Jurnal Ilmiah Matematika dan Pendidikan Matematika (JMP)
Vol. 13 No. 2, Desember 2021, hal. 27-44
ISSN (Cetak) : 2085-1456; ISSN (*Online*) : 2550-0422

IMPLEMENTASI PENYEDERHANAAN FUNGSI BOOLE DENGAN METODE *QUINE McCLUSKEY*

Alfatah Hidayat

Jurusan Matematika, Fakultas MIPA, Universitas Jenderal Soedirman
alfatah215@gmail.com

Siti Rahmah Nurshiami

Jurusan Matematika, Fakultas MIPA, Universitas Jenderal Soedirman
siti.nurshiami@unsoed.ac.id

Mashuri

Jurusan Matematika, Fakultas MIPA, Universitas Jenderal Soedirman
mashuri@unsoed.ac.id

ABSTRACT. *Quine McCluskey method is one method that can be used to simplify the Boolean function. The Quine McCluskey method has several advantages including having simpler, more systematic steps than other methods and it is easier to simplify the Boolean function with a large number of variables. This study discusses the design of a Boolean function simplification program for the Quine McCluskey method using Visual Basic 6.0. The resulting program can simplify the Boolean function with many variables less than equal to 26 variables and able to simplify the Boolean function in the form of Sum of Product (SOP), Product of Sum (POS), and don't care.*

Keywords: *Boolean function, Boolean function simplification, Quine McCluskey, Visual Basic* 6.0, *Sum of Product, Product of Sum, don't care.*

ABSTRAK. Metode *Quine McCluskey* merupakan salah satu metode yang dapat digunakan untuk menyederhanakan fungsi Boole. Metode *Quine McCluskey* memiliki beberapa keunggulan diantaranya memiliki langkah-langkah penyederhanaan yang lebih sistematis dibanding metode lain dan lebih mudah untuk menyederhanakan fungsi Boole dengan jumlah variabel besar. Penelitian ini membahas perancangan program penyederhanaan fungsi Boole metode *Quine McCluskey* menggunakan *Visual Basic* 6.0. Program yang dihasilkan mampu menyederhanakan fungsi Boole dengan jumlah variabel kurang dari sama dengan 26 variabel dan mampu menyederhanakan fungsi Boole dalam bentuk *Sum of Product* (SOP), *Product of Sum* (POS), maupun dengan kondisi *don't care.*

Kata kunci: fungsi Boole, penyederhanaan fungsi Boole, *Quine McCluskey, Visual Basic* 6.0, *Sum of Product, Product of Sum, don't care*.

1. PENDAHULUAN

Aljabar Boole diperkenalkan pertama kali oleh seorang matematikawan yang berasal dari Inggris, George Boole pada tahun 1854. Saat ini Aljabar Boole digunakan secara luas dalam perancangan rangkaian saklar, rangkaian digital, dan rangkaian *integrated circuit* (IC) komputer. Rangkaian-rangkaian tersebut dapat direpresentasikan ke dalam fungsi Boole. Fungsi Boole adalah pemetaan $f: B^n \rightarrow B$ dengan B^n himpunan yang beranggotakan pasangan terurut ganda-*n* (Munir, 2010). Elemen-elemen dari B^n berupa pasangan-pasangan terurut ganda-*n* yang merupakan semua kombinasi dari elemen B = {0,1}. Suku-suku dalam fungsi Boole *n* variabel $x_1, x_2, \ldots, x_n$ dikatakan *minterm* jika muncul dalam bentuk perkalian dari variabel x_i atau x_i' dengan i = 1, 2, ..., *n*. *Minterm* dilambangkan dengan *m* berindeks. Indeks menyatakan basis desimal dari *string biner* yang mempresentasikan suku tersebut. Suku-suku dalam fungsi Boole *n* variabel $x_1, x_2, \ldots, x_n$ dikatakan *maxterm* jika muncul dalam bentuk penjumlahan dari variabel x_i atau x_i' dengan i = 1, 2, ..., *n*. *Maxterm* dilambangkan dengan *M* berindeks (Rosen, 2012). Pada *minterm*, setiap variabel yang bernilai 0 dinyatakan dalam bentuk komplemen, sedangkan variabel yang bernilai 1 dinyatakan tanpa komplemen. Sebaliknya, pada *maxterm*, setiap variabel yang bernilai 1 dinyatakan dalam bentuk komplemen, sedangkan variabel yang bernilai 0 dinyatakan tanpa komplemen. Sebagai contoh, $A'BC'$ merupakan *minterm* ke-2 atau m_2, sedangkan $A + B + C'$ merupakan *maxterm* ke-1 atau M_1.

