Optimasi Loadout Combat–Stealth dalam Kingdom Come: Deliverance Menggunakan Pemrograman Dinamis

Wildan Abdurrahman Ghazali - 13524054
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: wildanag.10@gmail.com , 13524054@std.stei.itb.ac.id

***Abstrak*—Pemilihan loadout dalam Kingdom Come: Deliverance merupakan persoalan yang tidak sederhana karena pemain harus menyeimbangkan kebutuhan bertarung, kemampuan menyelinap, kapasitas membawa beban, penggunaan stamina, dan batasan kemampuan karakter. Makalah ini membahas penerapan pemrograman dinamis untuk mengoptimalkan loadout bertipe combat–stealth, yaitu kombinasi perlengkapan yang tetap cukup kuat untuk pertarungan tetapi tidak terlalu mengorbankan aspek penyamaran. Permasalahan dimodelkan sebagai variasi multiple-choice knapsack problem, dengan setiap item memiliki nilai utilitas yang dibangun dari skor combat, skor stealth, penalti berat, dan penalti stamina. Dengan pendekatan ini, proses pemilihan loadout dapat dilakukan secara sistematis melalui tabel pemrograman dinamis, sehingga kombinasi item optimal dapat ditentukan berdasarkan batas berat dan stamina yang telah ditetapkan. Pengujian pada 32 item menghasilkan loadout optimal dengan total utility 373,03, berat 28,60, dan penalti stamina 29,10.**

Kata kunci—pemrograman dinamis; knapsack problem; optimasi loadout; Kingdom Come: Deliverance; combat–stealth

I. Pendahuluan

Kingdom Come: Deliverance merupakan permainan RPG yang menekankan realisme dalam pertarungan, eksplorasi, dan perkembangan karakter. Dalam permainan ini, pemain berperan sebagai Henry, seorang pemuda yang harus meningkatkan kemampuan bertarung dan bertahan hidup melalui latihan, pengalaman, serta penggunaan perlengkapan yang sesuai. Sistem permainan menyediakan berbagai pendekatan dalam menyelesaikan konflik, seperti pertarungan jarak dekat, penggunaan senjata jarak jauh, maupun penyusupan secara diam-diam [1].

Gambar 1. Game Kingdom Come: Deliverance

Salah satu aspek penting yang memengaruhi performa karakter adalah pemilihan loadout, yaitu kombinasi perlengkapan yang digunakan oleh karakter. Loadout dapat terdiri atas senjata, armor kepala, armor badan, pakaian, sarung tangan, celana, sepatu, perisai, dan perlengkapan lain. Setiap item memiliki atribut yang berbeda, seperti nilai serangan, nilai pertahanan, berat, durabilitas, visibilitas, conspicuousness (tingkat kemencolokan), dan noise (tingkat kebisingan). Atribut-atribut tersebut berpengaruh terhadap kemampuan karakter dalam bertarung maupun dalam menghindari deteksi musuh [2].

Permasalahan menjadi lebih sulit ketika pemain menggunakan gaya bermain hybrid combat–stealth. Pada gaya bermain ini, pemain tidak hanya membutuhkan armor dan senjata yang kuat, tetapi juga perlengkapan yang tidak terlalu mencolok dan tidak terlalu bising. Armor berat biasanya memberikan perlindungan tinggi, tetapi dapat meningkatkan noise dan membuat karakter lebih mudah terdeteksi. Sebaliknya, perlengkapan ringan cenderung lebih baik untuk stealth, tetapi umumnya memberikan perlindungan lebih rendah ketika karakter harus bertarung. Oleh karena itu, loadout terbaik tidak dapat ditentukan hanya berdasarkan satu atribut tertinggi, melainkan harus mempertimbangkan beberapa faktor sekaligus.

Gambar 2. *Loadout* dan Inventaris game

Dari sudut pandang komputasi, pemilihan loadout dapat dipandang sebagai masalah optimasi kombinatorial. Pemain harus memilih kombinasi item dari beberapa kategori perlengkapan dengan memperhatikan batas berat, batas stamina, dan kelayakan penggunaan item. Masalah ini memiliki kemiripan dengan knapsack problem, khususnya multiple-choice knapsack problem, karena item dikelompokkan berdasarkan kategori dan solusi harus memilih item dari kelompok-kelompok tersebut [5].

