the memory consumption simulation are presented in Figure 3, in which the scale of the ordinate has been transformed to $\log_7$. While

[17]This allows us to compare both algorithms up to 15 condition factors. We use a regular end-user machine with the following components: Microsoft Windows 7 64-bit operating system, 6 GB RAM, Intel Core i7-2640 M CPU, 2.80 GHz.

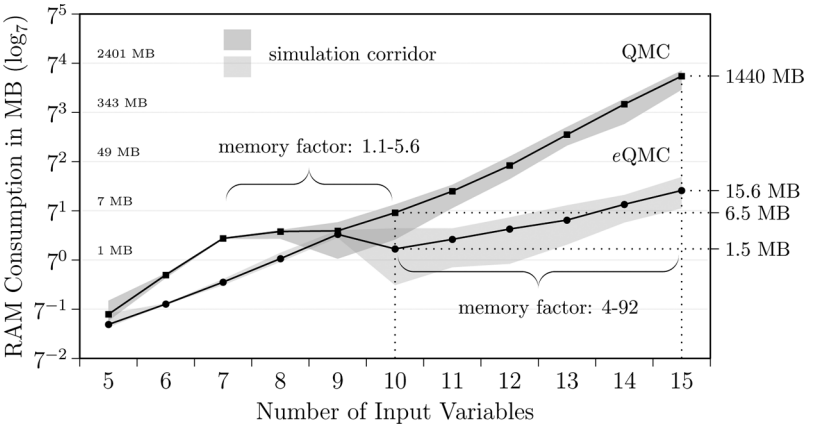

FIGURE 3 Memory consumption comparison between QMC and *e*QMC.

the advantage of *e*QMC over QMC is moderate and substantively unimportant at lower numbers of condition factors, the consumption gap widens enormously with increasing complexity. Between 7 and 10 condition factors, *e*QMC consumes only up to a sixth of the memory required by QMC, but beyond 10 condition factors, it plays out its strengths. At $k = 15$, memory consumption drops by a factor of around 90. In absolute numbers, QMC needs about 1.5 GB of RAM, whereas *e*QMC uses only 16 MB on average, with minimum requirements as low as 8 MB and maximum requirements not exceeding 30 MB.

In summary, the results from these simulations attest to the superiority of *e*QMC over QMC for social-scientific minimization problems of moderate to high complexity, whereas little is gained at lower ends. Thus, if complex problems have to be solved or if simulation requires a large number of minimization runs, for example, in order to estimate average result sensitivity effects, then the advantages of employing *e*QMC instead of QMC cannot be denied.

5. CONCLUSIONS

Sociologists and political scientists continue to further leverage the Boolean perspective on data analysis offered by QCA. However, while the basic logic of Boolean minimization has been well understood, the technicalities behind this process have so far been hidden from the eyes of the vast majority of social scientists. But it is these technicalities that set the limits of QCA's applicability, even though scientists rarely come close to reaching them. An important factor in this regard is the respective algorithm through which minimization is performed.

In this article, we first discussed the inherent weaknesses and strengths of the approach offered by the traditional QMC algorithm. If the minimization problem becomes increasingly complex, what with the inclusion of logical remainders into the output function, QMC slows down at a geometric rate and eventually uses up the computer's entire memory. In order to address these problems, we have presented the *e*QMC. Results obtained from simulations have shown memory

consumption and completion time to be reduced to an inconsiderable fraction of the amount and duration required by QMC in complex Boolean models. Over this range, *e*QMC pushes back the current boundaries of configurational comparative research.

All methodological enhancements come at some cost, and *e*QMC is no exception in this respect. As the set of configurations that are expressly excluded from the minimization process grows, *e*QMC becomes progressively slower until its superiority over QMC disappears. Our algorithm therefore enhances and extends QMC instead of replacing it fully. The exact location of the tipping point, the implications this carries for further algorithmic revisions, and possible integration with other approaches such as graph-based agents or coincidence analysis (Baumgartner, 2009, 2013) are aspects to be examined in future research.

ACKNOWLEDGMENTS

Previous versions of this article have been presented at the 2013 Annual Conference of the Swiss Political Science Association, University of Zurich (Switzerland), the 2013 Annual Conference of the Empirical Methods Section of the German Political Science Association, University of Konstanz (Germany) and the 2013 Annual Meeting of the American Political Science Association, Chicago (USA). An early draft of this article has been published as WP-2007-49 in the COMPASSS Working Paper series at www.compasss.org. We thank Barry Cooper, Charles Ragin and the two anonymous reviewers of this journal for very helpful comments and suggestions.

SUPPLEMENTAL MATERIAL

A replication file to this article is available from the corresponding author on request and archived in the Mathematical Sociology DataBank at http://www.mathematicalsociology.org/. It can also be accessed on the publisher's website at http://dx.doi.org/10.1080/0022250X.2014.897949

REFERENCES

Amenta, E., Caren, N., & Olasky, S. J. (2005). Age for leisure? Political mediation and the impact of the pension movement on U.S. old-age policy. *American Sociological Review*, *70*, 516–538.

Amenta, E., & Poulsen, J. D. (1994). Where to begin: A survey of five approaches to selecting independent variables for Qualitative Comparative Analysis. *Sociological Methods & Research*, *23*, 22–53.

Baumgartner, M. (2009). Inferring causal complexity. *Sociological Methods & Research*, *38*, 71–101.

Baumgartner, M. (2013). Detecting causal chains in small-n data. *Field Methods*, *25*, 3–24.

Berg-Schlosser, D., & De Meur, G. (1994). Conditions of democracy in interwar Europe: A Boolean test of major hypotheses. *Comparative Politics*, *26*, 253–279.