Fungsi Boole dapat dinyatakan dalam bentuk penjumlahan dari satu *minterm* atau lebih (*Sum of Product/SOP*) ataupun dapat dinyatakan sebagai perkalian dari satu *maxterm* atau lebih (*Product of Sum/POS*). Fungsi Boole $f(A,B,C) = A'BC' + ABC' + ABC$ merupakan fungsi Boole dengan 3 variabel dalam bentuk SOP, sedangkan $f(A,B,C) = (A + B + C)(A + B + C')(A + B' + C')(A' + B + C)(A' + B + C)$ merupakan fungsi Boole dengan 3 variabel dalam bentuk POS.

Fungsi Boole dapat direpresentasikan ke dalam suatu rangkaian digital. Fungsi Boole dengan variabel yang sangat banyak dan memuat operasi-operasi yang berlebihan akan rumit dalam mengimplementasikan ke dalam suatu

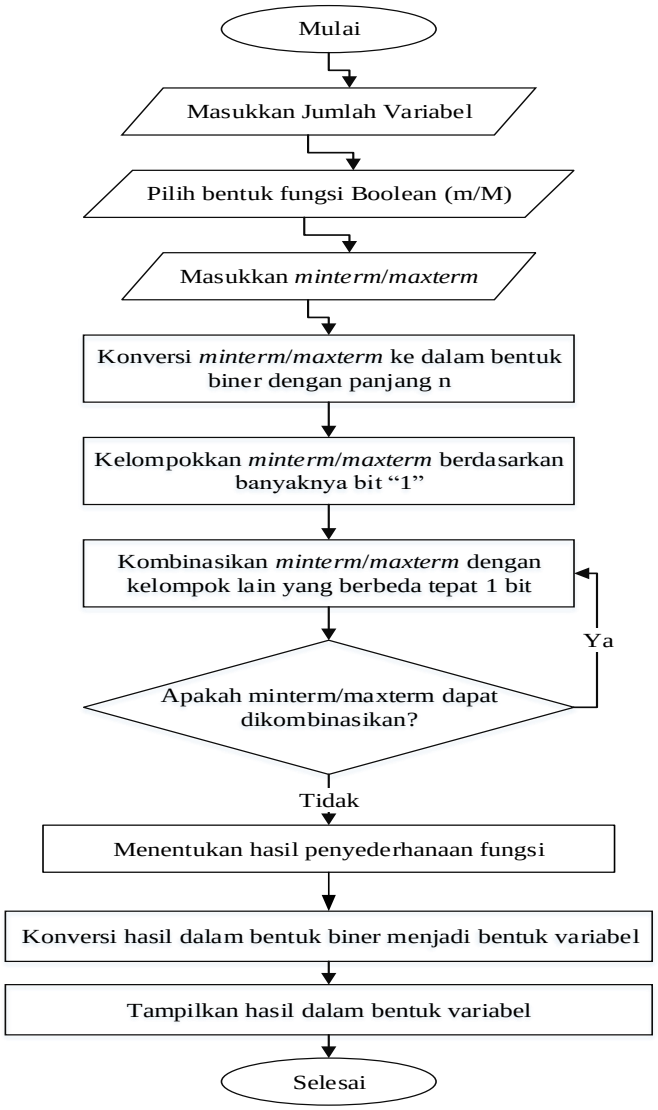
rangkaian digital, sehingga perlu dilakukan penyederhanaan fungsi Boole. Penyederhanaan fungsi Boole atau minimisasi fungsi merupakan pencarian bentuk fungsi lain yang ekuivalen tetapi dengan jumlah operasi yang lebih sedikit, sederhana dan efisien, sehingga lebih mudah untuk membuat suatu rangkaian digital dan biaya yang dikeluarkan lebih sedikit (Nugroho, 2021). Penyederhanaan fungsi Boole secara aljabar merupakan penyederhanaan fungsi Boole dengan menggunakan sifat-sifat aljabar Boole, seperti komutatif, distributif, assosiatif, identitas, idempoten, dan penyerapan. Sebagai contoh, fungsi Boole dengan 4 buah variabel, $f(A,B,C,D) = ABD + ABD' + A'C + A'BC + ABC$ dapat disederhanakan menjadi fungsi Boole $f(A,B,C,D) = AB + A'C$. Fungsi Boole dapat disederhanakan dengan beberapa cara diantaranya dengan penyederhanaan secara aljabar, *Karnaugh Map (K-Map)*, dan metode tabulasi.

