Analisis Kombinatorial Peluang dan Optimasi Resource pada Sistem Tuning Echo di Wuthering Waves

Sebastio Nugroho - 13525123
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: sebastionu15@gmail.com , 13525123@std.stei.itb.ac.id

***Abstrak*—Sistem *Echo* dalam permainan Wuthering Waves merupakan mekanik utama untuk memperkuat karakter, namun membutuhkan sumber daya (*resource*) berupa *tuner* dan *echo exp* yang sangat terbatas. Pemain sering kali menghabiskan *resource* secara tidak efisien karena sistem *tuning* menggunakan mekanisme acak (*Random number generation*) untuk menentukan atribut tambahan (*substat*). Makalah ini menggunakan pendekatan Kombinatorika untuk memodelkan sistem *tuning* tersebut dan menghitung probabilitas matematis dari setiap kemunculan *substat*. Dari hasil analisis ruang sampel dan probabilitas bersyarat, ditemukan bahwa persentase mendapatkan kombinasi *substat* yang sempurna sangatlah kecil. Oleh karena itu, makalah ini merumuskan strategi optimasi *resource* dengan *Markov Chain* untuk mendapatkan strategi yang optimal**

Kata Kunci—Kombinatorika, Echo Tuning, Wuthering Waves, Optimasi Resource, Peluang, Monte Carlo Markov Chain, Expected Value.

I. Pendahuluan

Dalam permainan *Action RPG* Wuthering Waves, mekanik utama untuk meningkatkan kekuatan karakter (*Resonator*) berpusat pada pengumpulan dan pengembangan *Echo* (sisa-sisa entitas dari monster *Tacet Discords)*. Setiap *Echo* memiliki atribut utama (*Mainstat*) serta lima slot atribut tambahan (*substat*) tersembunyi yang hanya dapat dibuka secara bertahap melalui proses *Tuning*. Karena mayoritas *Resonator* di dalam *game* berperan sebagai penghasil daya rusak (*Damage dealer*), atribut pengali kritikal seperti *Crit Rate* dan *Crit DMG* menjadi parameter paling krusial yang diincar oleh para pemain untuk mendongkrak performa karakter secara optimal, melampaui atribut sekunder lainnya seperti *ATK%* atau *Energy Regen*.

Namun, optimasi *stat*istik ini membentur batasan sumber daya (*resource*) di dalam *game* yang sangat ketat, meliputi kelangkaan item *Premium Tuner*, *Sealed Tube* selaku media penambah EXP, dan *Shell Credit*. Masalah esensial muncul akibat implementasi sistem probabilitas acak (*Random number generation*/RNG) penuh tanpa jaminan (*pity system*) dalam penentuan jenis *substat* yang muncul. Akibatnya, banyak pemain terjebak dalam siklus pemborosan *resource* yang masif karena terus memaksakan peningkatan level pada *Echo* yang secara matematis memiliki peluang sangat rendah untuk menghasilkan kombinasi atribut ideal, sehingga berujung pada akumulasi *stat* yang suboptimal.

Makalah ini disusun untuk menyelesaikan permasalahan manajemen sumber daya tersebut melalui pendekatan matematika diskrit, khususnya teori Kombinatorika. Melalui pemodelan ruang sampel, perhitungan peluang menggunakan kombinasi, serta penerapan Prinsip Inklusi-Eksklusi, makalah ini menganalisis probabilitas kemunculan kombinasi *substat* esensial secara rigid. Lebih lanjut, analisis dilakukan terhadap ekspektasi nilai (*Expected Value*) dari biaya pemutakhiran dan sistem pengembalian (*refund*) material untuk merumuskan sebuah aturan henti (*stopping rule*) yang presisi. Dengan demikian, kajian ini bertujuan memberikan landasan teoretis mengenai titik optimal bagi pemain untuk menghentikan proses *tuning* khususnya mengevaluasi efisiensi keputusan berhenti pada percobaan ke-2 (Level +10) guna menekan defisit *resource* secara signifikan.

II. Landasan Teori

A. Kaidah Dasar Menghitung (Rule of Product)

Dalam ilmu Kombinatorika dasar, penyusunan objek yang terdiri dari rangkaian tahapan kejadian independen dapat diselesaikan dengan Kaidah Perkalian (*Rule of Product*). Apabila sebuah kejadian pertama dapat terjadi dalam p cara, dan kejadian kedua dapat terjadi dalam q cara, maka kedua kejadian tersebut dapat terjadi secara berurutan dalam p x q cara. Kaidah ini diaplikasikan dalam makalah ini untuk menghitung peluang gabungan (irisan) antara mendapat *substat* yang diinginkan di beberapa slot, sekaligus mendapat *substat* "sampah" di slot lainnya.

