Bulletin of Artificial Intelligence
Volume 3, No 2, October 2024 Page: 42−49
ISSN 2962-3944 (media online)
DOI: 10.62866/buai.v3i2.200
https://journal.grahamitra.id/index.php/buai

Implementasi Algoritma Sattolo Shuffle Untuk Optimasi Pengacakan Pada Game Solitaire

Surya Darma Nasution*, Guidio Leonarde Ginting

Program Studi Manajemen, Sekolah Tinggi Ilmu Manajemen Sukma Medan, Medan, Indonesia
Email: 1,*darmashadow@gmail.com, 2guidio.leonard626@gmail.com
Email Penulis Korespondensi: darmashadow@gmail.com

Abstrak−Permainan solitaire merupakan salah satu jenis permainan kartu digital yang sangat populer di berbagai platform. Salah satu aspek kritis dalam pengembangan game solitaire adalah mekanisme pengacakan kartu yang menentukan kualitas pengalaman bermain. Pengacakan yang tidak optimal dapat menghasilkan pola yang mudah diprediksi, sehingga mengurangi tantangan dan daya tarik permainan. Penelitian ini bertujuan untuk mengimplementasikan algoritma Sattolo Shuffle sebagai metode pengacakan kartu pada game solitaire sekaligus mengevaluasi efektivitasnya dalam menghasilkan permutasi siklis yang optimal. Algoritma Sattolo Shuffle merupakan modifikasi dari algoritma Fisher-Yates Shuffle yang dirancang khusus untuk menghasilkan permutasi siklis, di mana setiap elemen dipastikan berpindah dari posisi asalnya. Hasil pengujian menunjukkan bahwa algoritma Sattolo Shuffle mampu menghasilkan distribusi kartu yang benar-benar acak tanpa pola berulang pada 52 kartu solitaire, menjaga tingkat kesulitan permainan secara konsisten, dan memberikan pengalaman bermain yang lebih dinamis dibandingkan metode pengacakan konvensional. Penelitian ini memberikan kontribusi terhadap pengembangan game kartu digital, khususnya dalam optimasi mekanisme pengacakan untuk meningkatkan kualitas gameplay.

Kata Kunci: Algoritma Sattolo Shuffle; Pembangkit Bilangan Acak; Game Solitaire; Pengacakan Kartu; Permutasi Siklis

Abstract−Solitaire is one of the most popular digital card games across various platforms. A critical aspect in the development of solitaire games is the card shuffling mechanism that determines the quality of gameplay experience. Suboptimal shuffling can produce predictable patterns, thereby reducing the challenge and appeal of the game. This study aims to implement the Sattolo Shuffle algorithm as a card shuffling method in solitaire games while evaluating its effectiveness in producing optimal cyclic permutations. The Sattolo Shuffle algorithm is a modification of the Fisher-Yates Shuffle algorithm specifically designed to generate cyclic permutations, where each element is guaranteed to move from its original position. Test results demonstrate that the Sattolo Shuffle algorithm can produce truly random card distributions without repeating patterns across 52 solitaire cards, consistently maintaining game difficulty levels, and providing a more dynamic playing experience compared to conventional shuffling methods. This research contributes to the development of digital card games, particularly in optimizing shuffling mechanisms to enhance gameplay quality.

Keywords: Sattolo Shuffle Algorithm; Random Number Generator; Solitaire Game; Card Shuffling; Cyclic Permutation

1. PENDAHULUAN

Perkembangan teknologi informasi telah mendorong pertumbuhan industri permainan digital secara signifikan. Di antara berbagai jenis permainan digital yang populer, permainan kartu solitaire menempati posisi istimewa sebagai salah satu permainan klasik yang telah dimainkan oleh jutaan orang di seluruh dunia. Solitaire, yang juga dikenal dengan nama Patience atau Klondike, merupakan permainan kartu satu pemain yang membutuhkan kombinasi keterampilan strategis dan unsur keberuntungan dalam penyusunan kartu. Permainan ini menggunakan satu set standar 52 kartu yang terdiri dari empat jenis suit yaitu hati, berlian, keriting, dan sekop, dengan setiap suit berisi 13 kartu dari As hingga King [1], [1].

Salah satu aspek fundamental yang menentukan kualitas pengalaman bermain solitaire adalah mekanisme pengacakan kartu sebelum permainan dimulai. Pengacakan yang optimal akan menghasilkan susunan kartu yang benar-benar acak, sehingga setiap permainan memberikan tantangan yang unik dan tidak dapat diprediksi. Sebaliknya, pengacakan yang kurang optimal dapat menyebabkan munculnya pola-pola tertentu yang mudah dikenali oleh pemain berpengalaman, sehingga mengurangi tingkat kesulitan dan daya tarik permainan secara keseluruhan [3], [2]. Oleh karena itu, diperlukan suatu algoritma pembangkit bilangan acak yang handal untuk memastikan bahwa distribusi kartu pada setiap sesi permainan bersifat acak dan tidak berpola.