II. Landasan Teori

A. Pemrograman Dinamis

Pemrograman dinamis adalah strategi algoritma yang digunakan untuk menyelesaikan persoalan optimasi dengan cara membagi persoalan menjadi beberapa tahap dan menyimpan hasil perhitungan subpersoalan agar tidak dihitung ulang. Dalam pemrograman dinamis, solusi total dibangun dari keputusan-keputusan pada setiap tahap dengan memanfaatkan prinsip optimalitas [4].

Prinsip optimalitas menyatakan bahwa jika suatu solusi total bersifat optimal, maka bagian solusi pada tahap sebelumnya juga harus optimal terhadap state yang bersangkutan. Dengan demikian, keputusan pada suatu tahap dapat dibuat berdasarkan nilai optimal dari tahap sebelumnya tanpa harus menelusuri ulang seluruh kemungkinan dari awal [4].

Dalam konteks optimasi loadout, setiap tahap dapat merepresentasikan kategori perlengkapan yang sedang diproses. State dapat merepresentasikan total berat dan total penalti stamina yang sudah digunakan. Nilai pada tabel DP menyimpan utilitas maksimum yang dapat diperoleh setelah memproses sejumlah kategori tertentu. Dengan pendekatan ini, algoritma dapat mencoba berbagai kemungkinan kombinasi item secara sistematis, tetapi tetap menghindari penghitungan ulang subpersoalan yang sama.

B. Knapsack Problem

Knapsack problem adalah salah satu persoalan klasik dalam optimasi kombinatorial. Pada bentuk dasarnya, terdapat sejumlah item yang masing-masing memiliki nilai dan berat. Tujuannya adalah memilih item dengan total nilai maksimum tanpa melebihi kapasitas maksimum yang tersedia. Pada 0/1 knapsack, setiap item hanya memiliki dua kemungkinan, yaitu dipilih atau tidak dipilih [6].

Permasalahan loadout dalam Kingdom Come: Deliverance memiliki kemiripan dengan knapsack problem karena pemain harus memilih item dengan nilai utilitas tertentu tanpa melampaui batas berat. Akan tetapi, persoalan loadout tidak sepenuhnya sama dengan 0/1 knapsack biasa karena item memiliki kategori perlengkapan. Pemain biasanya tidak memilih item secara bebas tanpa struktur, melainkan memilih item berdasarkan slot atau bagian tubuh tertentu.

Oleh karena itu, model yang lebih sesuai adalah multiple-choice knapsack problem. Pada variasi ini, item dikelompokkan ke dalam beberapa kelas, dan solusi memilih item dari setiap kelas. Dalam studi ini, kelas tersebut direpresentasikan sebagai kategori perlengkapan, misalnya weapon, head, body plate, body garment, gloves, legs, boots, dan shield [5].

C. Pemodelan State dan Transisi DP

Dalam model pemrograman dinamis, state digunakan untuk merepresentasikan kondisi sementara dari proses optimasi. Pada studi ini, state memuat tiga informasi utama, yaitu kategori perlengkapan yang sudah diproses, berat total yang sudah digunakan, dan penalti stamina total yang sudah digunakan.

Misalkan terdapat himpunan kategori perlengkapan (G_1, G_2, ..., G_R). Setiap kategori (G_i) berisi sejumlah item yang dapat dipilih pada kategori tersebut. Nilai (DP[i][w][t]) menyatakan utilitas maksimum setelah memproses (i) kategori pertama dengan total berat (w) dan total penalti stamina (t).

Relasi transisi yang digunakan adalah:

$$DP[i][w][t] = \max\{DP[i-1][w-weight_j][t-stamina_j] + u_j\}$$

dengan j adalah item yang dipilih dari kategori ke-i, $weight_j$ adalah berat item ke-j, $stamina_j$ adalah penalti stamina item ke-j, dan u_j adalah utilitas item ke-j). State yang tidak mungkin dicapai diberi nilai -∞, sedangkan kondisi awal didefinisikan sebagai:

$$DP[0][0][0] = 0$$

dan

$$DP[0][w][t] = -\infty$$

untuk semua w dan t selain nol. Setelah semua kategori diproses, solusi optimal diperoleh dari nilai maksimum pada DP[R][w][t] yang masih memenuhi batas berat dan batas stamina. Untuk mengetahui item apa saja yang membentuk loadout optimal, algoritma menyimpan informasi keputusan pada setiap state.