B. Teori Kombinasi dan Relevansinya dengan Sistem Tuning

Kombinasi adalah cabang dari matematika diskrit yang digunakan untuk menghitung jumlah cara pemilihan elemen dari suatu himpunan tanpa memperhatikan urutannya. Hal ini berbeda dengan Permutasi yang menganggap urutan sebagai entitas yang berbeda (contoh: AB ≠ BA). Dalam konteks *tuning* Echo di Wuthering Waves:

1. Tanpa Pengembalian (*Without Replacement*): Sebuah Echo tidak dapat memiliki dua buah *substat ATK%* (tidak ada duplikasi elemen).
2. Tanpa Memperhatikan Urutan (*Order Does Not Matter*): Mendapatkan *Crit Rate* di slot ke-1 dan *CRIT DMG* di slot ke-5 memberikan efek peningkatan *stat*istik karakter yang sama persis jika dibandingkan dengan mendapatkan *CRIT DMG* di slot ke-1 dan *Crit Rate* di slot ke-5.

Oleh karena itu, permasalahan ini adalah kasus murni Kombinasi. Rumus untuk menghitung pemilihan tidak terurut dari r elemen yang diambil dari himpunan n elemen didefinisikan sebagai:

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

C. Teori Peluang dan Peluang Bersyarat

Dalam kajian teori peluang, himpunan semua kemungkinan hasil dari suatu percobaan disebut sebagai ruang sampel, yang umumnya dilambangkan dengan S. Sementara itu, kejadian didefinisikan sebagai himpunan bagian dari ruang sampel tersebut.

Jika suatu percobaan memiliki ruang sampel S dengan anggota yang berhingga banyaknya dan setiap titik sampel memiliki kesempatan yang sama untuk muncul, maka peluang dari suatu kejadian A didefinisikan sebagai rasio antara banyak anggota kejadian A terhadap banyak anggota ruang sampel S. Secara matematis, hal ini diformulasikan sebagai:

$$P(A) = \frac{n(A)}{n(S)}$$

dengan n(A) merepresentasikan banyak anggota kejadian A, dan n(S) merepresentasikan banyak anggota ruang sampel S. Di dalam analisis komputasional, nilai n(A) dan n(S) ini dihitung menggunakan kaidah kombinatorika.

Dalam sistem yang beroperasi secara sekuensial (bertahap), nilai probabilitas sering kali berubah akibat adanya informasi tambahan dari tahapan sebelumnya. Peluang bersyarat adalah peluang terjadinya suatu kejadian A bila diketahui bahwa kejadian B telah terjadi terlebih dahulu. Peluang bersyarat kejadian A bersyarat B dilambangkan dengan P(A|B), dan diformulasikan sebagai:

$$P(A|B) = \frac{P(A \cap B)}{n(S)}$$

D. Mekanik Sistem Echo

Dalam permainan Wuthering Waves, Echo memiliki tingkatan kelangkaan (*rarity*) yang direpresentasikan melalui sistem bintang (Bintang 2 hingga Bintang 5). Tingkat kelangkaan ini secara linier menentukan batas maksimum level pemutakhiran dan jumlah slot atribut tambahan (*substat*) yang dapat dibuka melalui proses *tuning*. Rincian tingkatan kelangkaan tersebut dipaparkan pada Tabel 1.

Rarity	**Representasi Warna**	**Level Maksimum**	**Slot *Substat***
Bintang 2	Hijau (*Green*)	10	0
Bintang 3	Biru (*Blue*)	15	3
Bintang 4	Ungu (*Purple*)	20	4
Bintang 5	Emas (*Gold*)	25	5

Tabel 1. Karakteristik Atribut Berdasarkan Tingkat Kelangkaan Echo. Sumber : [3]

Echo Bintang 5 merupakan target utama bagi pemain karena memberikan peningkatan *stat*istik tertinggi. Pemain dapat memperoleh Echo Bintang 5 melalui dua metode utama:

1. Peningkatan Level Data Bank: Pemain harus meningkatkan level Data Bank hingga mencapai Level 15 atau lebih tinggi agar Echo Bintang 5 dapat dijatuhkan (*drop*) secara alami oleh monster di dunia terbuka (open-world).
2. Fitur Data Merge: Mekanik alternatif yang mengizinkan pemain menggabungkan 5 buah Echo acak yang tidak terpakai di dalam *inventory*. Peluang konversi menjadi Bintang 5 akan meningkat drastis seiring dengan pencapaian Data Bank Level 15 ke atas.

Untuk mencapai Data Bank Level 15, pemain secara umum harus menyentuh Union Level 30 (kondisi yang rata-rata dapat dicapai setelah sekitar 10 jam akumulasi waktu bermain) dan Mengumpulkan echo dari *tacet discord* berbeda. Mengingat Echo Bintang 5 merupakan standar absolut yang dicari oleh seluruh pemain dalam optimasi karakter, maka seluruh pemodelan matematika dan kalkulasi peluang di dalam makalah ini secara eksklusif mengasumsikan penggunaan Echo Kelangkaan Bintang 5 (Emas).

