Analisis Solvabilitas Berbasis Aljabar Boolean pada Varian "The 24 Game" Menggunakan Operasi Bitwise 5-bit

Pasaribu Fritz T.A.M. - 13525105
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: fritzkuliah01@gmail.com, 13525105@std.stei.itb.ac.id

***Abstrak*—Permainan kartu "The 24 Game: merupakan permainan matematika populer yang menantang pemain untuk menyusun ekspresi aritmetika dari empat kartu acak dan bertujuan untuk mencapai nilai target 24. Penelitian ini mengkaji modifikasi varian baru dari permainan tersebut dengan mengganti seluruh operasi aritmetika menggunakan operasi bitwise (NOT, AND, dan OR) berbasis representasi bilangan biner 5-bit. Tujuan penelitian ini adalah untuk mengukur tingkat solvabilitas dari varian permainan ini serta merumuskan karakteristik berdasarkan aljabar Boolean dari set kartu yang berhasil maupun gagal dipecahkan. Melalui pendekatan kombinatorika dengan prinsip *Stars and Bars*, ditemukan 1820 kombinasi set kartu unik dengan tingkat solvabilitas 61,26% set kartu yang solvable berdasarkan pengujian menggunakan algoritma *Depth-First Search*. Analisis karakteristik aljabar Boolean membantu dalam pencarian solusi suatu set kartu dalam penelitian ini melalui kewajiban operasi NOT, pencarian angka "kunci", dan identifikasi set kartu yang unsolvable.**

Keywords—Aljabar Boolean; The 24 Game; Operasi Bitwise; Peta Karnaugh; Solvabilitas

I. PENDAHULUAN

Permainan kartu masih menjadi salah satu jenis permainan tanpa gawai yang populer di kalangan remaja, salah satunya adalah permainan kartu *The 24 Game*. Permainan ini menjadi salah satu bentuk permainan yang diminati karena lebih ramah untuk pemula dan tidak banyak mengandalkan strategi maupun keberuntungan. Permainan ini unik dibandingkan permainan kartu lainnya karena lebih fokus pada kemampuan untuk menemukan solusi matematis dengan cepat, sehingga mampu meningkatkan kemampuan dan kecepatan *problem solving* secara matematis.

Aturan dasar dari permainan ini adalah untuk membuka tepat 4 kartu dari tumpukan kartu, kemudian mencari kombinasi operasi aritmetika $(+,-,\times,\div)$ untuk menghasilkan angka 24 dari keempat nilai angka dari kartu yang terbuka tepat satu kali. Meskipun *The 24 Game* ini sudah memiliki aturan baku, konsep dari permainan ini sendiri tetap merupakan *public domain* sehingga bebas untuk didesain dan dimodifikasi aturannya oleh masing-masing orang untuk menghasilkan variasi permainan baru[7].

Dalam pendekatan informatika, salah satu modifikasi yang dapat dilakukan adalah untuk membentuk varian *The 24 Game* yang bukan lagi menggunakan operasi aritmetika, melainkan operasi bitwise (*not*, *and*, , dan or) tanpa adanya sistem *carry-over*. Varian ini dapat memberikan tantangan baru (khususnya bagi orang-orang di bidang informatika) untuk bermain sambil meningkatkan kemampuan mencari solusi dalam batasan operasi bitwise.

Dengan banyaknya kemungkinan kombinasi kartu dan operasi yang dapat digunakan sebagai solusi, permainan *The 24 Game* ini sering menjadi topik pembahasan Matematika Diskrit. Pada penelitian ini, pendekatan kombinatorika dan aljabar boolean digunakan untuk menganalisis penyelesaian permainan varian bitwise The 24 Game. Analisis ini bertujuan untuk mengukur persentase kombinasi set kartu yang memiliki solusi, serta merancang aturan dalam ekspresi aljabar boolean untuk memperoleh karakteristik bit dari set kartu yang berhasil maupun gagal dipecahkan.

II. LANDASAN TEORI

A. The 24 Game

The 24 Game merupakan salah satu permainan matematika yang sering diimplementasikan dalam permainan kartu. Permainan ini tidak hanya sebatas menjadi hiburan, tetapi juga tantangan untuk melatih kemampuan penalaran matematika dan kecepatan berpikir dalam penyelesaian masalah. Dalam permainan ini simbol dari setiap kartu tidak menjadi penting, tetapi hanya nilai angka nya saja yang penting, di mana kartu As bernilai 1, Jack bernilai 11, Queen bernilai 12, King bernilai 13, dan kartu angka bernilai sesuai angka nya masing-masing [1].