D. Kompleksitas Algoritma

Misalkan R adalah jumlah kategori perlengkapan, B adalah jumlah maksimum item dalam satu kategori, W adalah batas berat setelah diskretisasi, dan T adalah batas penalti stamina setelah diskretisasi. Untuk setiap kategori, algoritma memeriksa setiap kemungkinan berat, setiap kemungkinan

penalti stamina, dan setiap item pada kategori tersebut. Dengan demikian, kompleksitas waktu algoritma adalah:

$$O(R \times W \times T \times B)$$

Kompleksitas ruang tanpa optimasi adalah:

$$O(R \times W \times T)$$

karena tabel menyimpan nilai untuk setiap kategori, berat, dan penalti stamina. Jika hanya nilai dari tahap sebelumnya yang dibutuhkan, memori dapat dikurangi dengan teknik rolling array menjadi:

$$O(W \times T)$$

III. Metodologi

A. Pendekatan Penelitian

Penelitian ini menggunakan pendekatan pemodelan dan eksperimen komputasi. Fokus utama penelitian adalah membangun model optimasi loadout combat–stealth dan menyelesaikannya menggunakan pemrograman dinamis. Pengujian dilakukan dengan menjalankan algoritma pada dataset item yang telah disusun, kemudian menghasilkan kombinasi loadout dengan nilai utilitas maksimum berdasarkan model estimasi yang digunakan.

B. Pengumpulan Dataset dan Item Pengujian

Dataset disusun dari data item Kingdom Come: Deliverance yang tersedia pada Kingdom Come: Deliverance Wiki. Wiki tersebut menyediakan berbagai atribut item, seperti nilai serangan atau pertahanan, visibility, conspicuousness, noise, durability, weight, price, dan kategori item [2][3]. Data tersebut digunakan karena mudah diakses, cukup lengkap untuk kebutuhan pemodelan, dan sesuai dengan atribut yang diperlukan dalam optimasi combat–stealth.

Jumlah item aktual yang digunakan dalam studi ini adalah 32 item. Item tersebut dipilih dari beberapa kategori perlengkapan, yaitu weapon, head, body plate, body garment, gloves, legs, boots, dan shield. Pemilihan item dilakukan dengan mempertimbangkan variasi karakteristik, sehingga dataset memuat item yang cenderung mendukung combat, item yang cenderung mendukung stealth, serta item yang berada di antara keduanya. Pemilihan item tidak memperhitungkan batasan minimum kemampuan karakter sebagai syarat penggunaan item.

Secara umum, item dikelompokkan menjadi tiga tipe. Pertama, item stealth ringan, yaitu item dengan noise, visibility, dan conspicuousness rendah. Kedua, item hybrid combat–stealth, yaitu item dengan perlindungan atau damage cukup baik tanpa penalti stealth yang terlalu besar. Ketiga, item combat berat, yaitu item dengan damage atau proteksi tinggi tetapi memiliki berat, noise, atau visibility yang lebih besar.

Contoh item yang digunakan antara lain St. George's sword, Longinus' sword, Stinger, dan Heavy warhammer untuk kategori senjata. Item stealth mencakup Dark Saxon gambeson, Light tarred jacket, Silent shoes, Quiet dark shoes, dan Leather gloves. Item hybrid mencakup Aachen dark brigandine, Dark combat jacket, dan Dark riding boots. Item combat berat mencakup Nurembergian cuirass, Magdeburg cuirass, Milanese brigandine, serta beberapa armor plate lainnya. Beberapa halaman item pada wiki menunjukkan atribut seperti stab defense, slash defense, blunt defense, visibility, conspicuousness, noise, durability, dan weight yang relevan untuk perhitungan skor [3].

Struktur dataset yang digunakan ditunjukkan pada tabel berikut.