E. Mekanik Sistem Echo, Substat, dan Tuning

Dalam permainan Wuthering Waves, Echo diklasifikasikan berdasarkan Cost (1, 3, dan 4) yang menentukan atribut utamanya (*Main Stat*). Namun, terlepas dari perbedaan Cost tersebut, potensi maksimal sebuah Echo sangat bergantung pada atribut tambahan yang disebut *substat*.

Substat merupakan atribut ekstra yang hanya dapat dibuka (di-unlock) melalui proses *tuning*. Proses kemunculan *substat* ini memiliki karakteristik probabilitas yang sepenuhnya independen dari *Main Stat*. Setiap *Echo* (*rarity* bintang 5) memiliki batas maksimal 5 slot *substat* yang posisinya tidak dapat diubah (di-reroll).

Berdasarkan mekanik *game*, rentang nilai (kualitas) dari setiap *substat* yang muncul juga bervariasi dan mengikuti

distribusi Gaussian, di mana persentase atribut memiliki 8 variasi roll, sementara atribut flat (seperti ATK dan DEF) memiliki 4 variasi roll. Data lengkap himpunan n=13 *substat* dan rentang nilainya ditunjukkan pada Tabel 2.

Substat	Range of Values	Median Value
ATK	30 - 60	50
HP	320 - 580	450
DEF	40 - 70	50
ATK%	6.4% - 11.6%	9.0%
HP%	6.4% - 11.6%	9.0%
DEF%	8.1% - 14.7%	11.39%
Energy Regen	6.8% - 12.4%	10.25%
Crit. Rate	6.3% - 10.5%	8.4%
Crit. DMG	12.6% - 21.0%	16.8%
Basic Attack DMG Bonus	6.4% - 11.6%	9.0%
Heavy Attack DMG Bonus	6.4% - 11.6%	9.0%
Resonance Skill DMG Bonus	6.4% - 11.6%	9.0%
Resonance Liberation DMG Bonus	6.4% - 11.6%	9.0%

Tabel 2. Himpunan *Substat* Echo dan Rentang Nilai. Sumber : [3]

Aturan probabilitas fundamental dari sistem *tuning* ini membuat sebuah echo tidak dapat memiliki jenis *substat* yang sama (duplikat) di dalam kelima slotnya dan hasil persentase *substat* yang dihasilkan juga *random*.

Untuk membuka slot *substat*, pemain harus menaikkan level Echo menggunakan EXP material dan *Shell Credit*, kemudian mengaktifkan slot tersebut menggunakan *Premium Tuner*. Setiap kelipatan 5 level membuka 1 slot *tuning* baru. Total biaya kumulatif ditunjukkan pada Tabel 3.

Level (Fase *Tuning*)	Total *Echo exp*	Total *Shell Credit*	Total *Premium Tuner*
+5 (*Tuning* ke-1)	4.400	2.440	10
+10 (*Tuning* ke-2)	16.500	5.650	20
+15 (*Tuning* ke-3)	39.600	9.960	30
+20 (*Tuning* ke-4)	79.100	15.910	40
+25 (*Tuning* ke-5)	142.600	24.260	50

Tabel 3. Akumulasi Biaya Leveling dan *Tuning* Echo Bintang 5. Sumber : [5]

Selain itu, *game* ini mengizinkan Echo yang gagal (ampas) dilebur menjadi EXP untuk Echo baru (Refund). Tingkat refund adalah 75% *Echo exp* dan 30% *Tuner*. *Shell Credit* hangus (tidak di-refund).

III. Pembahasan

A. *Analisis Ruang Sampel*

Berdasarkan mekanik *game*, kita dapat mendefinisikan:

1. Himpunan total *substat* yang tersedia adalah n = 13 dengan S = { ATK, HP, DEF, *ATK%*, HP%, DEF%, *Energy Regen*, *Crit Rate*, *Crit DMG*, Basic Attack DMG, Heavy Attack DMG, Resonance Skill DMG, Resonance Liberation DMG }.
2. Slot *substat* yang harus diisi pada satu Echo adalah r = 5.
3. Pengambilan dilakukan tanpa pengembalian (tidak ada *substat* kembar) dan tanpa memperhatikan urutan.

Untuk mengetahui total variasi susunan *substat* yang mungkin dimiliki sebuah Echo, kita menghitung kombinasi 5 elemen dari 13 elemen:

$$C(13,5) = \frac{13!}{5!(13-5)!} = 1287$$

Dengan demikian, terdapat 1287 kemungkinan susunan *substat* unik pada setiap *Echo*.