Tujuan utama pada permainan ini adalah untuk menghasilkan angka 24 dari 4 kartu yang terbuka pada setiap awal giliran dengan memanfaatkan operasi-operasi yang legal $(+,-,\times,\div)$. Pemain dapat menambahkan tanda kurung untuk mengatur urutan prioritas operasi sedemikian rupa untuk mencapai tujuan. Alur serta peraturan yang umum dari The 24 Game [1]:

1. Pada awal giliran, seorang pemain mengambil 4 kartu secara acak dari atas tumpukan kartu.

2. Keempat kartu tersebut disusun terbuka untuk bisa dilihat oleh semua permain.
3. Seluruh pemain perlu memikirkan cara menyusun suatu ekspresi matematika yang menghasilkan 24 dari keempat angka pada kartu dan operasi-operasi yang legal.
4. Setiap kartu harus digunakan tepat satu kali dalam penyelesaian.
5. Pemain yang sudah menemukan solusi harus menyebutkan namanya atau cara lainnya untuk memberi tanda bahwa ia telah menemukan solusinya.
6. Pemain terakhir yang menemukan solusi harus mengambil keempat kartu tersebut.
7. Jika tidak ada pemain yang dapat menemukan solusi dari set kartu tersebut, keempat kartu dimasukkan kembali ke tumpukan, tumpukan dikocok kembali, dan diambil 4 kartu baru.
8. Permainan berakhir jika jumlah kartu yang tersisa sudah kurang dari 4, atau set kartu yang tersisa di tumpukan sudah tidak bisa dicari solusinya.
9. Pemain dengan jumlah kartu pegangan paling sedikit pada akhir permainan akan menjadi pemenang.

Seiring perkembangannya, aturan dari The 24 Game mulai dimodifikasi dan menghasilkan varian permainan dengan aturan baru [7]. Salah satu varian permainan yang jarang dibahas, dan menjadi fokus utama pembahasan makalah ini, adalah varian permainan yang memanfaatkan operasi bitwise. Pada varian permainan ini, setiap kartu tetap memiliki nilai angka yang sama, namun representasinya diubah menjadi representasi bilangan biner 5-bit (agar bisa dimanipulasi menjadi 24). Tujuannya tetap sama, yaitu untuk mencapai nilai 24 (11000 dalam representasi biner), namun operasi yang digunakan bukan lagi operasi-operasi aritmetika, melainkan operasi bitwise seperti NOT, AND, dan OR. Dengan perubahan tersebut, varian permainan ini tentu memiliki solusi dan tingkat solvabilitas yang berbeda dari permainan The 24 Game yang umum.

B. *Kombinatorika*

Kombinatorika merupakan cabang matematika untuk menghitung jumlah penyusunan sekumpulan objek tanpa perlu menghitung semua kemungkinannya satu per satu [3]. Dalam ilmu kombinatorika, terdapat dua kaidah dasar menghitung:

1) Kaidah Penjumlahan

Kaidah penjumlahan mengindikasikan beberapa percobaan yang terjadi secara paralel. Jika percobaan 1 dapat dilakukan dengan x cara sementara percobaan 2 dapat dilakukan dengan y cara, terdapat x+y cara untuk melakukan percobaan 1 *atau* percobaan 2.

2) Kaidah Perkalian

Kaidah perkalian mengindikasikan beberapa percobaan yang terjadi secara seri. Jika percobaan 1 dapat dilakukan dengan x cara kemudian percobaan 2 dapat dilakukan dengan y cara, terdapat x*y cara untuk melakukan percobaan 1 *dan* percobaan 2.

Dalam penerapannya, kedua kaidah dasar menghitung tersebut dapat dikembangkan untuk menyelesaikan permasalahan yang lebih kompleks [1]. Konsep yang umum digunakan dalam kombinatorika adalah permutasi dan kombinasi, prinsip inklusi-eksklusi, dan prinsip *stars and bars*.

1) Prinsip Inklusi-Eksklusi

Prinsip inklusi-eksklusi digunakan untuk memastikan agar elemen irisan antara himpunan percobaan 1 dengan himpunan percobaan 2 hanya dihitung satu kali dalam gabungan kedua himpunan untuk menghindari perhitungan ganda dalam proses kombinatorika. Untuk dua himpunan P_1 dan P_2, penerapan prinsip inklusi-eksklusi dalam gabungan kedua himpunan [3]:

$$|P_1 \cup P_2| = |P_1| + |P_2| - |P_1 \cap P_2| \quad (1)$$

2) Permutasi

Permutasi merupakan jumlah cara penyusunan sekumpulan objek dengan urutan yang berbeda. Terdapat beberapa jenis penerapan permutasi [3]:

a) Permutasi penuh n objek, tanpa pengulangan

$$P(n) = n! \quad (2)$$

b) Permutasi sebagian r dari n objek, yaitu jumlah kemungkinan penyusunan r objek yang dipilih dari n objek dengan $r \leq n$ dengan urutan yang berbeda dan tanpa pengulangan

$$P(n,r) = \frac{n!}{(n-r)!} \quad (3)$$

c) Permutasi n objek dengan adanya objek yang berulang. n_1, n_2, ... , n_k merupakan jenis objek berbeda dengan frekuensi yang bisa berulang (≥ 1) dan totalnya jika dijumlahkan sama dengan n.