Berbagai algoritma pembangkit bilangan acak telah dikembangkan dan diterapkan dalam konteks permainan digital. Linear Congruent Method (LCM) merupakan salah satu metode yang paling banyak digunakan karena kecepatan dan kemudahan implementasinya. Namun, LCM memiliki kelemahan utama yaitu hasil bilangan acak yang dihasilkan cenderung berpola dan berulang pada periode tertentu [3], [4]. Algoritma Fisher-Yates Shuffle merupakan alternatif lain yang populer. Algoritma ini mampu menghasilkan permutasi acak yang tidak bias, namun memiliki karakteristik bahwa beberapa elemen dapat tetap berada di posisi asalnya setelah proses pengacakan [5], [6], [7].

Untuk mengatasi keterbatasan tersebut, penelitian ini mengusulkan penerapan algoritma Sattolo Shuffle sebagai metode pengacakan kartu pada game solitaire. Algoritma Sattolo Shuffle merupakan modifikasi dari algoritma Fisher-Yates Shuffle yang diperkenalkan oleh Sandra Sattolo pada tahun 1986. Perbedaan fundamental antara kedua algoritma ini terletak pada mekanisme pemilihan indeks acak, di mana pada Sattolo Shuffle indeks acak dipilih secara eksklusif dari rentang yang lebih kecil dari indeks saat ini, sehingga menghasilkan permutasi siklis di mana setiap elemen dipastikan berpindah dari posisi asalnya [8], [9].

Bulletin of Artificial Intelligence
Volume 3, No 2, October 2024 Page: 42−49
ISSN 2962-3944 (media online)
DOI: 10.62866/buai.v3i2.200
https://journal.grahamitra.id/index.php/buai

Penelitian terkait penerapan algoritma pengacakan pada permainan digital telah banyak dilakukan. Yusfrizal dkk. menerapkan algoritma Fisher-Yates Shuffle pada game edukasi mencocokkan gambar monumen dunia [10]. Ali Imron Panjaitan menggunakan metode Multiplicative Random Number Generation pada permainan memory card games [11]. Sugeng dkk. menerapkan Linear Congruent Method pada game edukasi pengenalan huruf alfabet [12]. Fujiati dan Lestari menerapkan algoritma Fisher-Yates Shuffle pada game edukasi sebagai media pembelajaran [13]. Diharjo dkk. menggunakan Fisher-Yates Shuffle pada game puzzle [14].

Meskipun berbagai penelitian telah mengeksplorasi penerapan algoritma pengacakan pada permainan digital, penerapan spesifik algoritma Sattolo Shuffle pada game solitaire dengan 52 kartu standar belum banyak dikaji. Penelitian ini bertujuan untuk mengimplementasikan algoritma Sattolo Shuffle pada game solitaire sebagai metode pengacakan kartu, sekaligus mengevaluasi efektivitasnya dalam menghasilkan distribusi kartu yang optimal.

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Penelitian ini termasuk ke dalam jenis penelitian eksperimental yang bertujuan untuk menerapkan dan mengevaluasi algoritma Sattolo Shuffle pada permainan solitaire. Tahapan penelitian meliputi:

1. Studi literatur tentang algoritma pengacakan dan permainan solitaire;
2. Perancangan sistem yang mendukung representasi 52 kartu dalam array dan distribusi ke layout solitaire;
3. Implementasi algoritma Sattolo Shuffle untuk pengacakan 52 kartu; dan
4. Pengujian serta evaluasi hasil pengacakan.

2.2 Permainan Solitaire

Solitaire atau Klondike merupakan permainan kartu satu pemain menggunakan 52 kartu standar. Tujuan permainan adalah memindahkan seluruh kartu ke empat tumpukan foundation yang diurutkan berdasarkan suit dari As hingga King. Permainan dimulai dengan mendistribusikan kartu ke tujuh kolom tableau (kolom ke-i berisi i kartu), di mana hanya kartu paling atas yang menghadap ke atas. Kartu tersisa ditempatkan pada stock pile [1].

2.3 Permainan Solitaire

Algoritma Sattolo Shuffle merupakan modifikasi dari algoritma Fisher-Yates Shuffle yang diperkenalkan oleh Sandra Sattolo pada tahun 1986. Perbedaan utama terletak pada rentang pemilihan indeks acak: pada Fisher-Yates indeks j dipilih dari 0 hingga i (inklusif), sedangkan pada Sattolo dari 0 hingga i-1 (eksklusif terhadap i). Perbedaan ini menghasilkan permutasi siklis dari (n-1)! kemungkinan, di mana setiap elemen dipastikan berpindah dari posisi asalnya [8], [9], [13].

Langkah-langkah algoritma Sattolo Shuffle:

1. Input Data: Masukkan 52 kartu ke dalam array A[0..51].
2. Inisialisasi: Set i = panjang array (52).
3. Pemilihan Bilangan Acak: Pilih bilangan acak r antara 0 dan i-1 (eksklusif terhadap i).
4. Pertukaran Elemen: Tukar elemen A[i-1] dengan A[r].
5. Iterasi: Kurangi i dengan 1. Jika i > 1, ulangi langkah 3; jika tidak, proses selesai.

yang terdapat pada Microsoft Word. Penomoran rumus di buat berurut berdasarkan urutan rumus yang terdapat pada artikel, dan penulisannya seperti (1).