Metode tabulasi atau lebih dikenal dengan metode *Quine McCluskey* dikembangkan oleh W.V. Quine dan E.J. McCluskey pada tahun 1950. Metode ini merupakan metode terbaik dan efisien dalam penyederhanaan fungsi Boole dengan 6 variabel atau lebih apabila dibandingkan dengan metode *Karnaugh Map* (Widianto, 2014). Hal ini dikarenakan metode *Karnaugh Map* tidak dapat untuk menggambarkan realisasi rangkaian secara pasti sehingga tidak dapat menjamin untuk mendapatkan hasil yang terbaik. Selain itu, pada metode *Karnaugh Map* ada hal-hal tertentu yang bisa dikatakan sangat subyektif, yaitu hal-hal yang sangat dipengaruhi oleh intuisi pribadi. Hal ini dapat dilihat dalam langkah-langkah pembuatan suatu program komputer, dimana antara satu orang dengan yang lainnya tidak sama; demikian juga dalam Langkah-langkah penyederhanaan suatu rangkaian dengan metode *Karnaugh Map* (Polosoro, 2009). Metode *Quine McCluskey* mempunyai dua kelebihan dibandingkan dengan metode *Karnaugh Map*. Pertama, metode *Quine McCluskey* merupakan metode yang sistematis untuk menyederhanakan fungsi Boole. Kedua, metode *Quine McCluskey* lebih mudah untuk menyederhanakan fungsi Boole dengan jumlah variabel yang cukup banyak. Selain itu, metode *Quine McCluskey* efektif dalam menyederhanakan suatu rangkaian digital dari 8 *IC* yang semula dibutuhkan menjadi 3 *IC* (Purba, 2021).

Penelitian sebelumnya mengenai program penyederhanaan fungsi Boole dengan metode *Quine McCluskey* diantaranya menggunakan pemrograman Java dan fungsi yang disederhanakan hanya memiliki 4 variabel (Tomaszewski, *et al.*, 2003), menggunakan pemrograman C++ dan fungsi Boole yang disederhanakan memiliki 3 – 5 variabel (Joshi, 2020). Kemudian penyederhanakan fungsi Boole dengan 10 variabel (Ismayanto & Sukmaindrayana, 2018), (Saputra, 2015), penyederhanaan fungsi Boole dengan 2 variabel sampai 26 variabel (Wamiliana, *et al.,* 2013). Namun keempat penelitian tersebut hanya untuk menyederhanakan fungsi Boole dalam bentuk kanonik *Sum of Product* (SOP). Selanjutnya penyederhanaan fungsi Boole dengan bentuk kanonik SOP dan *Product of Sum* (POS) dilakukan oleh (Sipayung, 2020) menggunakan pemrograman Visual Basic dan (Nugroho, 2021) menggunakan modifikasi metode *Quine McCluskey* dan metode *Petrick*. Pada penelitian ini akan membahas penyederhanakan fungsi Boole dengan 2 variabel sampai 26 variabel menggunakan metode *Quine McCluskey* dan metode *Petrick*. Fungsi Boole yang digunakan dalam bentuk kanonik SOP, POS, dan kondisi *don't care*. Implementasi penyederhanaan fungsi Boole yang dilakukan menggunakan *Visual Basic* 6.0.

2. METODE PENELITIAN

Metode yang digunakan untuk menyelesaikan permasalahan ini adalah studi literatur. Langkah awal yang dilakukan menganalisa tahapan-tahapan dalam penyederhanaan fungsi Boole menggunakan metode Quine McCluskey. Kemudian merancang dan mengembangkan aplikasi berdasarkan hasil analisa. Langkah terakhir melakukan uji coba aplikasi. Berikut merupakan algoritma program penyederhanaan fungsi Boole menggunakan metode *Quine McCluskey* yang disajikan dalam bentuk *flowchart* seperti terlihat pada Gambar 1.



Gambar 1. *Flowchart* penyederhanaan fungsi Boole metode Quine McCluskey

3. HASIL DAN PEMBAHASAN

3.1 Metode Quine McCluskey

Selain secara aljabar, penyederhanaan fungsi Boole dapat dilakukan dengan menggunakan metode *Quine McCluskey*. Metode *Quine McCluskey* mengubah sebuah fungsi Boole menjadi sebuah implikan prima (*prime implicant*), dimana sebanyak mungkin peubah dieliminasi (dihilangkan) secara maksimal, hingga didapat fungsi Boole yang paling sederhana. Ini dapat dilakukan dengan

melakukan perulangan penggunaan hukum komplemen, a + a’ = 1 (Nasution, 2016).

Adapun langkah-langkah dalam penyederhanaan fungsi Boole bentuk kanonik SOP/POS menggunakan metode *Quine McCluskey* adalah sebagai berikut