$$P(n; n_1, n_2, \ldots, n_k) = \frac{n!}{n_1! n_2! \ldots n_k!} \quad (4)$$

3) Kombinasi

Kombinasi merupakan jumlah cara penyusunan sekumpulan objek, tanpa memerhatikan urutan kemunculan objek yang dipilih atau disusun. Jumlah cara pemilihan r dari n objek menggunakan prinsip kombinasi adalah [3]

$$C(n,r) = \binom{n}{r} = \frac{n!}{r!(n-r)!} \quad (5)$$

4) Prinsip Stars and Bars

Stars and bars merupakan suatu metode untuk menghitung jumlah cara pendistribusian objek identik ke beberapa wadah yang berbeda. Metode ini memanfaatkan konsep kombinasi dengan elemen yang berulang. Untuk menghitung jumlah cara menempatkan n objek ke dalam k wadah dan setiap wadah boleh diisi lebih dari 1 objek, dapat digunakan rumus [3]:

$$Jumlah\ Cara\ = C(n+k-1, k-1) \quad (6)$$

C. *Aljabar Boolean*

Aljabar boolean yang ditemukan George Boole pada tahun 1854 merupakan representasi dari suatu sistem aljabar yang berhubungan dengan nilai logika, benar atau salah, yang direpresentasikan dalam bentuk bilangan biner 0 dan 1 [2]. Nilai

kebenaran tersebut dapat dimanipulasi melalui operasi dasar dalam aljabar boolean:

1) Operasi OR (+)
i. 1+1 = 1
ii. 1+0 = 1
iii. 0+0 = 0

2) Operasi AND (·)
i. 1·1 = 1
ii. 1·0 = 0
iii. 0·0 = 0

3) Operasi NOT (')
i. 1' = 0
ii. 0' = 1

Operasi-operasi dalam aljabar boolean mengikuti beberapa hukum yang mengatur interaksi antar operan dan operator. Hukum-hukum ini memiliki kemiripan dengan hukum-hukum pada aljabar logika dan aljabar himpunan [4].

1) Hukum Identitas
i. a+0 = a
ii. a·1 = a

2) Hukum Idempoten
i. a+a = a
ii. a·a = a

3) Hukum Komplemen
i. a+a' = 1
ii. a·a' = 0

4) Hukum Dominasi
i. a+1 = 1
ii. a·0 = 0

5) Hukum Involusi
i. (a')' = a

6) Hukum Penyerapan
i. a+ab = a
ii. a·(a+b) = a

7) Hukum Komutatif
i. a+b = b+a
ii. a·b = b·a

8) Hukum Asosiatif
i. a+(b+c) = (a+b)+c
ii. a·(b·c) = (a·b) ·c

9) Hukum Distributif
i. a+(b·c) = (a+b)·(a+c)
ii. a·(b+c) = (a·b)+(a·c)

10) Hukum De Morgan
i. (a+b)' = a' · b'
ii. (a·b)' = a' + b'

Salah satu penerapan aljabar Boolean adalah dalam fungsi Boolean, yaitu pemetaan atau fungsi dalam bentuk ekspresi Boolean dengan peubah dan range hasil yang bernilai 0 atau 1. Setiap peubah pada fungsi Boolean (termasuk dalam bentuk komplemen) disebut sebagai literal. Bagian dari suatu fungsi Boolean yang mencakup seluruh literal nya secara lengkap dan dihubungkan oleh operasi yang sama disebut sebagai *minterm* (operasi product) atau *maxterm* (operasi sum) [5]. Suatu fungsi Boolean dapat disajikan dalam dua jenis bentuk kanonik:

1) Sum of Product (SOP)
Merupakan bentuk kanonik fungsi Boolean yang diperoleh dari penjumlahan minterm dari setiap output fungsi yang bernilai 1.

$$F(x,y,z) = xyz + x'yz' \quad (7)$$

2) Product of Sum (POS)
Merupakan bentuk kanonik fungsi Boolean yang diperoleh dari perkalian maxterm dari setiap output fungsi yang bernilai 0.

$$F(x,y,z) = (x+y+z)(x'+y+z') \quad (8)$$

Fungsi Boolean yang terlalu panjang dapat dipersingkat menjadi bentuk yang lebih sederhana. Salah satu metode untuk menyederhanakan fungsi Boolean adalah Peta Karnaugh (K-Map). Metode ini memanfaatkan sebuah diagram yang terdiri dari kotak-kotak bersisian yang merepresentasikan sebuah minterm. Literal yang memiliki tanda komplemen direpresentasikan dengan nilai 0 pada variabelnya, sementara untuk yang tidak memiliki tanda komplemen direpresentasikan dengan nilai 1. Kotak-kotak yang merepresentasikan minterm yang terdapat pada fungsi Boolean diisi dengan 1, sementara sisanya diisi dengan 0 [5].

$$F(x, y, z) = x'yz' + xyz' + xyz$$

x \ yz	00	01	11	10
0	0	0	0	1
1	0	0	1	1

Fig. 1. Contoh Peta Karnaugh [5]

Penyederhanaan fungsi Boolean menggunakan peta Karnaugh dilakukan dengan mencari gabungan kotak-kotak bertetangga yang bernilai 1 (untuk SOP) atau 0 (untuk POS) dengan ketentuan kotak-kotak tersebut membentuk suatu pasangan (1×2 kotak), kuad (1×4 atau 2×2 kotak), atau oktet (2×4 kotak). Dari gabungan kotak tersebut, kemudian disusun kembali fungsi boolean yang baru dengan mengabaikan literal yang sama dari pasangan, kuad, atau oktet yang diperoleh pada masing-masing minterm [5].