3. HASIL DAN PEMBAHASAN

Pada permainan solitaire di penelitian ini, kartu yang digunakan berjumlah 52 kartu standar. Setiap kartu direpresentasikan dengan bilangan integer dari 1 sampai 52, di mana kartu 1-13 merepresentasikan suit Hati, kartu 14-26 suit Berlian, kartu 27-39 suit Keriting, dan kartu 40-52 suit Sekop. Array awal sebelum pengacakan adalah A = [1, 2, 3, 4, ..., 51, 52] dengan indeks 0 sampai 51.

3.1 Proses Iterasi Pengacakan 52 Kartu

Berikut ini adalah proses pengacakan lengkap 52 kartu menggunakan algoritma Sattolo Shuffle. Proses pengacakan membutuhkan 51 iterasi (n-1 iterasi untuk n=52 elemen). Pada setiap iterasi, dipilih bilangan acak r dari rentang 0 sampai i-1 (eksklusif terhadap i), kemudian dilakukan pertukaran elemen pada posisi i dengan elemen pada posisi r.
Diketahui:

1. Array A = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52]
2. Panjang array n = 52, sehingga dilakukan 51 iterasi.
3. Proses iterasi dibagi menjadi 2 tabel yang setiap tabel masing-masing terdapat 25 atau 26 proses iterasi, hal ini dilakukan dilakukan karena proses yang dilakukan terlalu panjang untuk disajikan dalam 1 tabel dalam 1 halaman.

Bulletin of Artificial Intelligence
Volume 3, No 2, October 2024 Page: 42−49
ISSN 2962-3944 (media online)
DOI: 10.62866/buai.v3i2.200
https://journal.grahamitra.id/index.php/buai

Tabel 1. Proses Iterasi Tahap-1 Algoritma Sattolo Shuffle pada 52 Kartu Solitaire

Iterasi	i	r	Operasi Tukar	Hasil Array A
1	51	48	A[51]↔A[48]	[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28, 29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,52,50,51,49]
2	50	30	A[50]↔A[30]	[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28, 29,30,51,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,52,50,31,49]
3	49	6	A[49]↔A[4]	[1,2,3,4,5,6,50,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28, 29,30,51,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,52,7,31,49]
4	48	16	A[48]↔A[14]	[1,2,3,4,5,6,50,8,9,10,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,51,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,17,7,31,49]
5	47	35	A[47]↔A[35]	[1,2,3,4,5,6,50,8,9,10,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,51,32,33,34,35,48,37,38,39,40,41,42,43,44,45,46,47,36,17,7,31,49]
6	46	42	A[46]↔A[42]	[1,2,3,4,5,6,50,8,9,10,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,51,32,33,34,35,48,37,38,39,40,41,42,47,44,45,46,43,36,17,7,31,49]
7	45	34	A[45]↔A[34]	[1,2,3,4,5,6,50,8,9,10,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,51,32,33,34,46,48,37,38,39,40,41,42,47,44,45,35,43,36,17,7,31,49]
8	44	32	A[44]↔A[32]	[1,2,3,4,5,6,50,8,9,10,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,51,32,45,34,46,48,37,38,39,40,41,42,47,44,33,35,43,36,17,7,31,49]
9	43	9	A[43]↔A[7]	[1,2,3,4,5,6,50,8,9,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,51,32,45,34,46,48,37,38,39,40,41,42,47,10,33,35,43,36,17,7,31,49]
10	42	30	A[42]↔A[30]	[1,2,3,4,5,6,50,8,9,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,47,32,45,34,46,48,37,38,39,40,41,42,51,10,33,35,43,36,17,7,31,49]
11	41	1	A[41]↔A[1]	[1,42,3,4,5,6,50,8,9,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,47,32,45,34,46,48,37,38,39,40,41,2,51,10,33,35,43,36,17,7,31,49]
12	40	8	A[40]↔A[6]	[1,42,3,4,5,6,50,8,41,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,47,32,45,34,46,48,37,38,39,40,9,2,51,10,33,35,43,36,17,7,31,49]
13	39	1	A[39]↔A[1]	[1,40,3,4,5,6,50,8,41,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,47,32,45,34,46,48,37,38,39,42,9,2,51,10,33,35,43,36,17,7,31,49]
14	38	30	A[38]↔A[30]	[1,40,3,4,5,6,50,8,41,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27,28, 29,30,39,32,45,34,46,48,37,38,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
15	37	2	A[37]↔A[1]	[1,40,38,4,5,6,50,8,41,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27, 28,29,30,39,32,45,34,46,48,37,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
16	36	8	A[36]↔A[6]	[1,40,38,4,5,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27, 28,29,30,39,32,45,34,46,48,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
17	35	29	A[35]↔A[29]	[1,40,38,4,5,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,25,26,27, 28,29,48,39,32,45,34,46,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
18	34	24	A[34]↔A[24]	[1,40,38,4,5,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,46,26,27, 28,29,48,39,32,45,34,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
19	33	4	A[33]↔A[2]	[1,40,38,4,34,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,46,26,27, 28,29,48,39,32,45,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
20	32	0	A[32]↔A[0]	[45,40,38,4,34,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,46,26, 27,28,29,48,39,32,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
21	31	24	A[31]↔A[24]	[45,40,38,4,34,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,32,26, 27,28,29,48,39,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
22	30	26	A[30]↔A[26]	[45,40,38,4,34,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,32,26, 39,28,29,48,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
23	29	0	A[29]↔A[0]	[48,40,38,4,34,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,21,22,23,24,32,26, 39,28,29,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
24	28	20	A[28]↔A[20]	[48,40,38,4,34,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,29,22,23,24,32,26, 39,28,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
25	27	2	A[27]↔A[1]	[48,40,28,4,34,6,50,8,37,44,11,12,13,14,15,16,52,18,19,20,29,22,23,24,32,26, 39,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
26	26	16	A[26]↔A[14]	[48,40,28,4,34,6,50,8,37,44,11,12,13,14,15,16,39,18,19,20,29,22,23,24,32,26, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]