1. Nyatakan tiap *minterm/maxterm* dengan *don’t care* atau tidak, dalam *n* variabel menjadi string *n*- bit.
2. Kelompokkan tiap *minterm/maxterm* dan *don’t care* berdasarkan banyaknya angka "1” yang dimiliki.
3. Kombinasikan *minterm/maxterm* dan *don’t care* dalam *n* variabel dengan kelompok lain yang jumlah angka “1’nya berbeda tepat satu, sehingga diperoleh implikan prima (*prime implicant*) yang terdiri dari *n* – 1 variabel. *Minterm/maxterm* yang dikombinasikan diberi tanda “√”
4. Kombinasikan *minterm/maxterm* dan *don’t care* dalam *n* – 1 variabel dengan kelompok lain yang jumlah angka “1’nya berbeda tepat satu, sehingga diperoleh implikan prima (*prime implicant*) yang terdiri dari *n* – 2 variabel.
5. Teruskan langkah 4 sampai diperoleh implikan prima yang paling sederhana.
6. Buat tabel yang memperlihatkan *minterm/maxterm* dari fungsi Boole yang dicakup oleh implikan prima yang tidak diberi tanda “√”. Setiap *maxterm* yang dicakup oleh masing-masing implikan prima diberi tanda “×”. Setiap *minterm/maxterm* harus mencakup paling sedikit satu buah implikan prima.
7. Pilih implikan prima yang memiliki jumlah literal paling sedikit namun mencakup sebanyak mungkin *minterm/maxterm* dari fungsi Boole awal dengan cara menandai kolom-kolom yang mempunyai satu tanda “×”dengan tanda “∗”, lalu beri tanda “√ ” di sebelah kiri implikan prima yang berasosiasi dengan tanda “∗”.
8. Untuk setiap yang ditandai dengan “√ ”, beri tanda *minterm/maxterm* yang dicakup oleh implikan prima tersebut dengan tanda “√” di baris bawah.
9. Periksa apakah masih ada *minterm/maxterm* yang belum tercakup oleh implikan prima terpilih. Jika ada, pilih dari tersisa yang mencakup sebanyak mungkin *minterm/maxterm*. Beri tanda “√ ” yang dipilih beserta

minterm/maxterm yang dicakupinya. Lakukan sampai semua *minterm/maxterm* tercakup oleh implikan prima terpilih.

10. Apabila setiap kolom memiliki lebih dari satu tanda "×" dan tidak ada baris yang mendominasi, maka pilih salah satu (utamakan pilih implikan prima yang lebih banyak mencakup *minterm/maxterm* yang belum tercakup oleh implikan prima yang sudah terpilih sebelumnya).
11. Konversi *string biner* kedalam bentuk kanonik SOP, "0" dinyatakan dengan variabel komplemen, "1" dinyatakan dengan variabel bukan komplemen, sedangkan "-" dinyatakan dengan tidak ada variabel.

3.2 Kondisi *Don't Care*

Kondisi *don't care* merupakan suatu kondisi dimana nilai yang dihasilkan oleh suatu fungsi Boole tidak akan mempengaruhi hasil output. *Don't care* bisa diasumsikan dengan 1 atau 0 yang biasanya dinotasikan dengan *X* atau *d.* Dalam penyedehanaan fungsi Boole metode *Quine McCluskey*, kondisi *don't care* dilibatkan dalam langkah pertama sampai langkah ke-5, namun tidak dilibatkan pada langkah ke-6 sampai terakhir baik dalam bentuk kanonik SOP maupun POS.

Contoh 1

Misal diberikan fungsi Boole $f(A,B,C,D,E) = \prod M(3,5,10,11,12,14)$. Fungsi Boole tersebut akan disederhanakan menggunakan metode *Quine McCluskey.*

1. Menkonversi *maxterm* ke dalam *string biner* dengan panjang *n* dan mengelompokkan *maxterm* berdasarkan banyaknya angka 1 pada *string biner*, seperti terlihat pada Tabel 1.

Tabel 1. Pengelompokan maxterm

Banyaknya 1	***Term***	A	B	C	D	E
2	3	0	0	0	1	1
	5	0	0	1	0	1
	10	0	1	0	1	0
	12	0	1	1	0	0
3	11	0	1	0	1	1
	14	0	1	1	0	1

2. Mengkombinasikan kelompok *maxterm* yang satu dengan yang lainnya yang memiliki perbedaan tepat satu *string biner*.

Term	***A***	***B***	***C***	***D***	***E***
3	0	0	0	1	1 √
5	0	0	1	0	1
10	0	1	0	1	0 √
12	0	1	1	0	0 √
11	0	1	0	1	1 √
14	0	1	1	1	0 √

⇒

Term	***A***	***B***	***C***	***D***	***E***
3,11	0	-	0	1	1
10,11	0	1	0	1	-
10,14	0	1	-	1	0
12,14	0	1	1	-	0

3. Langkah di atas sudah diperoleh bentuk prima paling sederhana, maka langkah berikutnya membuat tabel implikan prima.

Tabel 2. Implikan prima

Implikan prima		***Maxterm***					
		3*	**5***	**10**	**11**	**12***	**14**
√	5		×				
√	3,11	×			×		
	10,11			×	×		
	10,14			×			×
√	12,14					×	×
		√	√		√	√	√

Bentuk prima terpilih pada langkah 7 belum mencakum *maxterm* 10, sehingga perlu dipilih lagi bentuk prima yang mencakup *maxterm* 10. Dalam kasus ini terdapat dua bentuk prima yang mencakup *maxterm* 10 dan memiliki jumlah literal sama sehingga dapat dipilih salah satunya. Implikan prima yang terpilih adalah