B. *Analisis Kombinatorial Berdasarkan Target Pemain*

Pemain dihadapkan pada kelangkaan sumber daya, sehingga probabilitas di setiap fase *tuning* harus dievaluasi untuk menghindari pemborosan material. Berikut adalah perhitungan peluang matematis pada kondisi awal (Level 0) dan probabilitas bersyarat pasca-kegagalan *tuning* (Level +5, +10, dst), dibagi berdasarkan jumlah target *substat* esensial. Diasumsikan bahwa player mengincar lebih dari 2 target *substat* esensial.

1. Target 2 *Substat* (Contoh: *Crit Rate* dan *Crit DMG*)

- Kondisi Awal (Level 0): Pemain harus mengamankan 2 *substat* target, sehingga sisa 3 slot akan diisi oleh *substat* acak dari 11 *substat* lainnya.

$$P = \frac{C(2,2)xC(11,3)}{C(13,5)} \approx 12.82\%$$

- Gagal di *Tuning* Pertama (Level +5): Jika 1 slot pertama terisi *substat* suboptimal, sisa slot kosong adalah 4 dengan himpunan tersisa 12 *substat*.

$$P = \frac{C(2,2)xC(10,2)}{C(12,4)} \approx 9.09\%$$

- Gagal di Dua *Tuning* Pertama (Level +10): Tersisa 3 slot kosong dan 11 *substat* di dalam himpunan. Peluang menyusut menjadi:

$$P = \frac{C(2,2)xC(9,1)}{C(11,3)} \approx 5.45\%$$

- Gagal di Tiga *Tuning* Pertama (Level +15): Tersisa 2 slot kosong dengan himpunan berkurang menjadi 10 *substat*.

Pemain kini diwajibkan mendapatkan kedua *substat* target tersebut di dua slot terakhir secara mutlak. Peluang anjlok menjadi:

$$P = \frac{C(2,2)xC(8,0)}{C(10,2)} \approx 2.22\%$$

2. Target 3 *Substat* (Contoh: *Crit Rate*, *Crit DMG*, dan *ATK%*)

- Kondisi Awal (Level 0): Pemain harus mengamankan 3 *substat* target, sehingga sisa 2 slot akan diisi oleh *substat* acak dari 10 *substat* lainnya.

$$P = \frac{C(3,3)xC(10,2)}{C(13,5)} \approx 3.50\%$$

- Gagal di *Tuning* Pertama (Level +5): Jika 1 slot pertama terisi *substat* suboptimal, sisa slot kosong adalah 4 dengan himpunan tersisa 12 *substat*.

$$P = \frac{C(3,3)xC(9,1)}{C(12,4)} \approx 1.82\%$$

- Gagal di Dua *Tuning* Pertama (Level +10): Tersisa 3 slot kosong dan 11 *substat* di dalam himpunan. Pemain wajib menyapu bersih sisa slot dengan *substat* target secara berturut-turut. Peluang anjlok menjadi:

$$P = \frac{C(3,3)xC(8,0)}{C(11,3)} \approx 0.61\%$$

3. Target 4 *Substat* (Contoh: *Crit Rate*, *Crit DMG*, *ATK%*, dan *Energy Regen*)

- Kondisi Awal (Level 0): Pemain mengamankan 4 *substat* target, sisa 1 slot diisi *substat* acak dari 9 *substat* lainnya.

$$P = \frac{C(4,4)xC(9,1)}{C(13,5)} \approx 0.7\%$$

- Gagal di *Tuning* Pertama (Level +5): Jika 1 slot pertama terisi *substat* suboptimal, sisa slot kosong adalah 4. Pemain diwajibkan menyapu bersih 4 sisa slot tersebut secara mutlak.

$$P = \frac{C(4,4)xC(8,0)}{C(12,4)} \approx 0.2\%$$

4. Target 5 *Substat* (*God Roll*)
Skenario idealis di mana pemain mengharapkan kelima slot terisi sempurna tanpa *substat* tidak terpakai.

$$P = \frac{C(5,5)}{C(13,5)} \approx 0.077\%$$

C. Simulasi Substat dengan Monte Carlo

Untuk memverifikasi perhitungan peluang kombinatorial secara empiris pada subbab sebelumnya, serangkaian program simulasi dikembangkan menggunakan bahasa pemrograman Python. Implementasi dibagi menjadi dua model komputasi: validasi probabilitas dengan Monte Carlo dan kalkulasi ekonomi dengan Relasi Rekurens.