Setelah dilakukan proses iterasi yang menghasilkan urutan acak sebanyak 26, maka langkah selanjutnya adalah melanjutkan proses iterasi dari 27 sampai 51. Untuk proses iterasi selanjutnya dapat dilihat pada tabel 2 berikut ini.

Tabel 2. Proses Iterasi Tahap-2 Algoritma Sattolo Shuffle pada 52 Kartu Solitaire

Iterasi	i	r	Operasi Tukar	Hasil Array A
27	25	16	A[25]↔A[14]	[48,40,28,4,34,6,50,8,37,44,11,12,13,14,15,16,26,18,19,20,29,22,23,24,32,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
28	24	8	A[24]↔A[6]	[48,40,28,4,34,6,50,8,32,44,11,12,13,14,15,16,26,18,19,20,29,22,23,24,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
29	23	16	A[23]↔A[14]	[48,40,28,4,34,6,50,8,32,44,11,12,13,14,15,16,24,18,19,20,29,22,23,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
30	22	0	A[22]↔A[0]	[23,40,28,4,34,6,50,8,32,44,11,12,13,14,15,16,24,18,19,20,29,22,48,26,37,39,

Bulletin of Artificial Intelligence
Volume 3, No 2, October 2024 Page: 42−49
ISSN 2962-3944 (media online)
DOI: 10.62866/buai.v3i2.200
https://journal.grahamitra.id/index.php/buai

Iterasi	i	r	Operasi Tukar	Hasil Array A
				52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
31	21	20	A[21]↔A[20]	[23,40,28,4,34,6,50,8,32,44,11,12,13,14,15,16,24,18,19,20,22,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
32	20	16	A[20]↔A[14]	[23,40,28,4,34,6,50,8,32,44,11,12,13,14,15,16,22,18,19,20,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
33	19	0	A[19]↔A[0]	[20,40,28,4,34,6,50,8,32,44,11,12,13,14,15,16,22,18,19,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
34	18	0	A[18]↔A[0]	[19,40,28,4,34,6,50,8,32,44,11,12,13,14,15,16,22,18,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
35	17	1	A[17]↔A[1]	[19,18,28,4,34,6,50,8,32,44,11,12,13,14,15,16,22,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
36	16	0	A[14]↔A[0]	[22,18,28,4,34,6,50,8,32,44,11,12,13,14,15,16,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
37	15	0	A[13]↔A[0]	[16,18,28,4,34,6,50,8,32,44,11,12,13,14,15,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
38	14	7	A[12]↔A[5]	[16,18,28,4,34,6,50,15,32,44,11,12,13,14,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
39	13	12	A[11]↔A[10]	[16,18,28,4,34,6,50,15,32,44,11,12,14,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
40	12	8	A[10]↔A[6]	[16,18,28,4,34,6,50,15,14,44,11,12,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
41	11	3	A[9]↔A[3]	[16,18,28,12,34,6,50,15,14,44,11,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
42	10	6	A[8]↔A[4]	[16,18,28,12,34,6,11,15,14,44,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
43	9	0	A[7]↔A[0]	[44,18,28,12,34,6,11,15,14,16,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
44	8	0	A[6]↔A[0]	[14,18,28,12,34,6,11,15,44,16,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
45	7	1	A[5]↔A[1]	[14,15,28,12,34,6,11,18,44,16,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
46	6	4	A[4]↔A[2]	[14,15,28,12,11,6,34,18,44,16,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
47	5	1	A[3]↔A[1]	[14,6,28,12,11,15,34,18,44,16,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
48	4	0	A[2]↔A[0]	[11,6,28,12,14,15,34,18,44,16,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
49	3	2	A[3]↔A[1]	[11,6,12,28,14,15,34,18,44,16,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
50	2	0	A[1]↔A[0]	[12,6,11,28,14,15,34,18,44,16,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]
51	1	0	A[1]↔A[0]	[6,12,11,28,14,15,34,18,44,16,50,4,32,13,8,22,19,40,20,23,24,29,48,26,37,39, 52,38,21,45,27,46,1,5,25,30,41,3,47,42,9,2,51,10,33,35,43,36,17,7,31,49]

Setelah semua proses iterasi selesai, maka hasil akhirnya adalah = **[6, 12, 11, 28, 14, 15, 34, 18, 44, 16, 50, 4, 32, 13, 8, 22, 19, 40, 20, 23, 24, 29, 48, 26, 37, 39, 52, 38, 21, 45, 27, 46, 1, 5, 25, 30, 41, 3, 47, 42, 9, 2, 51, 10, 33, 35, 43, 36, 17, 7, 31, 49]**.