Atribut	Keterangan
dataset_id	kode unik item
name	nama item
category	jenis item
slot	posisi perlengkapan
stab	nilai stab
slash	nilai slash
blunt	nilai blunt
visibility	tingkat keterlihatan
conspicuousness	tingkat mencolok
noise	tingkat kebisingan
durability	ketahanan item
weight	berat item
price	harga item
stamina_factor	faktor penalti stamina
stamina_penalty_raw	nilai penalti stamina
source	sumber data

Tabel 1. Struktur atribut dataset

C. Praproses dan Normalisasi Data

Sebelum digunakan dalam algoritma, data item melalui tahap praproses. Tahap ini dilakukan dengan menyeragamkan nama atribut, memeriksa nilai kosong, mengelompokkan item berdasarkan slot perlengkapan, dan mengubah atribut numerik ke skala yang sama. Karena atribut item memiliki skala yang berbeda, nilai atribut dinormalisasi ke rentang 0 sampai 100.

Untuk atribut yang semakin besar semakin baik, seperti damage, defense, dan durability, digunakan rumus:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \times 100$$

Untuk atribut yang semakin kecil semakin baik, seperti visibility, conspicuousness, dan noise dibalik setelah normalisasi dengan rumus:

$$X' = 100 - (\frac{X - X_{min}}{X_{max} - X_{min}} \times 100)$$

D. Perhitungan Skor Combat dan Stealth

Skor combat digunakan untuk merepresentasikan kemampuan item dalam mendukung pertarungan. Karena item dalam dataset terdiri atas senjata dan armor, perhitungan skor combat dibedakan berdasarkan jenis item.

Untuk senjata, skor combat dihitung dari nilai serangan stab, slash, dan blunt. Rumus yang digunakan adalah:

$$C_i = 0,4 \times Stab'_i + 0,4 \times Slash_i + 0,2 \times Blunt'_i$$

Untuk armor, skor combat dihitung dari nilai pertahanan terhadap stab, slash, blunt, serta durability. Rumus yang digunakan adalah:

$$C_i = 0,3 \times StabDef'_i + 0,3 \times SlashDef'_i + 0,25 \times BluntDef'_i + 0,15 \times Durability'_i$$

Skor stealth digunakan untuk merepresentasikan kemampuan item dalam mendukung penyusupan. Atribut yang digunakan adalah noise, visibility, dan conspicuousness. Dalam Kingdom Come: Deliverance, noise menunjukkan seberapa mudah karakter terdengar ketika bergerak, visibility menunjukkan seberapa mudah karakter terlihat, dan conspicuousness menunjukkan seberapa mencolok karakter di lingkungan normal [2]. Rumus skor stealth adalah:

$$S_i = 0,4 \times Noise'_i + 0,3 \times Visibility'_i + 0,3 \times Cons'_i$$

E. Penalti Berat dan Penalti Stamina

Selain skor combat dan stealth, studi ini juga memperhitungkan penalti berat dan penalti stamina. Berat item digunakan karena loadout yang terlalu berat dapat mengurangi mobilitas karakter. Penalti stamina digunakan sebagai pendekatan untuk merepresentasikan pengaruh perlengkapan berat terhadap penggunaan stamina saat bergerak atau bertarung.

Penalti berat dihitung dari berat item yang telah dinormalisasi. Untuk penalti stamina, digunakan rumus estimasi:

$$T_i = F_i \times Weight_i$$

dengan (T_i) adalah penalti stamina item ke-i, $Weight_i$ adalah berat item asli, dan F_i adalah faktor kategori item berikut:

Kategori	**F_i**
Plate armor dan shield	1,30
Senjata melee berat	1,15
Senjata melee ringan dan Armor ringan	1,00
Armor sedang dan pakaian pelindung	1,00
Pakaian ringan, gloves, boots stealth	0,80