1. Deklarasi Ruang Sampel dan Parameter Target

Program diawali dengan mendefinisikan ruang sampel himpunan semesta (n=13). Parameter target (*substat* yang dicari) dan syarat minimal kemenangan (wanted_*stat*) juga diatur.

```python
import random

all_substats = ['ATK', 'HP', 'DEF', 'ATK%', 'HP%', 'DEF%', 'Energy Regen',
                'Crit Rate', 'Crit DMG', 'Basic Attack DMG', 'Heavy Attack DMG',
                'Resonance Skill DMG', 'Resonance Liberation DMG']
wanted_stat = 2
target_substats = {'Crit Rate', 'Crit DMG', 'ATK%', 'Energy Regen', 'Basic Attack DMG'}
target_substat_count = len(target_substats)
```

Gambar 1. Deklarasi Ruang Sampel dan Parameter Simulasi.
Sumber : Dokumentasi Pribadi

2. Implementasi Algoritma Monte Carlo

Fungsi utama simulasi menjalankan perulangan (*looping*) sebanyak jumlah iterasi yang ditentukan (secara default 100.000 kali) berdasarkan hukum bilangan besar (*Law of Large Numbers*) untuk menekan variansi dan mendekati nilai probabilitas teoretis.

```python
def simulate_echo_tuning(simulations=100000):

    success_count = 0
    for _ in range(simulations):
        rolled_stats = set(random.sample(all_substats, 5))
        if len(rolled_stats.intersection(target_substats)) >= wanted_stat:
            success_count += 1

    return (success_count / simulations) * 100
```

Gambar 2. Simulasi Substat dengan Algoritma Monte Carlo.
Sumber : Dokumentasi Pribadi

3. Hasil

Setelah melakukan beberapa kali perubahan pada wanted_*stat* dan jumlah dari target_*substat*s, didapat hasil berikut :

Target *Stat*	*Stat* Sesuai Target	Persentase
5	5	0.089%
5	4	3.170%
5	3	25.063%
5	2	68.164%
5	1	95.771%
4	4	0.713%
4	3	11.961%
4	2	50.763%

4	1	90.272%
3	3	3.560%
3	2	31.308%
3	1	80.546%
2	2	13.048%
2	1	63.967%
1	1	38.484%

Tabel 4. Hasil Probabilitas Simulasi *Substat* dengan Monte Carlo. Sumber : Dokumentasi Pribadi

4. Analisis Hasil Simulasi

Angka yang dihasilkan dari eksperimen komputasi ini sangat presisi dan selaras dengan perhitungan matematis menggunakan rumus Kombinasi pada subbab sebelumnya.

a. Sebagai contoh, saat pemain menargetkan 5 *substat* dan berhasil mendapatkannya secara sempurna (baris pertama), hasil simulasi adalah 0.089%. Hal ini mendekati hitungan teori yakni 0.077%. Selisih desimal yang sangat kecil ini wajar terjadi dalam pendekatan probabilitas empiris *Monte Carlo*.

b. Ketika pemain menargetkan 2 *substat* prioritas (seperti *Crit Rate* dan *CRIT DMG*) dan ingin kedua-duanya muncul, simulasi menunjukkan angka 13.048%, yang membuktikan validitas perhitungan teori yakni 12.82%.

D. Simulasi dengan Expected Value dengan Markov Chain

Untuk memvalidasi relasi rekurens pada Rantai Markov dan mengukur dampak pada *resource player* dari setiap aturan henti (*stopping rule*), sebuah program kalkulator *Expected Value* (EV) dikembangkan menggunakan bahasa Python. Program ini mengkalkulasi probabilitas bersyarat dari Echo yang dibuang secara prematur, serta menimbang kerugian netto pasca-sistem refund.

1. Implementasi Fungsi Kombinasi

Program diawali dengan mendefinisikan fungsi kombinasi. Fungsi ini dirancang dengan mekanisme pencegahan galat (*error handling*) untuk menangani kondisi matematis yang mustahil (misalnya jika target *substat* lebih besar dari sisa slot).

```python
import math

def combination(n, r):
    if r > n or r < 0:
        return 0
    return math.comb(n, r)
```

Gambar 3. Implementasi Fungsi Kombinasi. Sumber : Dokumentasi Pribadi

2. Definisi Parameter dan Tabel Biaya

Fungsi utama kalkulasi_ev_gabungan menerima parameter target *substat* yang dicari dan batas checkpoint level (dalam satuan jumlah slot K).

```python
def calculate_combined_ev(target_count, k_slot):
    # 1. Cumulative cost table for leveling & 5-star tuning
    cost = {
        1: {'exp': 4400,   'cred': 2440,  'tuner': 10},
        2: {'exp': 16500,  'cred': 5650,  'tuner': 20},
        3: {'exp': 39600,  'cred': 9960,  'tuner': 30},
        4: {'exp': 79100,  'cred': 15910, 'tuner': 40},
        5: {'exp': 142600, 'cred': 24260, 'tuner': 50}
    }
```