5 yang bersesuaian dengan $A + B + C' + D + E'$

3,11 yang bersesuaian dengan $A + C + D' + E'$

12,14 yang bersesuaian dengan $A + B' + C' + E$

Jadi bentuk sederhana dari fungsi Boole

$$f(A,B,C,D,E) = \prod M(3,5,10,11,12,14)$$

dalam bentuk POS adalah

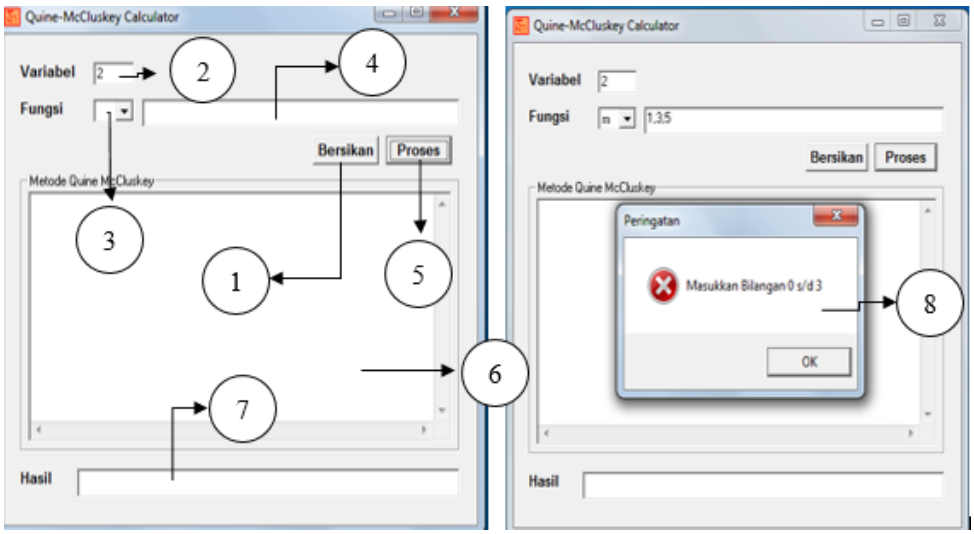
$$f(A,B,C,D,E) = (A + B + C' + D + E')(A + C + D' + E')(A + B' + C' + E)(A + B' + C + D')$$

atau

$$f(A,B,C,D,E) = (A + B + C' + D + E')(A + C + D' + E')(A + B' + C' + E)(A + B' + D' + E).$$

3.3 Implementasi

Algoritma atau langkah-langkah penyederhanaan fungsi Boole menggunakan metode *Quine McCluskey* dapat diimplemetasikan kedalam pemrograman *Visuial Basic* sehingga dihasilkan suatu program yang memudahkan untuk menyederhanakan fungsi Boole. Berikut adalah beberapa *interface* program penyederhanaan fungsi Boole menggunakan metode *Quine McCluskey.*



Gambar 2. *Interface* program

Penjelasan fungsi/kegunaan dari masing-masing komponen pada *Interface* sebagai berikut:

1. Tombol Bersihkan, digunakan untuk kembali ke *interface* awal.
2. *TextBox variabel*, digunakan untuk memasukkan jumlah variabel pada fungsi Boole.
3. *ComboBox fungsi*, digunakan untuk memilih jenis fungsi Boole SOP atau POS. Apabila fungsi Boole dalam bentuk SOP maka memilih *m* atau *minterm*, sedangkan fungsi Boole dalam bentuk POS memilih *M* atau *maxterm*.
4. *TextBox term*, digunakan untuk memasukkan *minterm/maxterm* dari fungsi Boole.

5. Tombol Proses, digunakan untuk menyederhanakan fungsi Boole yang telah dimasukkan.
6. *TextBox step*, digunakan untuk menampilkan langkah-langkah penyederhanaan fungsi Boole menggunakan metode *Quine McCluskey.*
7. *TextBox hasil*, digunakan untuk menampilkan hasil penyederhanaan fungsi Boole dalam bentuk POS maupun SOP.
8. Peringatan apabila fungsi yang dimasukkan tidak sesuai.

Adapun cara menggunakan program *Quine McCluskey Calculator* untuk menyederhanakan fungsi Boole adalah sebagai berikut

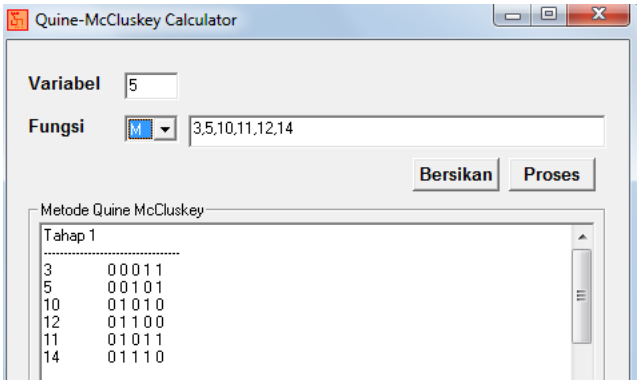
1. Masukkan jumlah variabel pada *TextBox variabel.*
2. Tentukan bentuk kanonik fungsi Boole dengan cara memilih m/M pada *ComboBox fungsi*, "m" untuk bentuk kanonik SOP dan "M" untuk bentuk kanonik POS.
3. Masukkan *minterm/maxterm* pada *TextBox term* dengan cara menuliskan indeks dari *minterm/maxterm*, memberi tanda koma (,) untuk memisahkan *term* yang satu dengan yang lainnya dan tanda *plus* (+) untuk memisahkan kondisi *don't care* dengan yang bukan *don't care.* Kondisi *don't care* ditulis setelah tanda *plus* (+).
4. Klik tombol Proses untuk menampilkan langkah-langkah penyederhanaan fungsi Boole menggunakan metode *Quine McCluskey* dan hasil penyederhanaan.
5. Klik tombol Bersihkan untuk mengembalikan ke *interface* awal.