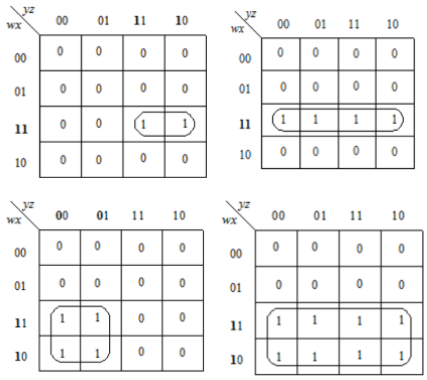


Fig. 2. Ilustrasi Penyederhanaan Peta Karnaugh [5]

D. Bilangan Biner

Bilangan biner merupakan salah satu bentuk representasi bilangan dalam basis 2 untuk menyatakan suatu bilangan hanya dalam '0' atau '1'. Bilangan desimal (basis 10) bisa dikonversi menjadi bilangan biner dengan mencari penjumlahan kombinasi bilangan perpangkatan 2 yang menghasilkan bilangan desimal tersebut [6].

Pada arsitektur komputer modern, bilangan bulat direpresentasikan dalam sistem biner menggunakan sistem *Two's Complement* yang berarti bit paling kiri dari suatu bilangan menyimpan nilai negatif. Oleh karena itu, dalam melakukan operasi bitwise untuk bilangan yang kecil, umumnya digunakan teknik *masking* yaitu menambahkan operasi AND dengan suatu bilangan yang representasi binernya berisi 1 semua sejumlah bit yang ditargetkan (misalnya adalah 31 atau 0x1F untuk masking pada bilangan biner 5-bit).

Manipulasi bilangan biner dapat dilakukan dengan operasi-operasi bitwise, yaitu operasi yang memanipulasi masing-masing bit dalam suatu bilangan biner, baik berupa operasi uner (seperti NOT) maupun operasi biner (AND, OR, dan XOR) [6].

Tabel 2.2 Tabel Kebenaran Operasi Uner

X	NOT X
0	1
1	0

Tabel 2.3 Tabel Kebenaran Operasi Biner

X	Y	X AND Y	X OR Y	X XOR Y
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

III. PEMBAHASAN

A. Formulasi Penyelesaian Permainan

Untuk menemukan penyelesaian varian The 24 Game berbasis operasi bitwise, nilai angka-angka pada kartu harus diubah menjadi representasinya dalam biner terlebih dahulu. Representasi biner dibuat dalam 5-bit agar mampu dimanipulasi menjadi representasi biner dari 24 yaitu 11000.

Tabel 3.1 Konversi Bilangan Desimal Ke Biner

Desimal (Basis 10)	Biner (Basis 2)
0	00000
1	00001
2	00010
3	00011
4	00100
5	00101
6	00110
7	00111
8	01000
9	01001
10	01010
11	01011
12	01100
13	01101

Operasi bitwise bekerja pada masing-masing bit secara independen, sehingga permasalahan dapat dipecah menjadi 5 masalah yang lebih kecil untuk mencari solusi pada masing-masing bit. Setiap bit dikodekan menjadi variabel x_{ij} (i = posisi bit, j = nomor urut kartu) pada aljabar Boolean dan fungsi pada masing-masing bit harus menghasilkan bit yang tepat dengan angka 24.

Tabel 3.2 Fungsi Boolean Pada Setiap Bit

Bit	Bit Angka Pada Kartu	Target Hasil
0	$F(x_{01}, x_{02}, x_{03}, x_{04})$	0
1	$F(x_{11}, x_{12}, x_{13}, x_{14})$	0
2	$F(x_{21}, x_{22}, x_{23}, x_{24})$	0
3	$F(x_{31}, x_{32}, x_{33}, x_{34})$	1
4	$F(x_{41}, x_{42}, x_{43}, x_{44})$	1

Pencarian solusi penyelesaian dari setiap set kartu (serta untuk menentukan apakah set tersebut solvable atau tidak) dapat dilakukan secara sistematis dan pasti dengan menggunakan pendekatan aljabar Boolean untuk meminimalisasi proses menebak-nebak atau berspekulasi. Sebagai contoh akan digunakan satu set kartu yang telah diketahui solvable yaitu kartu As (00001), 2 (00010), 4 (00100), dan 8 (01000). Langkah-langkahnya sebagai berikut,

1) Memecah permasalahan untuk posisi masing-masing bit
   i. Bit 0: $F(1,0,0,0) = 0$
   ii. Bit 1: $F(0,1,0,0) = 0$
   iii. Bit 2: $F(0,0,1,0) = 0$
   iv. Bit 3: $F(0,0,0,1) = 1$
   v. Bit 4: $F(0,0,0,0) = 1$
2) Menyusun tabel kebenaran untuk mengisi setiap kombinasi input dan output yang terlibat, sementara sisa baris output diisi dengan kondisi *Don't Care (X)*. Nilai bit angka pada masing-masing kartu direpresentasikan sebagai x_n yaitu angka pada kartu ke-n.