Gambar 4. Definisi Parameter dan Tabel Biaya. Sumber : Dokumentasi Pribadi

3. Evaluasi Probabilitas Bersyarat

Program menghitung peluang Echo lolos seleksi awal (checkpoint). Lalu, menghitung *p_win_but_fail*, yaitu persentase Echo potensial yang secara kombinatorial dapat sukses di akhir, tetapi terlanjur dibuang oleh pemain karena gagal memunculkan target di fase awal.

```python
# 2. Sample space and absolute Echo win outcomes
total_combinations = combination(13, 5)
echo_win = combination(target_count, target_count) * combination(13 - target_count, 5 - target_count)

# 3. Probability of passing the checkpoint (at least 1 target appears in the first K slots)
min_targets_in_k = max(1, target_count - (5 - k_slot))
pass_combinations = 0
for i in range(min_targets_in_k, min(target_count, k_slot) + 1):
    pass_combinations += combination(target_count, i) * combination(13 - target_count, k_slot - i)
p_pass_checkpoint = pass_combinations / combination(13, k_slot)

# 4. Probability of wasted potential Echo
p_win_but_fail = combination(5 - target_count, k_slot) / combination(5, k_slot)
p_final_success = (echo_win / total_combinations) * (1 - p_win_but_fail)

if p_final_success <= 0 or p_pass_checkpoint <= 0:
    return float('inf'), float('inf'), float('inf'), float('inf')
```

Gambar 5. Implementasi Probabilitas Peluang Echo. Sumber : Dokumentasi Pribadi

4. Distribusi Status Rantai Markov dan Sistem *Refund*

Peluang sukses akhir digunakan untuk memprediksi total Echo mentah yang dibutuhkan (n_echo). Populasi Echo tersebut kemudian didistribusikan ke dalam tiga status Rantai Markov: *success*, *initial_failure* (langsung di-*refund*), dan *final_failures*. Sistem *refund* diaplikasikan untuk menghitung biaya netto dari setiap kegagalan (pengembalian 75% EXP, 30% *Tuner*, dan 0% *Credit*).

```python
# 5. Echo outcome distribution using a Markov chain approximation
n_echo = 1 / p_final_success
n_success = 1
n_initial_failures = n_echo * (1 - p_pass_checkpoint)
n_final_failures = (n_echo * p_pass_checkpoint) - 1

# 6. Net cost calculation
def net_cost(entry):
    return {
        'exp': entry['exp'] * 0.25,
        'cred': entry['cred'],
        'tuner': entry['tuner'] * 0.70
    }

net_initial = net_cost(cost[k_slot])
net_final = net_cost(cost[5])
```

Gambar 6. Distribusi *Markov Chain* dan Sistem *Refund*. Sumber : Dokumentasi Pribadi

5. Implementasi *Expected Value* (EV)

Pada tahap akhir, jumlah Echo pada masing-masing status transisi Markov dikalikan dengan biaya nettonya. Hasil akumulasi ini direturn sebagai *Expected Value* dari setiap resource utama.

```python
# 7. Total expected value
ev_exp = (n_success * cost[5]['exp']) + (n_initial_failures * net_initial['exp']) + (n_final_failures * net_final['exp'])
ev_cred = (n_success * cost[5]['cred']) + (n_initial_failures * net_initial['cred']) + (n_final_failures * net_final['cred'])
ev_tuner = (n_success * cost[5]['tuner']) + (n_initial_failures * net_initial['tuner']) + (n_final_failures * net_final['tuner'])

return n_echo, ev_exp, ev_cred, ev_tuner
```

Gambar 7. Implementasi Sistem EV. Sumber : Dokumentasi Pribadi

6. Pelaksana Simulasi

Sebuah fungsi yang dibuat untuk menjalankan algoritma utama secara iteratif dari Level +5 (k=1) hingga Level +25 (k=5), kemudian mencetak hasilnya.

```python
def run_full_simulation(target):
    print(f"\n=== EV for Target {target} Specific Substat ===")
    for k in range(1, 6):
        level = k * 5
        echo, exp, cred, tuner = calculate_combined_ev(target, k)
        print(f"Strategy +{level:<2} -> {echo:>5.1f} Echo | {tuner:>6.1f} Tuner | {exp:>8,.0f} EXP | {cred:>8,.0f} Credit")
```