Tabel 2. Faktor penalti loadout

F. Fungsi Utilitas Loadout

Setiap item diberi nilai utilitas yang menjadi dasar optimasi pemrograman dinamis. Fungsi utilitas menggabungkan skor combat, skor stealth, penalti berat, dan penalti stamina. Rumus yang digunakan adalah:

$$U_i = \lambda_C \times C_i + \lambda_S \times S_i - \lambda_W \times W_i - \lambda_T \times T_i$$

dengan C_i adalah skor combat, S_i adalah skor stealth, W_i adalah penalti berat, dan T_i adalah penalti stamina. Untuk skenario hybrid combat–stealth, bobot yang digunakan adalah:

$$\lambda_C = 0,5;\ \lambda_S = 0,5;\ \lambda_W = 0,2;\ \lambda_T = 0,2$$

G. Pemodelan Pemrograman Dinamis

Permasalahan optimasi loadout dimodelkan sebagai variasi multiple-choice knapsack problem. Setiap kategori perlengkapan dianggap sebagai satu kelompok pilihan, dan algoritma memilih satu item dari setiap kelompok untuk membentuk loadout. Kategori yang digunakan adalah weapon, head, body plate, body garment, gloves, legs, boots, dan shield. Untuk slot yang bersifat opsional, seperti shield, ditambahkan opsi kosong agar algoritma tidak selalu memaksa penggunaan item pada slot tersebut. Opsi kosong pada kategori shield diberi nilai utilitas 0 karena opsi tersebut merepresentasikan tidak adanya item.

Tujuan optimasi adalah memaksimalkan total utilitas loadout dengan tetap memenuhi batas berat dan batas penalti stamina. State pemrograman dinamis didefinisikan sebagai:

$$DP[i][w][t]$$

yang menyatakan utilitas maksimum setelah memproses (i) kategori pertama dengan total berat (w) dan total penalti stamina (t).

Transisi dilakukan dengan mencoba setiap item (j) pada kategori (G_i):

$$DP[i][w][t] = max\{DP[i-1][w - weight_j][t - stamina_j] + u_j\}$$

Kondisi awal algoritma adalah:

$$DP[0][0][0] = 0$$

dan

$$DP[0][w][t] = -\infty$$

untuk state lain yang tidak dapat dicapai.

Setelah seluruh kategori diproses, solusi terbaik diperoleh dari:

$$max(DP[R][w][t])$$

IV. Hasil dan Pembahasan

A. Hasil dan Penyusunan Dataset

Pengujian dilakukan menggunakan dataset 32 item Kingdom Come: Deliverance yang telah disusun berdasarkan kategori perlengkapan. Dataset ini digunakan sebagai masukan utama untuk algoritma pemrograman dinamis. Setiap item memiliki atribut numerik seperti nilai stab, slash, blunt, visibility, conspicuousness, noise, durability, weight, price, min strength, serta penalti stamina hasil estimasi. Atribut tersebut kemudian digunakan untuk menghitung skor combat, skor stealth, penalti berat, penalti stamina, dan utilitas item.

Dataset dibagi ke dalam delapan kategori perlengkapan. Setiap kategori berisi empat item aktual, sehingga jumlah item yang diuji adalah 32 item. Pembagian item berdasarkan kategori ditunjukkan pada Tabel 3.

Kategori	Jumlah Item
Weapon	4
Head	4
Body Plate	4
Body Garment	4
Gloves	4
Legs	4
Boots	4
Shield	4
Total	32

Tabel 3. Jumlah item setiap kategori

B. Hasil Normalisasi dan Perhitungan Skor

Sebelum digunakan dalam algoritma, setiap atribut numerik dinormalisasi ke rentang 0 sampai 100. Untuk atribut yang semakin besar semakin baik, seperti stab, slash, blunt, dan durability, digunakan normalisasi langsung. Untuk atribut stealth seperti visibility, conspicuousness, dan noise, nilai normalisasi dibalik karena nilai yang lebih kecil lebih baik untuk stealth.

Berbeda dengan atribut stealth, penalti berat dan penalti stamina tidak dibalik pada fungsi utilitas. Keduanya diperlakukan sebagai nilai penalti, sehingga semakin besar nilainya, semakin besar pula pengurang terhadap utilitas item. Dengan demikian, fungsi utilitas yang digunakan tetap konsisten, yaitu memberi nilai positif pada skor combat dan stealth, serta memberi pengurangan pada berat dan stamina.

Contoh hasil perhitungan skor beberapa item ditunjukkan pada Tabel 4.