Tabel 3.3 Tabel Kebenaran Penyelesaian Set Kartu

x_1	x_2	x_3	x_4	Target
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	X
0	1	0	0	0
0	1	0	1	X
0	1	1	0	X
0	1	1	1	X
1	0	0	0	0
1	0	0	1	X
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

3) Buat peta Karnaugh yang merepresentasikan tabel kebenaran dan lakukan penyederhanaan fungsi Boolean menurut aturan penyederhanaan peta Karnaugh. Penyederhanaan perlu dilakukan jika penyelesaian yang dimiliki saat ini menggunakan sebuah kartu lebih dari satu kali.

$x_1\ x_2$ \ $x_3\ x_4$	00	01	11	10
00	1	1	X	0
01	0	X	X	X
11	X	X	X	X
10	0	X	X	X

Fig. 3. Peta Karnaugh Penyelesaian Set Kartu

$$F(x_1, x_2, x_3, x_4) = x_1' \cdot x_2' \cdot x_3' \quad (8)$$

4) Berdasarkan (8) telah diperoleh fungsi aljabar Boolean yang menghasilkan 11000, namun baru menggunakan 3 kartu yaitu x_1, x_2, x_3, oleh karena itu langkah terakhir yang perlu dilakukan adalah untuk mencari cara untuk melibatkan kartu keempat (01000) tanpa mengubah nilai fungsi 11000 yang sudah diperoleh. Dalam contoh ini, operasi yang dapat dilakukan adalah operasi tambah (OR) pada fungsi yang kita miliki saat ini, sehingga fungsinya menjadi

$$F(x_1, x_2, x_3, x_4) = (x_1' \cdot x_2' \cdot x_3') + x_4 \quad (9)$$

5) Langkah terakhir adalah untuk mengubah fungsi Boolean ke bentuk operasi bitwise (fungsi juga dapat dimanipulasi dengan hukum-hukum aljabar Boolean untuk menghasilkan bentuk solusi yang bervariasi).

6) Beberapa contoh solusi untuk set kartu As, 2, 4, 8 antara lain:

$$(\sim x_1\ \&\ \sim x_2\ \&\ \sim x_3)\ |\ x_4 \quad (10a)$$

$$(\sim(x_1\ |\ x_2)\ \&\ \sim x_3)\ |\ x_4 \quad (10b)$$

$$\sim(x_1\ |\ x_2\ |\ x_3)\ |\ x_4 \quad (10c)$$

Berdasarkan langkah-langkah tersebut, ditemukan beberapa karakteristik set kartu yang mengklasifikasikan variasi cara penyelesaian secara alajabar Boolean. Karakteristik-karakteristik tersebut antara lain,

1) Apabila pada posisi bit yang berbeda terdapat kombinasi input fungsi yang sama namun target outputnya berbeda, set kartu tersebut tidak solvable karena berdasarkan aturan fungsi, suatu input pada fungsi hanya bisa dipetakan ke satu jenis output.

2) Apabila set kartu hanya terdiri dari kombinasi angka yang kurang dari 4 {1,2,3} set kartu tersebut dipastikan tidak solvable. Hal ini karena angka-angka tersebut memiliki nilai bit ke-2 hingga ke-4 yang sama dengan 0, sehingga fungsi pada bit ketiga dan keempat F(0,0,0,0) mengharapkan output 1, sementara fungsi pada bit kedua F(0,0,0,0) mengharapkan output 0. Hal ini merujuk kembali ke karakteristik 1 yang menunjukkan bahwa set seperti ini tidak solvable.

3) Apabila set kartu terdiri dari 3 kartu yang ≤ 4 {1,2,3} dan 1 kartu yang ≥ 8, set kartu tersebut dipastikan tidak solvable. Hal ini karena angka-angka tersebut memiliki nilai bit ke-2 dan ke-4 yang sama dengan 0, sehingga fungsi pada bit ketiga dan keempat F(0,0,0,0) mengharapkan output 1, sementara fungsi pada bit kedua F(0,0,0,0) mengharapkan output 0. Hal ini merujuk kembali ke karakteristik 1 yang menunjukkan bahwa set seperti ini tidak solvable.

4) Apabila set kartu memiliki 4 angka yang sama persis (kecuali untuk angka 7), set kartu tersebut dipastikan tidak solvable. Hal ini karena selain angka 7, angka-angka lain memiliki angka 0 pada bit ke-0, ke-1, atau ke-2 yang dapat menyebabkan kontradiksi output fungsi pada bit tersebut dengan fungsi pada bit ke-3 atau ke-4.