Gambar 8. Implementasi Pelaksana Program. Sumber : Dokumentasi Pribadi

7. Hasil

```
=== EV for Target 2 Specific Substat ===
Strategy +5  ->  19.5 Echo |  235.5 Tuner |  232,050 EXP |  113,040 Credit
Strategy +10 ->  11.1 Echo |  240.0 Tuner |  256,496 EXP |  124,104 Credit
Strategy +15 ->   8.7 Echo |  248.3 Tuner |  287,167 EXP |  138,753 Credit
Strategy +20 ->   7.8 Echo |  262.8 Tuner |  327,870 EXP |  159,168 Credit
Strategy +25 ->   7.8 Echo |  288.0 Tuner |  385,020 EXP |  189,228 Credit

=== EV for Target 3 Specific Substat ===
Strategy +5  ->  47.7 Echo |  656.7 Tuner |  539,433 EXP |  356,327 Credit
Strategy +10 ->  31.8 Echo |  742.2 Tuner |  661,869 EXP |  429,746 Credit
Strategy +15 ->  28.6 Echo |  848.0 Tuner |  817,540 EXP |  522,236 Credit
Strategy +20 ->  28.6 Echo |  856.4 Tuner |  764,590 EXP |  503,456 Credit
Strategy +25 ->  28.6 Echo | 1016.0 Tuner | 1,126,540 EXP |  693,836 Credit

=== EV for Target 4 Specific Substat ===
Strategy +5  -> 178.8 Echo | 2806.2 Tuner | 2,203,825 EXP | 1,636,250 Credit
Strategy +10 -> 143.0 Echo | 3634.0 Tuner | 3,124,250 EXP | 2,240,920 Credit
Strategy +15 -> 143.0 Echo | 3424.0 Tuner | 2,269,400 EXP | 1,838,980 Credit
Strategy +20 -> 143.0 Echo | 4070.8 Tuner | 3,052,250 EXP | 2,336,920 Credit
Strategy +25 -> 143.0 Echo | 5020.0 Tuner | 5,204,900 EXP | 3,469,180 Credit

=== EV for Target 5 Specific Substat ===
Strategy +5  -> 1287.0 Echo | 22884.0 Tuner | 18,624,900 EXP | 13,941,180 Credit
Strategy +10 -> 1287.0 Echo | 21498.0 Tuner | 10,617,450 EXP | 10,342,200 Credit
Strategy +15 -> 1287.0 Echo | 27672.0 Tuner | 14,007,000 EXP | 13,462,020 Credit
Strategy +20 -> 1287.0 Echo | 36114.0 Tuner | 25,700,250 EXP | 20,551,320 Credit
Strategy +25 -> 1287.0 Echo | 45060.0 Tuner | 45,988,500 EXP | 31,222,620 Credit
```

Gambar 9. Hasil EV Echo dengan Markov Chain. Sumber : Dokumentasi Pribadi

E. Penentuan Strategi

Penurunan probabilitas yang drastis pada perhitungan peluang bersyarat di subbab sebelumnya membuktikan bahwa meneruskan pemutakhiran Echo yang terdeteksi di fase awal merupakan sebuah inefisiensi. Untuk merumuskan aturan henti (*stopping rule*) yang definitif, analisis kombinatorial murni harus dikorelasikan dengan ekspektasi nilai (*Expected Value*) dari *resource* di dalam sistem *game*, yang melibatkan akumulasi biaya dan tingkat pengembalian (*refund*).

Berdasarkan pemodelan Rantai Markov (Markov Chain), komputasi menghasilkan sebuah temuan matematis berupa korelasi berbanding terbalik antara kuantitas objek Echo mentah dengan volume material peningkatan (*Tuner*, EXP, dan Credit). Menghentikan *tuning* di awal akan membuang Echo potensial yang mungkin berhasil di slot-slot akhir, sehingga meningkatkan kebutuhan Echo. Sebaliknya, mengeksplorasi Echo lebih dalam akan menekan kebutuhan Echo, namun meningkatkan risiko pembakaran material.

Berdasarkan observasi data empiris dari komputasi EV, evaluasi strategi dibagi menjadi tiga konklusi utama:

1. Strategi Efisiensi Kuantitas Echo

Strategi ini menuntut pemain untuk mengeksplorasi probabilitas hingga 3 slot *substat* (Level +15) sebelum menyerah. Karena pemain memberikan toleransi kegagalan awal pada Echo, potensi keberhasilan akhir dapat diselamatkan. Pada pencarian Target 3 *substat*, strategi ini menekan kebutuhan objek Echo mentah hingga titik minimumnya, yakni rata-rata 28,6 Echo. Namun, eksplorasi mendalam ini memicu pembengkakan biaya relasi rekurens hingga menyentuh 848 *Tuner* dan 817.540 EXP.

2. Strategi Efisiensi Material

Strategi ini menginstruksikan pemain untuk segera melebur Echo jika target *substat* tidak muncul pada percobaan pertama (Level +5). Pendekatan ini merupakan strategi paling optimal untuk menyelamatkan material. Sebagai contoh pada pencarian Target 3 *substat* esensial, strategi ini membatasi kerugian material di angka rata-rata 656,7 *Tuner* dan 539.433 EXP. Sebagai konsekuensi dari pembuangan echo secara agresif, pemain diharuskan melakukan pencarian Echo mentah dalam kuantitas yang lebih masif, yakni rata-rata 47,7 Echo per keberhasilan.

3. Anomali Titik Temu Strategi Level +10

Aturan henti pada percobaan kedua (Level +10) kerap diasumsikan sebagai titik tengah yang seimbang. Namun, analisis EV membuktikan bahwa hal tersebut merupakan miskonsepsi matematis (*fallacy of compromise*) apabila diterapkan pada pencarian atribut standar (Target 2 dan 3 *substat*). Pada Target 3, strategi Level +10 membakar 742,2 *Tuner* (lebih boros dari strategi +5) namun tetap menelan 31,8 Echo (lebih boros dari strategi +15).