3.4 Hasil Pengujian

Pengujian dilakukan dengan menyederhanakan fungi Boole dengan dua contoh berbeda yaitu bentuk POS, dan fungsi Boole dengan kondisi *don't care*.

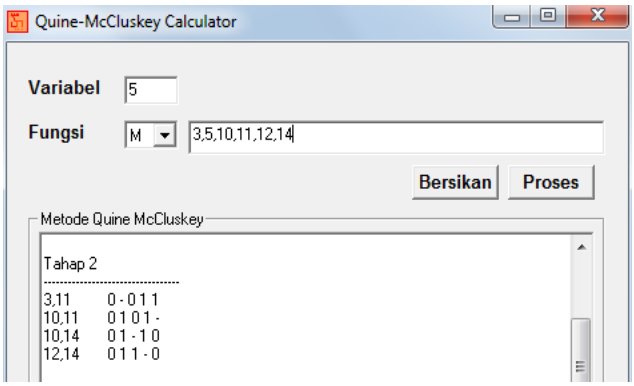
Misal diberikan fungsi Boole $f(A,B,C,D,E) = \prod M\ (3,5,10,11,12,14)$. Berikut hasil penyederhanaan fungsi Boole menggunakan metode *Quine McCluskey.*

Langkah 1, mengkonversi *maxterm* menjadi *string biner* dan mengurutkannya bedasarkan banyaknya angka 1 pada *string biner.*



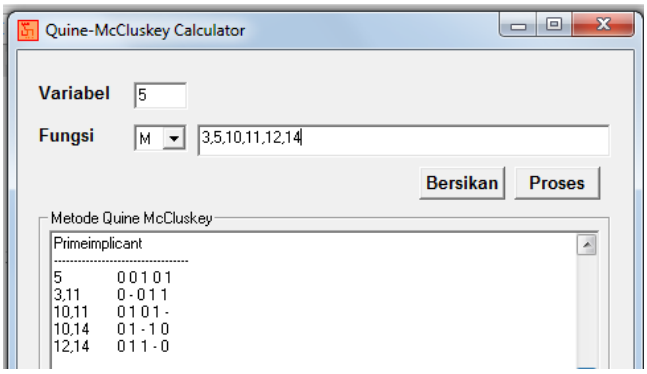
Gambar 3. Tahap 1 penyederhanaan fungsi Boole bentuk kanonik POS

Langkah 2, mengkombinasikan *maxterm* satu dengan yang lainnya yang memiliki perbedaan tepat 1 digit sehingga diperoleh bentuk prima dengan panjang *string* *n*-1.



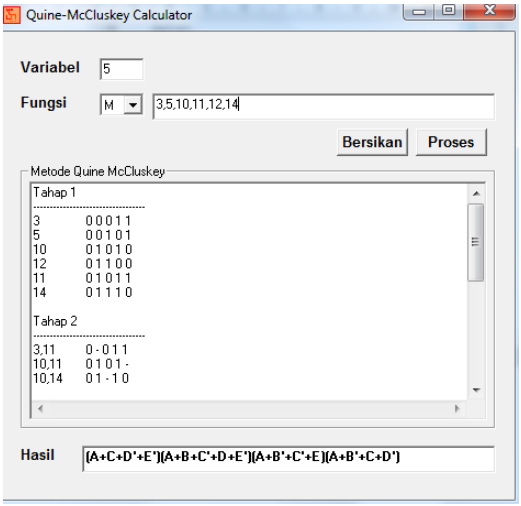
Gambar 4. Tahap 2 penyederhanaan fungsi Boole bentuk kanonik POS

Langkah 3, menampilkan bentuk implikan prima yang tidak memiliki pasangan kombinasi.