3.2 Verifikasi Hasil Pengacakan

Verifikasi hasil pengacakan merupakan tahapan kritis untuk membuktikan bahwa algoritma Sattolo Shuffle benar-benar menghasilkan permutasi siklis (cyclic permutation). Dalam teori permutasi, sebuah permutasi dikatakan siklis apabila tidak terdapat satupun elemen yang tetap berada di posisi asalnya setelah proses transformasi, yang dalam terminologi matematika disebut sebagai derangement atau fixed-point-free permutation [1], [8]. Verifikasi dilakukan melalui tiga mekanisme pengujian: (a) pengecekan fixed point pada seluruh 52 elemen, (b) analisis pola perpindahan antar posisi, dan (c) pelacakan siklus permutasi.

Mekanisme pertama adalah pengecekan fixed point, yaitu membandingkan nilai pada setiap indeks sebelum dan sesudah pengacakan. Pada array awal, indeks ke-i berisi nilai i+1. Jika setelah pengacakan masih terdapat indeks ke-i yang berisi nilai i+1, maka elemen tersebut merupakan fixed point dan permutasi tidak bersifat siklis. Tabel 3 menunjukkan hasil verifikasi untuk seluruh 52 posisi dalam format dua kolom.Berisi hasil implementasi penerapan metode, ataupun hasil dari pengujian metode.

Tabel 3. Verifikasi Perpindahan Seluruh 52 Elemen (Fixed Point Analysis)

Idx	Awal	Akhir	Pindah?	Idx	Awal	Akhir	Pindah?
0	1	6	Ya	26	27	52	Ya
1	2	12	Ya	27	28	38	Ya

Bulletin of Artificial Intelligence
Volume 3, No 2, October 2024 Page: 42−49
ISSN 2962-3944 (media online)
DOI: 10.62866/buai.v3i2.200
https://journal.grahamitra.id/index.php/buai

Idx	Awal	Akhir	Pindah?	Idx	Awal	Akhir	Pindah?
2	3	11	Ya	28	29	21	Ya
3	4	28	Ya	29	30	45	Ya
4	5	14	Ya	30	31	27	Ya
5	6	15	Ya	31	32	46	Ya
6	7	34	Ya	32	33	1	Ya
7	8	18	Ya	33	34	5	Ya
8	9	44	Ya	34	35	25	Ya
9	10	16	Ya	35	36	30	Ya
10	11	50	Ya	36	37	41	Ya
11	12	4	Ya	37	38	3	Ya
12	13	32	Ya	38	39	47	Ya
13	14	13	Ya	39	40	42	Ya
14	15	8	Ya	40	41	9	Ya
15	16	22	Ya	41	42	2	Ya
16	17	19	Ya	42	43	51	Ya
17	18	40	Ya	43	44	10	Ya
18	19	20	Ya	44	45	33	Ya
19	20	23	Ya	45	46	35	Ya
20	21	24	Ya	46	47	43	Ya
21	22	29	Ya	47	48	36	Ya
22	23	48	Ya	48	49	17	Ya
23	24	26	Ya	49	50	7	Ya
24	25	37	Ya	50	51	31	Ya
25	26	39	Ya	51	52	49	Ya

Berdasarkan Tabel 3, seluruh 52 elemen telah berhasil diverifikasi dan tidak ditemukan satupun fixed point (0 dari 52 elemen). Ini berarti probabilitas elemen tetap di posisi asal adalah 0%, yang secara matematis membuktikan bahwa hasil pengacakan merupakan permutasi siklis sempurna. Sebagai perbandingan, pada algoritma Fisher-Yates Shuffle standar, probabilitas terdapat minimal satu fixed point dalam permutasi acak n elemen mendekati 1 - 1/e (sekitar 63,2%), yang berarti sekitar 63% dari seluruh permutasi acak akan memiliki setidaknya satu elemen yang tidak berpindah [1], [9]. Mekanisme kedua adalah analisis pola perpindahan. Dilakukan penghitungan jarak perpindahan setiap elemen dari posisi asalnya, yaitu selisih absolut antara posisi awal dan posisi akhir. Sebagai contoh, elemen bernilai 1 yang semula di indeks 0 kini berada di indeks 32 (jarak 32 posisi), sedangkan elemen bernilai 52 berpindah dari indeks 51 ke indeks 26 (jarak 25 posisi). Variasi jarak perpindahan yang besar dan tidak seragam menunjukkan distribusi pengacakan yang merata dan tidak terpola [3], [13].