Secara analitis, Level +10 baru bertindak sebagai titik temu (sweet spot) yang mutlak ideal hanya pada skenario ekstrem pencarian 5 *substat* sempurna (*God Roll*).

IV. Kesimpulan

Penerapan teori Kombinatorika dan probabilitas bersyarat dalam memodelkan sistem *Tuning* Echo pada permainan Wuthering Waves membuktikan bahwa peluang mendapatkan kombinasi atribut ideal secara acak sangatlah kecil. Melalui pemodelan Rantai Markov (Markov Chain) untuk mengevaluasi ekspektasi nilai (*Expected Value*), ditemukan adanya korelasi berbanding terbalik (trade-off) antara kuantitas objek Echo mentah dengan volume material peningkatan (*Premium Tuner*, Echo *exp*, dan *Shell Credit*). Menghentikan *tuning* di awal akan membuang potensi Echo yang sebenarnya dapat sukses di akhir, sehingga meningkatkan beban kuantitas pencarian Echo mentah. Sebaliknya, mengeksplorasi Echo lebih dalam akan menekan kebutuhan objek Echo mentah hingga titik minimum, namun memicu pembengkakan biaya material secara drastis akibat risiko kegagalan akumulatif.

Berdasarkan temuan tersebut, aturan henti (*stopping rule*) yang paling rasional bagi pemain harus disesuaikan dengan kondisi kelangkaan inventaris secara objektif. Bagi pemain yang mengincar atribut standar (2 hingga 4 *substat* esensial), Strategi *tuning* hingga level +5 secara perhitungan merupakan pilihan paling optimal karena memakan *resource* yang lebih sedikit. Sebaliknya, *tuning* hingga Level +15 merupakan langkah yang dapat diterapkan pada objek Echo berkategori langka seperti echo 4-*Cost* guna memaksimalkan potensi keberhasilan objek bernilai tinggi tersebut. Melalui analisis ini, keputusan untuk berhenti pada Level +10 terbukti menjadi sebuah kekeliruan pada pencarian atribut standar karena menghasilkan pengeluaran yang tidak efisien di sisi *resource* dan jumlah echo, dan secara unik baru bertindak sebagai strategi ideal pada skenario ekstrem mengejar 5 *substat* sempurna (*God Roll*).

V. Lampiran

Berikut terdapat link GitHub Gist yang mencantumkan 2 python program yang saya implementasi dan gunakan untuk mensimulasikan persebaran *substat* echo dengan *Monte Carlo* dan perhitungan *Expected Value (EV)* dengan *Markov Chain.*

https://gist.github.com/StackSaint/7978dfb8b1e624fbb005ce0eded4ec96

Referensi

[1] R. Munir, "Kombinatorika (Bagian 1)," Dokumen Perkuliahan IF1220 Matematika Diskrit, Program Studi Teknik Informatika, Institut Teknologi Bandung, Bandung, 2026.

[2] R. Munir, "Kombinatorika (Bagian 2)," Dokumen Perkuliahan IF1220 Matematika Diskrit, Program Studi Teknik Informatika, Institut Teknologi Bandung, Bandung, 2026.

[3] Wuthering Waves Wiki Contributors, "Echo/*Stat*s," *Wuthering Waves Wiki*. [Online]. Tersedia: https://wutheringwaves.fandom.com/wiki/Echo/*Stat*s. [Diakses: 17 Juni 2026].

[4] Wuthering Waves Wiki Contributors, "Echo Development Materials," *Wuthering Waves Wiki*. [Online]. Tersedia: https://wutheringwaves.fandom.com/wiki/Echo_Development_Materials . [Diakses: 17 Juni 2026].

[5] Wuthering Waves Wiki Contributors, "Echo/Leveling," *Wuthering Waves Wiki*. [Online]. Tersedia: https://wutheringwaves.fandom.com/wiki/Echo/Leveling. [Diakses: 17 Juni 2026].

[6] I. F. Kurnia, I. Yudistira, K. Ahadiyah, dan I. M. Tirta, "MCMC (Markov Chain Monte Carlo)," *Komputasi Statistika*, Laboratorium *Stat*istik FMIPA Universitas Jember. [Online]. Tersedia: https://*stat*slab-rshiny.fmipa.unej.ac.id/RDoc/MCMC/. [Diakses: 18 Juni 2026].

[7] E. P. Simorangkir, "Teori Peluang dan Kombinatorika: Peluang Bersyarat," Makalah Program Studi Pendidikan Matematika, Fakultas Keguruan dan Ilmu Pendidikan, Universitas Kristen Indonesia, Jakarta, 2021. Tersedia: http://repository.uki.ac.id/6158/1/TeoriPeluangdanKombinatorika.pdf. [Diakses: 19 Juni 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, 18 Juni 2026

Sebastio Nugroho,13525123