Gambar 5. Bentuk implikan prima yang diperoleh

Langkah 4, hasil penyederhanaan fungsi Boole



Gambar 6. Hasil penyederhanaan fungsi Boole bentuk kanonik POS

Hasil penyederhanaan fungsi Boole $f(A,B,C,D,E) = \prod M$ (3,5,10,11,12,14) dalam bentuk POS adalah

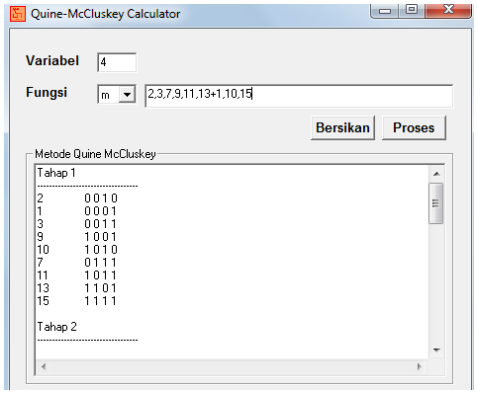
$$f(A,B,C,D,E) = (A + C + D' + E')(A + B + C' + D + E')(A + B' + C' + E)\,(A + B' + C + D').$$

Selanjutnya, misal diberikan fungsi Boole

$$f(A,B,C,D) = \sum m(2,3,7,9,11,13) + \sum d(1,10,15)\,.$$

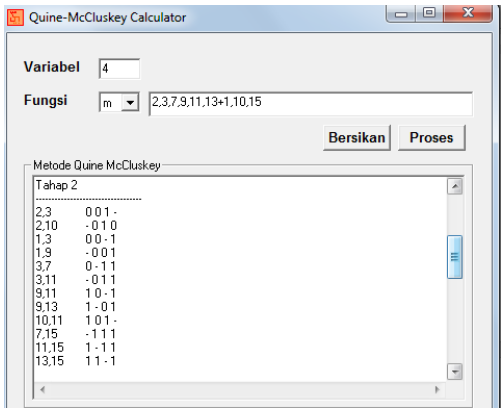
Berikut hasil penyederhanaan fungsi Boole dengan menggunakan metode *Quine McCluskey.*

Langkah 1, mengkonversi *minterm* menjadi *string biner* dan mengurutkannya bedasarkan banyaknya angka 1 pada *string biner.*



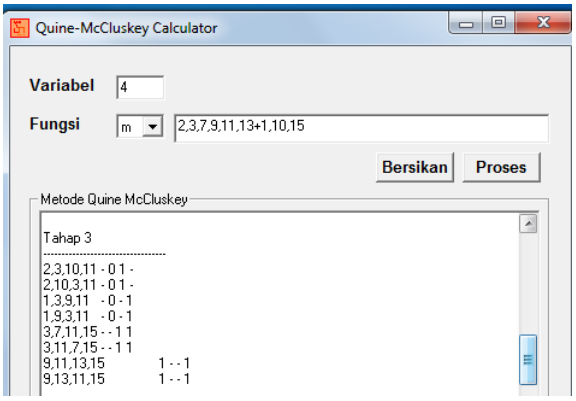
Gambar 7. Tahap 1 penyederhanaan fungsi Boole dengan kondisi *don't care*

Langkah 2, mengkombinasikan minterm satu dengan yang lainnya yang memiliki perbedaan tepat 1 digit sehingga diperoleh bentuk implikan prima dengan panjang *string n*-1.



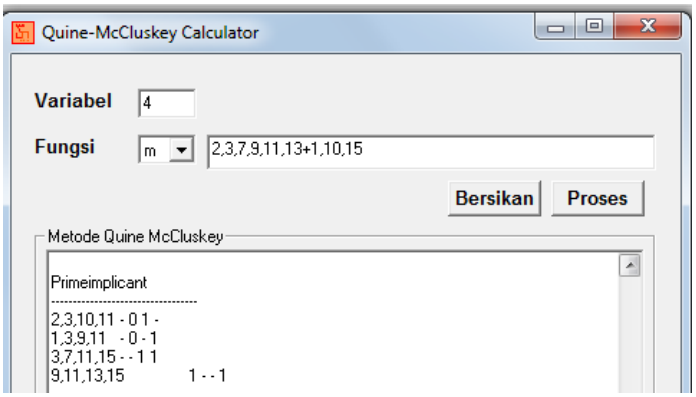
Gambar 8. Tahap 2 penyederhanaan fungsi Boole dengan kondisi *don't care*

Langkah 3, mengkombinasikan bentuk prima dengan panjang *string n*-1 sehingga diperoleh bentuk implikan prima dengan panjang *sting n*-2.



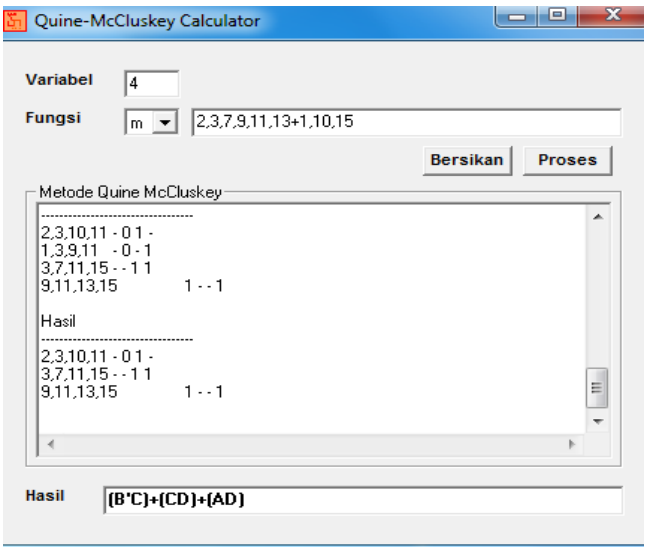
Gambar 9. Tahap 3 penyederhanaan fungsi Boole dengan kondisi *don't care*

Langkah 4, menampilkan bentuk implikan prima yang tidak memiliki pasangan kombinasi.