Mekanisme ketiga adalah pelacakan siklus permutasi. Dalam permutasi siklis yang dihasilkan Sattolo Shuffle, seluruh elemen membentuk satu siklus tunggal. Jika dimulai dari elemen tertentu dan mengikuti rantai perpindahannya, seluruh 52 elemen akan terlewati sebelum kembali ke elemen awal. Pada hasil ini, rantai siklus dimulai dari posisi 0 yang berisi nilai 6, kemudian posisi 5 berisi 15, posisi 14 berisi 8, dan seterusnya hingga kembali ke posisi 0 setelah melewati seluruh 52 posisi. Satu siklus tunggal yang mencakup semua elemen ini mengkonfirmasi keberhasilan algoritma [8], [9].

3.3 Distribusi Kartu ke Layout Solitaire

Setelah proses pengacakan selesai, 52 kartu yang telah teracak harus didistribusikan ke layout permainan solitaire sesuai dengan aturan standar Klondike. Distribusi dilakukan ke tiga area utama: tableau (area bermain, 28 kartu dalam 7 kolom), stock pile (tumpukan cadangan, 24 kartu), dan foundation (tumpukan tujuan, awalnya kosong). Proses distribusi dilakukan secara sekuensial dari array hasil pengacakan [1].

Area tableau terdiri dari 7 kolom dengan jumlah kartu bertingkat: kolom ke-i mendapat i kartu, sehingga total 1+2+3+4+5+6+7 = 28 kartu. Pada setiap kolom, hanya kartu paling atas yang menghadap ke atas (terbuka), sedangkan kartu lainnya menghadap ke bawah (tertutup). Sisa 24 kartu dari array ditempatkan pada stock pile. Foundation dimulai kosong dan diisi secara berurutan dari As hingga King per suit. Berisi hasil implementasi penerapan metode, ataupun hasil dari pengujian metode.

Tabel 4. Distribusi Kartu Hasil Pengacakan ke Layout Solitaire

Area	Jml	Nomor Kartu	Kartu Teratas (Terbuka)
Kolom 1	1	6	6 Hati
Kolom 2	2	12, 11	J Hati
Kolom 3	3	28, 14, 15	2 Berlian
Kolom 4	4	34, 18, 44, 16	3 Berlian
Kolom 5	5	50, 4, 32, 13, 8	8 Hati
Kolom 6	6	22, 19, 40, 20, 23, 24	J Berlian

Bulletin of Artificial Intelligence
Volume 3, No 2, October 2024 Page: 42−49
ISSN 2962-3944 (media online)
DOI: 10.62866/buai.v3i2.200
https://journal.grahamitra.id/index.php/buai

Area	Jml	Nomor Kartu	Kartu Teratas (Terbuka)
Kolom 7	7	29, 48, 26, 37, 39, 52, 38	Q Keriting
Stock Pile	24	21, 45, 27, 46, 1, 5, 25, 30, 41, 3, 47, 42, 9, 2, 51, 10, 33, 35, 43, 36, 17, 7, 31, 49	*(Semua tertutup)*
Foundation	0	*(Kosong)*	*Menunggu As dari tiap suit*

Untuk menganalisis kualitas distribusi, dilakukan penghitungan sebaran suit pada setiap area. Setiap kartu termasuk dalam salah satu dari empat suit: Hati (1–13), Berlian (14–26), Keriting (27–39), dan Sekop (40–52). Tabel 5 menyajikan hasilnya.

Tabel 5. Analisis Distribusi Suit pada Layout Solitaire

Area	Hati (1-13)	Berlian (14-26)	Keriting (27-39)	Sekop (40-52)
Tableau (28 kartu)	6	10	7	5
Stock Pile (24 kartu)	7	3	6	8
Total	13	13	13	13

Berdasarkan Tabel 5, distribusi suit pada tableau menunjukkan persebaran: 6 kartu Hati, 10 kartu Berlian, 7 kartu Keriting, dan 5 kartu Sekop. Pada stock pile: 7 Hati, 3 Berlian, 6 Keriting, 8 Sekop. Persebaran ini menunjukkan tidak terdapat konsentrasi berlebihan dari satu suit pada satu area, mengindikasikan kualitas pengacakan yang baik. Distribusi merata antar suit penting karena mempengaruhi tingkat kesulitan dan variasi strategi pemain [1], [14].

Selain itu, kartu-kartu terbuka pada setiap kolom tableau menunjukkan variasi suit dan nilai yang baik, memastikan pemain memiliki beberapa opsi strategi sejak awal permainan. Hal ini sejalan dengan prinsip game design bahwa tingkat kesulitan harus seimbang untuk menjaga engagement pemain [10].

3.4 Analisis Efektifitas Algoritma

Untuk mengevaluasi efektivitas algoritma Sattolo Shuffle secara komprehensif dalam konteks permainan solitaire, analisis dilakukan dari empat perspektif:

1. Perbandingan dengan algoritma pengacakan lainnya,
2. Analisis kompleksitas komputasi,
3. Pengujian konsistensi melalui pengacakan berulang, dan
4. Evaluasi kesesuaian untuk game solitaire.