5) Apabila set kartu hanya terdiri dari kombinasi angka yang kurang dari 8 namun ada yang lebih dari 3 (tanpa memiliki angka 0 pada bit ke-0, ke-1, atau ke-2 untuk keempat angka), set kartu tersebut dipastikan solvable. Hal ini karena pada setiap bit nya tidak mengalami kontradiksi fungsi, dan bit ketiga dan keempat memiliki nilai input dan output fungsi yang sama persis, sehingga output 1 pada tabel kebenaran hanya muncul sekali sehingga langsung diperoleh solusi $F(x_1, x_2, x_3, x_4) = x_1' \cdot x_2' \cdot x_3' \cdot x_4'$

B. Implementasi Kode Program

Analisis solvabilitas untuk setiap kombinasi set kartu dilakukan dengan bantuan kode program berbasis algoritma *State-Space Search* menggunakan *Depth-First Search* untuk melakukan pencarian semua kemungkinan prosedur dan operasi yang dibutuhkan untuk menyelesaikan semua set kartu dengan mereduksi array 4 kartu hingga menjadi 1 elemen terakhir secara rekursif.

Ketika jumlah elemen pada array kartu yang menjadi input dari fungsi masih lebih besar dari 1, program akan mencoba semua kemungkinan posisi operasi NOT, AND, dan OR untuk setiap elemen atau pasangan elemen. Hasil dari operasi tersebut akan di-masking oleh bilangan 0x1F untuk menghindari bilangan negatif, dan kemudian dimasukkan ke array baru yang jumlah elemennya telah berkurang 1 dari array yang lama. Array baru tersebut kemudian menjadi input baru untuk pemanggilan fungsi kembali secara rekursif. Proses ini terus diulang hingga jumlah elemen array tersisa 1, jika elemen tersebut sama dengan 24, maka solusi tersebut di simpan sebagai salah satu kemungkinan solusi, dan program akan melakukan backtracking untuk mencari kemungkinan solusi yang lain

```
if (n == 1) {
    if (arr[0].val == 24) {
        solved = true;
        totSolve += 1;

        if (!silentMode) {
            printf(" Solusi %d: %s = 24 (11000)\n", totSolve, arr[0].exp);
        }
    }
    return;
}

for (i=0; i<n; i++) {
    if (arr[i].legalNot) {
        Element notArr[4];
        for (j=0; j<n; j++) {
            notArr[j] = arr[j];
        }

        notArr[i].val = (~arr[i].val) & 0x1F; //NOT
        char tempArr[100];
        sprintf(notArr[i].exp, "(~%s)", arr[i].exp);

        notArr[i].legalNot = false;
        solveBitwise24(notArr, n);

        if (silentMode && solved) return;
    }
}
```

```
for (i=0; i<n; i++) {
    for (j=0; j<n; j++) {
        if (i != j) {
            Element newArr[4];
            int idx = 0;

            for (k=0; k<n; k++) {
                if (k!=i && k!=j) {
                    newArr[idx] = arr[k];
                    idx++;
                }
            }
            newArr[idx].val = (arr[i].val & arr[j].val) & 0x1F; //AND
            sprintf(newArr[idx].exp, "(%s & %s)", arr[i].exp, arr[j].exp);
            newArr[idx].legalNot = true;
            solveBitwise24(newArr, n-1);
            if (silentMode && solved) return;

            newArr[idx].val = (arr[i].val | arr[j].val) & 0x1F;//OR
            sprintf(newArr[idx].exp, "(%s | %s)", arr[i].exp, arr[j].exp);
            newArr[idx].legalNot = true;
            solveBitwise24(newArr, n-1);
            if (silentMode && solved) return;
        }
    }
}
```

Fig. 4. Algoritma pencarian solusi untuk sebuah set kartu

```
Solusi 7053: (8 | (~(1 | (4 | 2)))) = 24 (11000)
Solusi 7054: ((~(1 | (4 | 2))) | 8) = 24 (11000)
Solusi 7055: (8 | (~((4 | 2) | 1))) = 24 (11000)
Solusi 7056: ((~((4 | 2) | 1)) | 8) = 24 (11000)

Hasil: Sukses menemukan 7056 variasi solusi (SOLVABLE)!
```

Fig. 5. Contoh output solusi dari program untuk set {1,2,4,8}

Jumlah kombinasi untuk membentuk suatu set yang terdiri dari 4 kartu dihitung dengan menggunakan prinsip stars and bars untuk mendistribusikan 4 elemen (kartu) ke dalam 13 wadah (representasi jumlah kemunculan jenis kartu), sehingga dapat dimodelkan sebagai berikut,

$$n_1 + n_2 + n_3 + n_4 + n_5 + n_6 + n_7 + n_8 + n_9 + n_{10} + n_{11} + n_{12} + n_{13} = 4$$