Gambar 10. Bentuk implikan prima hasil penyederhanaan fungsi Boole dengan kondisi *don't care*

Langkah 5, hasil penyederhanaan fungsi Boole



Gambar 11. Hasil penyederhanaan fungsi Boole bentuk SOP dengan *don't care*

Hasil penyederhanaan dari fungsi Boole

$$f(A,B,C,D) = \sum m(2,3,7,9,11,13) + \sum d(1,10,15)$$

dalam bentuk SOP dengan kondisi *don't care* menggunakan pemrograman *Visual Basic* adalah $f(A,B,C,D) = B'C + CD + AD$.

3 KESIMPULAN DAN SARAN

Berdasarkan hasil dan pembahasan mengenai penyederhanaan fungsi Boole menggunakan metode *Quine McCluskey* didapatkan suatu program aplikasi yang dapat menyederhanakan fungsi Boole. Aplikasi yang dihasilkan berfungsi dengan baik dalam menyederhanakan fungsi Boole dalam bentuk kanonik SOP, POS, maupun dengan kondisi *don't care* dengan n variabel, $2 \leq n \leq 26$. Pada penelitian sebelumnya aplikasi yang diperoleh hanya untuk fungsi Boole dalam bentuk SOP dan POStanpa memperhatikan kondisi *don't care*. Dalam penggunaan program ini, pengguna hanya perlu menginputkan jumlah variabel. Kemudian pilih bentuk kanonik fungsi Boole, "m" untuk bentuk kanonik SOP sedangkan "M" untuk bentuk kanonik POS. Setelah itu masukkan minterm/maxterm dan beri tanda plus (+) untuk memisahkan dengan kondisi *don't care.* Program akan menampilkan

tahapan-tahapan penyederhanaan fungsi Boole dengan metode *Quine McCluskey* sampai diperoleh hasil penyederhanaan.

DAFTAR PUSTAKA

Ismayanto, & Sukmaindrayana, A., *Penyederhanaan Fungsi Boolean dengan Metode Quine McCluskey*, Jumantaka, **1**(1) (2018), 231 - 240.

Joshi, M. S., Formulation of C++ program for Quine–McCluskey Method of Boolean Function Minimization, *Machine Learning, Advanced in Computing, Renewable Energy and Communications*, India, 2020, 341-346.

Munir, R., *Matematika Diskrit*, Edisi Keempat, Informatika Bandung, Bandung, 2010.

Nasution, A., *Implementasi Fungsi Boolean Dengan Metode Quine-McCluskey*. Kultura, **17**(1) (2016), 5786 - 5792.

Nugroho, E. D., *Development of Applications for Simplification of Boolean Functions using Quine-McCluskey Method*, Telematika : Jurnal Informatika dan Teknologi Informasi, **18**(1) (2021), 27-36.

Polosoro, E., *Sistem Digital*, Graha Ilmu, Yogyakarta, 2009.

Purba, D. P., *Efisiensi Komponen Rangkaian Logika dengan Menggunakan Metode Penyederhanaan Quine-McCluskey*, Citra Sains Teknologi, 1(1) (2021), 43-49.

Rosen, K. H., *Discrete Mathematics and Its Applications*, 7th Edition, Mc. Graw Hill, New York City, 2012.

Saputra, H., *Penerapan Metode Quine-Mc Cluskey untuk Menyederhanakan Fungsi Boolean*, Jurnal Teknologi dan Sistem Informasi, **2**(1) (2015), 31-41.

Sipayung, L. Y., *Perancangan Perangkat Lunak Pembelajaran Untuk Penyederhanaan Fungsi Boolean dengan Metode Quine-McCluskey*. Jurnal Sains dan Teknologi ISTP, **13**(1) (2020), 50 - 57.

Tomaszewski, S. P., Ilgas U., C., George E, A., *www-based Boolean Function Minimization*, Applied Mathematics and Computer Sciences, **13**(4) (2003), 577 - 583.

Wamiliana, Ossy D, E., Shara S, Z., *Pengembangan Aplikasi Penyederhanaan Aljabar Boolean dengan Menggunakan Metode Quine Mccluskey dalam Bentuk Sum-Of-Product*, Jurnal Komputasi, **1**(2) (2013), 50-58.

Widianto, E. D., *Sistem Digital : Analisis, Desain dan Implementasi*, Graha Ilmu, Yogyakarta, 2014.