3.4.1 Perbandingan Dengan Algoritma Lain

Tabel 6. Perbandingan Algoritma Sattolo Shuffle dengan Metode Lainnya

Aspek	Sattolo Shuffle	Fisher-Yates Shuffle	LCM
Jenis Permutasi	Siklis	Acak Umum	Deterministik
Jumlah Kemungkinan	(n-1)! = 51!	n! = 52!	Terbatas (periodik)
Elemen Tetap di Posisi Asal	Tidak mungkin (0%)	Mungkin (~63%)	Tidak relevan
Pola Berulang	Tidak ada	Tidak ada	Ada (periodik)
Kompleksitas Waktu	O(n)	O(n)	O(n)
Kompleksitas Ruang	O(1) in-place	O(1) in-place	O(n) baru
Jumlah Iterasi (n=52)	51	51	52+
Derangement	Selalu (guaranteed)	Tidak selalu	Tidak dijamin
Kesesuaian Solitaire	Sangat Baik	Baik	Kurang Baik

Berdasarkan Tabel 6, sifat derangement Sattolo Shuffle menjamin 100% elemen berpindah posisi. Pada Fisher-Yates, probabilitas seluruh elemen berpindah hanya sekitar 36,8% (1/e), sehingga rata-rata terdapat 1 kartu yang tetap di posisi asal pada setiap pengacakan. Dibandingkan LCM yang bersifat deterministik dan menghasilkan pola periodik, Sattolo Shuffle menghasilkan distribusi yang lebih tidak terduga [1], [2], [3], [4], [9].

3.4.2 Analisis Kompleksitas Komputasi

Algoritma Sattolo Shuffle memiliki kompleksitas waktu O(n) dan kompleksitas ruang O(1). Untuk 52 kartu, dilakukan tepat 51 operasi pertukaran, masing-masing terdiri dari: (1) pembangkitan bilangan acak r dalam rentang [0, i-1], (2) penyimpanan sementara nilai A[i], dan (3) pertukaran nilai A[i] dengan A[r]. Pengacakan bersifat in-place tanpa alokasi memori tambahan selain variabel sementara, menjadikannya lebih efisien dibandingkan LCM yang memerlukan array terpisah [3], [3], [8], [12].

3.4.3 Pengujian Konsistensi Pengacakan Berulang

Untuk memastikan konsistensi kinerja, dilakukan tiga kali pengujian berulang terhadap 52 kartu yang sama. Tabel 7 menunjukkan hasilnya.

Bulletin of Artificial Intelligence
Volume 3, No 2, October 2024 Page: 42−49
ISSN 2962-3944 (media online)
DOI: 10.62866/buai.v3i2.200
https://journal.grahamitra.id/index.php/buai

Tabel 7. Hasil Pengujian Konsistensi Pengacakan Berulang

Parameter	Pengujian 1	Pengujian 2	Pengujian 3
Jumlah Iterasi	51	51	51
Fixed Point	0 (0%)	0 (0%)	0 (0%)
Pola Berulang	Tidak ditemukan	Tidak ditemukan	Tidak ditemukan
Jenis Permutasi	Siklis (1 siklus)	Siklis (1 siklus)	Siklis (1 siklus)
Distribusi Suit Merata	Ya	Ya	Ya
Hasil Identik dgn Lainnya	Tidak	Tidak	Tidak

3.4.4 Evaluasi Kesesuaian Untuk Game Solitaire

Dari perspektif game design, algoritma Sattolo Shuffle memberikan beberapa keuntungan spesifik untuk solitaire. Pertama, jaminan bahwa setiap kartu berpindah dari posisi asalnya memastikan setiap sesi permainan benar-benar baru dan berbeda, mencegah pemain menghafal pola tertentu [11], [13]. Kedua, distribusi suit yang merata pada tableau dan stock pile memastikan tingkat kesulitan yang seimbang, menghindari konsentrasi berlebihan dari satu suit pada satu area yang dapat membuat permainan terlalu mudah atau terlalu sulit [14]. Ketiga, efisiensi komputasi O(n) waktu dan O(1) ruang menjadikannya sangat cocok untuk implementasi pada berbagai platform termasuk perangkat mobile dengan sumber daya terbatas.

4. KESIMPULAN

Penelitian ini berhasil mengimplementasikan algoritma Sattolo Shuffle pada permainan solitaire untuk mengoptimalkan proses pengacakan 52 kartu. Proses pengacakan dilakukan melalui 51 iterasi, dan hasil verifikasi menunjukkan bahwa tidak ada satupun kartu yang tetap berada di posisi asalnya (0 fixed point), membuktikan keberhasilan algoritma dalam menghasilkan permutasi siklis yang sempurna. Algoritma Sattolo Shuffle terbukti efisien dengan kompleksitas waktu O(n) dan kompleksitas ruang O(1), menjadikannya cocok untuk implementasi pada berbagai platform permainan digital. Distribusi kartu ke layout solitaire menunjukkan penyebaran suit yang merata tanpa pola tertentu, sehingga meningkatkan kualitas pengalaman bermain. Sebagai saran untuk pengembangan lebih lanjut, algoritma Sattolo Shuffle dapat diterapkan pada variasi permainan solitaire lainnya seperti Spider Solitaire, FreeCell, dan Pyramid, serta dapat dikombinasikan dengan algoritma pembangkit bilangan acak lainnya untuk menghasilkan pengacakan yang lebih kompleks.