$$C(4 + 13 - 1, 13 - 1) = C(16,12)$$

$$C(16,12) = \frac{16!}{(16-12)!\,12!} = \frac{16!}{4!\,12!}$$

$$C(16,12) = 1820 \; kombinasi$$

Seluruh 1820 kemungkinan set kartu tersebut kemudian diuji solvabilitasnya menggunakan fungsi pada kode program untuk menghitung persentase set kartu yang solvable.

```
void checkPossibilities() {
    printf("MEMULAI ENUMERASI 1.820 KEMUNGKINAN SET KARTU...\n");
    for (k1=1; k1<=13; k1++) {
        for (k2=k1; k2<=13; k2++) {
            for (k3=k2; k3<=13; k3++) {
                for (k4=k3; k4<=13; k4++) {
                    result[totalComb].kartu[0] = k1;
                    result[totalComb].kartu[1] = k2;
                    result[totalComb].kartu[2] = k3;
                    result[totalComb].kartu[3] = k4;

                    Element Kartu[4];
                    int curVal[4] = {k1,k2,k3,k4};
                    for (i=0; i<4; i++) {
                        Kartu[i].val = curVal[i] & 0x1F;
                        sprintf(Kartu[i].exp, "%d", curVal[i]);
                        Kartu[i].legalNot = true;
                    }

                    solved = false;
                    totSolve = 0;
                    solveBitwise24(Kartu,4);

                    result[totalComb].isSolvable = solved;

                    if (solved) {
                        countSolve += 1;
                    }
                    else {
                        countUnsolve += 1;
                    }
                    totalComb += 1;
```

Fig. 6. Algoritma perhitungan persentase set kartu solvable

Hasil perhitungan program menunjukkan ada 1115 set kartu yang solvable dari 1820 kombinasi yang ada, sehingga persentase kombinasi set kartu yang dapat diselesaikan adalah sebesar 61,26%. Berdasarkan penelitian terdahulu, persentase ini lebih rendah dari persentase solvabilitas pada versi normal dari The 24 Game yang sebesar 74,8% (1362 solvable dari 1820 kombinasi set kartu yang ada) [1]. Hal ini bisa terjadi karena batasan karakteristik operasi bitwise untuk memanipulasi setiap bit secara independen, sehingga tidak ada sistem carry-over seperti yang terjadi pada operasi aritmetika.

solvabilitas24Bitwise

=== TABEL SOLVED ===			
1115 SET (61,26%)			
Kartu 1	Kartu 2	Kartu 3	Kartu 4
1	1	1	6
1	1	1	7
1	1	2	4
1	1	2	5
1	1	2	6
1	1	2	7
1	1	3	4
1	1	3	5
1	1	3	6
1	1	3	7
1	1	4	6
1	1	4	7
1	1	5	6
1	1	5	7

=== TABEL UNSOLVED ===			
705 SET (38,74%)			
Kartu 1	Kartu 2	Kartu 3	Kartu 4
1	1	1	1
1	1	1	2
1	1	1	3
1	1	1	4
1	1	1	5
1	1	1	8
1	1	1	9
1	1	1	10
1	1	1	11
1	1	1	12
1	1	1	13
1	1	2	2
1	1	2	3
1	1	2	8

Fig. 7. Tabel solvabilitas set kartu pada varian Bitwise The 24 Game

Berdasarkan karakteristik aljabar Boolean yang telah diperoleh sebelumnya, serta hasil pengolahan data kode program yang mendata seluruh 1820 kombinasi kartu, dapat disusun strategi permainan varian Bitwise The 24 Game yang dapat meningkatkan probabilitas seseorang untuk menang (menemukan solusi secepat mungkin).

1) Jika set kartu memiliki minimal salah satu dari keempat karakteristik unsolvable yang telah diperoleh sebelumnya, pemain dapat langsung menyimpulkan bahwa set kartu tersebut tidak bisa dicari solusinya.
2) Jika menemukan karakteristik set seperti pada karakteristik kelima sebelumnya, langsung bisa didapatkan solusi yaitu $\sim(x_1|x_2|x_3|x_4)$.
3) Operasi NOT (~) wajib digunakan minimal satu kali untuk memanipulasi bit keempat agar bernilai satu, karena dari semua nilai angka input yang mungkin (1–13) semuanya memiliki bit ke-4 bernilai 0.
4) Jika set kartu memiliki angka 7, berdasarkan hasil analisis program persentase solvabilitasnya adalah 100% yang berarti sudah dipastikan set tersebut memiliki sebuah solusi. Strategi utama jika terdapat angka 7 adalah untuk langsung melakukan operasi NOT pada angka 7 (00111) yang akan langsung menghasilkan angka 24 (11000). Langkah berikutnya adalah untuk menambahkan operasi OR atau AND pada angka-angka yang lain yang tidak akan menghasilkan angka 1 pada bit 0, 1 atau 2.
5) Keberadaaan angka 3, 5, dan 6 meskipun tidak memiliki tingkat solvabilitas yang sempurna, tetapi persentasenya juga tetap tinggi (di atas 70%). Hal ini karena angka-angka tersebut jika dinegasikan akan menghasilkan angka yang mendekati biner dari 24 yaitu (11100, 11010, atau 11001). Oleh karena itu, prioritaskan memberi operasi NOT pada angka 3, 5, atau 6.
6) Jika tidak menemukan angka-angka emas tersebut (3, 5, 6, atau 7), prioritaskan untuk menghasilkan angka 1 pada bit

3 dan 4 terlebih dahulu, kemudian baru buat bit sisanya menjadi 0.