Brayton, R. K., Hachtel, G. D., McMullen, C. T., & Sangiovanni-Vincentelli, A. L. (1984). *Logic minimization algorithms for VLSI synthesis*. Boston, MA: Kluwer Academic.

Brueggemann, J., & Boswell, T. (1998). Realizing solidarity—Sources of interracial unionism during the Great Depression. *Work and Occupations*, *25*, 436–482.

Cornell, S., & Kalt, J. P. (2000). Where's the glue? Institutional and cultural foundations of American Indian economic development. *Journal of Socio-Economics*, *29*, 443–470.

Cress, D. M., & Snow, D. A. (1996). Mobilization at the margins: Resources, benefactors, and the viability of homeless social movement organizations. *American Sociological Review*, *61*, 1089–1109.

Cress, D. M., & Snow, D. A. (2000). The outcomes of homeless mobilization: The influence of organization, disruption, political mediation, and framing. *American Journal of Sociology*, *105*, 1063–1104.

Cronqvist, L., & Berg-Schlosser, D. (2009). Multi-value QCA (mvQCA). In B. Rihoux & C. C. Ragin (Eds.), *Configurational comparative methods: Qualitative Comparative Analysis (QCA) and related techniques* (pp. 69–86). London, UK: Sage Publications.

Dixon, M., Roscigno, V. J., & Hodson, R. (2004). Unions, solidarity, and striking. *Social Forces*, *83*, 3–33.

Duşa, A. (2007). *Enhancing Quine-McCluskey* (COMPASSS WP Series 2007-49). Retrieved from http://www.compasss.org/wpseries/Dusa2007b.pdf

Duşa, A. (2010). A mathematical approach to the Boolean minimization problem. *Quality & Quantity*, *44*, 99–113.

Duşa, A. (2011). *Enhancing Quine-McCluskey to multi-value minimization* Unpublished manuscript.

Duşa, A. & Thiem, A. (2014). *QCA: A package for Qualitative Comparative Analysis, R package* (Version 1.1-4). Retrieved from http://cran.r-project.org/package=QCA

Edwards, F. H. (1973). *The principles of switching circuits*. Cambridge, MA: MIT Press.

Hicks, A., Misra, J., & Ng, T. N. (1995). The programmatic emergence of the social security state. *American Sociological Review*, *60*, 329–349.

Hohn, F. E. (1966). *Applied Boolean algebra: An elementary introduction* (2nd ed.). New York, NY: Macmillan.

Lam, W., & Ostrom, E. (2010). Analyzing the dynamic complexity of development interventions: Lessons from an irrigation experiment in Nepal. *Policy Sciences*, *43*, 1–25.

Lewin, D., & Protheroe, D. (1992). *Design of logic systems* (2nd ed.). London, UK: Chapman & Hall.

McCluskey, E. J. (1956). Minimization of Boolean functions. *Bell Systems Technical Journal*, *35*, 1417–1444.

McCluskey, E. J. (1965). *Introduction to the theory of switching circuits*. Princeton, NJ: Princeton University Press.

Quine, W. V. (1952). The problem of simplifying truth functions. *American Mathematical Monthly*, *59*, 521–531.

R Development Core Team. (2014). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing.

Ragin, C. C. (1987). *The comparative method: Moving beyond qualitative and quantitative strategies*. Berkeley: University of California Press.

Ragin, C. C. (2000). *Fuzzy-set social science*. Chicago, IL: University of Chicago Press.

Ragin, C. C. (2006). Set relations in social research: Evaluating their consistency and coverage. *Political Analysis*, *14*, 291–310.

Ragin, C. C. (2008). *Redesigning social inquiry: Fuzzy sets and beyond*. Chicago, IL: University of Chicago Press.

Ragin, C. C., & Davey, S. (2012). *fs/QCA: fuzzy-set/Qualitative Comparative Analysis* (Version 2.5). Irvine: Department of Sociology, University of California.

Ragin, C. C., Mayer, S. E., & Drass, K. A. (1984). Assessing discrimination: A Boolean approach. *American Sociological Review*, *49*, 221–234.

Schlager, E., & Heikkila, T. (2009). Resolving water conflicts: A comparative analysis of interstate river compacts. *Policy Studies Journal*, *37*, 367–392.

Thiem, A. (2013). Clearly crisp, and not fuzzy: A reassessment of the (putative) pitfalls of multi-value QCA. *Field Methods*, *25*, 197–207.

Thiem, A. (2014). Unifying configurational comparative methods: Generalized-set Qualitative Comparative Analysis. *Sociological Methods & Research*, *43*, 331–337.

Thiem, A. (2015). Parameters of fit and intermediate solutions in multi-value Qualitative Comparative Analysis. *Quality & Quantity*, *49*, 657–674.

Thiem, A., & Duşa, A. (2013a). Boolean minimization in social science research: A review of current software for Qualitative Comparative Analysis (QCA). *Social Science Computer Review*, *31*, 505–521.

Thiem, A., & Duşa, A. (2013b). QCA: A package for Qualitative Comparative Analysis. *The R Journal*, *5*, 87–97.

Thiem, A., & Duşa, A. (2013c). *Qualitative Comparative Analysis with R: A user's guide*. New York, NY: Springer.

Thomas, R. (1973). Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, *42*, 563–585.

Vis, B. (2009). Governments and unpopular social policy reform: Biting the bullet or steering clear? *European Journal of Political Research*, *48*, 31–57.

Weyland, K. (1998). The political fate of market reform in Latin America, Africa, and Eastern Europe. *International Studies Quarterly*, *42*, 645–673.