REFERENCES

[1] A. Ibijola and A. Olu, “A Simulated Enhancement of Fisher-Yates Algorithm for Shuffling in Virtual Card Games using Domain-Specific Data Structures,” *Int. J. Comput. Appl.*, vol. 54, no. 11, pp. 975–8887, 2012.
[2] S. D. Nasution and S. Suginam, “Modifikasi Algoritma Fisher Yates Shuffle Menggunakan Linear Congruent Method Untuk Pembangkitan Bilangan Acak,” *Jurnal Ilmu Komputer*, vol. 12, no. 2, pp. 101–106, 2019.
[3] S. Bulolo, “Implementasi Metode Linear Congruent Method (LCM) pada Simulasi Ujian Akhir Sekolah Menengah Kejuruan Lolomatua,” in *Prosiding Seminar Nasional Teknologi Informatika*, 2019, pp. 60–64.
[4] S. Angelina and A. D. Wowor, “Optimasi Pembangkit Bilangan Acak Dengan Fungsi Polinomial Dan Kombinasi Metode Iterasi,” *JIKO (Jurnal Informatika dan Komputer)*, vol. 8, no. 2, p. 367, Sep. 2024, doi: 10.26798/jiko.v8i2.1313.
[5] M. Yohanna, F. G. N. Larosa, and D. F. Malau, “Aplikasi Ujian Try Out Dengan Menerapkan Algoritma Fisher Yates Shuffle,” *Jurnal Teknik Informatika*, vol. 14, no. 2, 2022.
[6] M. A. Hasan, S. Supriadi, and Z. Zamzami, “Implementasi Algoritma Fisher-Yates Untuk Mengacak Soal Ujian Online Penerimaan Mahasiswa Baru (Studi Kasus : Universitas Lancang Kuning Riau),” *Jurnal Nasional Teknologi dan Sistem Informasi*, vol. 3, no. 2, pp. 291–298, Sep. 2017, doi: 10.25077/teknosi.v3i2.2017.291-298.
[7] I. Febriani, R. Ekawati, U. Supriadi, and M. I. Abdullah, “Fisher-Yates shuffle algorithm for randomization math exam on computer based-test,” in *AIP Conference Proceedings*, American Institute of Physics Inc., Apr. 2021. doi: 10.1063/5.0042534.
[8] S. C. Santo and N. M. S. Iswari, “Design and Development of Animal Recognition Application Using Gamification and Sattolo Shuffle Algorithm on Android Platform Case Study: Kebun Binatang Ragunan,” *IJNMT (International Journal of New Media Technology)*, vol. IV, no. 1, 2017.
[9] Y. Arviansyah, N. Nurfaizah, and R. Waluyo, “Penerapan Algoritma Fisher Yates Shuffle Pada Aplikasi TOEFL Preparation Berbasis Web,” *Jurnal Buana Informatika*, vol. 11, no. 2, pp. 112–122, 2020.
[10] Yusfrizal, D. Adhar, U. Indriani, E. Panggabean, A. Sabir, and H. Kurniawan, “Application of the Fisher-Yates Shuffle Algorithm in the Game Matching the World Monument Picture,” in *2020 2nd International Conference on Cybernetics and Intelligent System, ICORIS 2020*, Institute of Electrical and Electronics Engineers Inc., Oct. 2020. doi: 10.1109/ICORIS50180.2020.9320766.
[11] A. Imron Panjaitan, “Perancangan Aplikasi Memory Card Games Dengan Menerapkan Metode Multiplicative Random Number Generation,” *Journal Global Tecnology Computer*, vol. 2, no. 1, pp. 24–30, 2022.
[12] S. Supriyadi, D. Hamdani, and Y. M. Furqon, “Rancang Bangun Alfabet Memory Game Menggunakan Linear Congruent Method (LCM),” *Jurnal Teknologi dan Manajemen Informatika*, vol. 3, no. 1, 2018.
[13] F. Fujiati and S. Lestari Rahayu, “Implementasi Algoritma Fisher Yate Shuffle Pada Game Edukasi Sebagai Media Pembelajaran,” *Cogito Smart Journal*, vol. 6, no. 1, 2020.

Bulletin of Artificial Intelligence
Volume 3, No 2, October 2024 Page: 42−49
ISSN 2962-3944 (media online)
DOI: 10.62866/buai.v3i2.200
https://journal.grahamitra.id/index.php/buai

[14] W. Diharjo, D. Ahkam Sani, and M. Firman Arif, "Game Edukasi Bahasa Indonesia Menggunakan Metode Fisher Yates Shuffle Pada Genre Puzzle Game," *INTEGER: Journal of Information Technology*, vol. 5, no. 2, pp. 23–35, 2020.