IV. Kesimpulan

Permainan kartu *The 24 Game* dengan batasan baru, yaitu hanya menggunakan operasi bitwise, mampu menyajikan tantangan baru untuk mengasah logika dan kemampuan berpikir cepat dalam bentuk permainan yang sudah umum dikenal banyak orang.

Berdasarkan hasil pemodelan kombinasi set kartu dan pengujian seluruh kemungkinan set menggunakan algoritma Depth-First Search, ditemukan bahwa varian permainan ini memiliki tingkat solvabilitas sebesar 61,26% (1115 set solvable dari 1820 kombinasi). Persentase ini lebih rendah dibandingkan versi normal permainan *The 24 Game* yang memiliki tingkat solvabilitas sebesar 74,8%. Penurunan tingkat solvabilitas ini disebabkan oleh karakteristik operasi bitwise yang memanipulasi setiap bit dalam bilangan biner secara independen tanpa adanya sistem *carry-over*, sehingga banyak muncul kontradiksi antara nilai input untuk mencapai target.

Terdapat banyak cara dan strategi dalam varian permainan *The 24 Game* menggunakan operasi bitwise, namun pendekatan aljabar Boolean mampu meningkatkan intuisi dan kemampuan berpikir logis untuk menemukan solusi sebuah set kartu secepat mungkin. Berdasarkan hasil analisis karakteristik aljabar Boolean pada setiap kemungkinan set kartu, diperoleh beberapa karakteristik utama yang dapat digunakan untuk menemukan solusi dari sebuah set kartu, seperti wajib menggunakan operasi NOT, pencarian angka-angka emas dengan tingkat solvabilitas tinggi (3, 5, 6, dan 7), serta pengecekan karakteristik set kartu yang unsolvable.

V. Lampiran

Source code untuk analisis solvabilitas seluruh kombinasi set kartu: https://github.com/fritziez/Bitwise24GameSolver

Tabel set kartu *Solvable* dan *Unsolvable*: https://docs.google.com/spreadsheets/d/18ygyQcpaV-ksWcQeobPTq92Nd2vfZ3fK/edit?usp=drive_link&ouid=111811360575635888371&rtpof=true&sd=true

Video Makalah: https://youtu.be/1eS385sXCrs

VI. Ucapan Terima Kasih

Penulis mengucapkan puji dan syukur kepada Tuhan Yang Maha Esa, karena berkat rahmat dan karunia-Nya penulis dapat menyelesaikan makalah ini dengan baik dan tepat waktu. Penulis juga ingin menyampaikan terima kasih kepada Prof. Dr. Ir. Rinaldi, M.T. selaku dosen pengampu mata kuliah IF1220 Matematika Diskrit atas bimbingan dan ilmu pengetahuan yang telah diberikan sehingga penulis dapat menyelesaikan makalah ini. Tak lupa penulis juga ingin mengucapkan terima kasih kepada keluarga dan teman-teman yang telah memberi semangat dan dukungan selama penulis mengerjakan makalah ini hingga selesai. Penulis berharap bahwa makalah ini bisa bermanfaat dan dapat digunakan sebagai referensi baik bagi sesama pelajar, maupun masyarakat luas untuk terus mengembangkan ketertarikan dan pengetahuannya terhadap bidang ilmu yang berikaitan.

Referensi

[1] Z. Nayla, "Pendekatan kombinatorika dalam penyelesaian permainan kartu '24'", 2024. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/Makalah/Makalah-IF1220-Matdis-2024%20(129).pdf [Accessed: Jun. 15, 2026].

[2] K. Louis Caesa, "Analisis end-to-end encryption dengan Diffie-Hellman key exhange menggunakan prinsip aljabar Boolean dan teori bilangan", 2022. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2022-2023/Makalah2022/Makalah-Matdis-2022%20(5).pdf [Accessed: Jun. 15 2026].

[3] M. Rinaldi, "Kombinatorika (Bagian 1)", 2026. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/18-Kombinatorika-Bagian1-2026.pdf [Accessed: Jun. 15 2026].

[4] M. Rinaldi, "Aljabar Boolean (Bagian 1)", 2026. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/10-Aljabar-Boolean-(2026)-bagian1.pdf [Accessed: Jun. 15 2026].

[5] M. Rinaldi, "Aljabar Boolean (Bagian 2)", 2026. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/11-Aljabar-Boolean-(2026)-bagian2.pdf [Accessed: Jun. 15 2026].

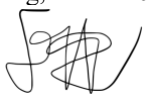
[6] A. Hairil, "Penggunaan mesin Turing multitrack untuk operasi bilangan biner", 2014. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/TeoriKomputasi/2014-2015/Makalah2014/Makalah-IF5110-2014-02.pdf [Accessed: Jun. 15 2026].

[7] M. John, "Twenty-Four", 2009. [Online]. Available: https://www.pagat.com/adders/24.html [Accessed: Jun. 14 2026].

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 1 Juni 2025



Pasaribu Fritz T.A.M.
1352505