Item	Kategori	Combat Score	Stealth Score	Penalti Berat Ternormalisasi	Penalti Stamina Ternormalisasi	Utility
St. George's sword	Weapon	74,91	100,00	20,80	23,89	78,52
Stinger	Weapon	65,71	100,00	15,20	17,70	76,28
Heavy warhammer	Weapon	20,00	100,00	44,00	57,52	39,70
Padded coif	Head	4,40	90,77	20,00	17,70	40,04
Arching bascinet	Head	22,43	8,50	44,00	65,49	-6,43
Nurembergian cuirass	Body plate	23,87	14,71	28,00	42,48	5,19
Aachen dark brigandine	Body plate	18,29	52,69	52,00	58,41	13,41
Dark Saxon gambeson	Body garment	10,41	92,55	76,00	67,26	22,83
Dark combat jacket	Body garment	11,73	68,95	44,00	49,56	21,63
Leather gloves	Gloves	3,16	97,16	0,00	0,00	50,16
Tight black hose	Legs	5,77	93,26	4,00	3,54	48,00
Quiet dark shoes	Boots	4,62	90,76	12,00	10,62	43,17
Bouche shield	Shield	100,00	100,00	44,00	65,49	78,10

Tabel 4. Contoh hasil skor perhitungan

Dari Tabel 4, terlihat bahwa item dengan combat score tinggi tidak selalu memiliki utilitas paling tinggi. Sebagai contoh, Nurembergian cuirass memiliki combat score lebih tinggi daripada Aachen dark brigandine. Namun, stealth score Nurembergian cuirass jauh lebih rendah, sehingga utilitas

akhirnya lebih kecil. Sebaliknya, Aachen dark brigandine memiliki combat score lebih rendah, tetapi masih memberikan keseimbangan yang lebih baik antara combat dan stealth.

Hal yang sama juga terlihat pada item stealth. Leather gloves dan Quiet dark shoes memiliki combat score rendah, tetapi stealth score tinggi serta penalti berat dan stamina yang kecil. Akibatnya, kedua item tersebut tetap menghasilkan utilitas tinggi dalam model hybrid. Ini menunjukkan bahwa fungsi utilitas tidak hanya memilih item paling kuat, tetapi memilih item yang memberikan kontribusi paling seimbang terhadap kebutuhan combat–stealth.

C. Parameter Pengujian Pemrograman Dinamis

Pemrograman dinamis dijalankan dengan kategori perlengkapan sebagai tahap perhitungan. State menyimpan total berat dan total penalti stamina yang telah digunakan. Nilai pada tabel DP menyimpan utilitas maksimum yang dapat dicapai setelah memproses sejumlah kategori tertentu.

Parameter pengujian yang digunakan ditunjukkan pada Tabel 5.

Parameter	**Nilai**
Jumlah item aktual	32
Jumlah kategori	8
Jumlah opsi shield	5 (termasuk opsi no shield)
Total kombinasi eksplisit	81.920
Batas berat maksimum (W_{max})	30
Batas penalti stamina maksimum (T_{max})	35

Tabel 5. Parameter pengujian pemrograman dinamis

Nilai (W_{max} = 30) dan (T_{max} = 35) dipilih sebagai batas eksperimental untuk menghasilkan loadout hybrid yang tidak terlalu berat. Nilai tersebut masih memungkinkan algoritma memilih perlengkapan combat yang cukup kuat, tetapi tetap membatasi penggunaan armor berat secara berlebihan. Karena beberapa nilai berat dan stamina berbentuk desimal, nilai tersebut dikalikan dengan faktor 10 agar dapat digunakan sebagai indeks tabel DP.

D. Hasil Optimasi dengan Pemrograman Dinamis

Berdasarkan parameter pengujian pada Tabel 5, algoritma pemrograman dinamis menghasilkan loadout optimal seperti pada Tabel 6.

Kategori	**Item Terpilih**	**Combat Score**	**Stealth Score**	**Berat Asli**	**Penalti Stamina Asli**	**Utility**
Weapon	St. George’s sword	74,91	100,00	3,10	3,10	78,52
Head	Padded coif	4,40	90,77	3,00	2,40	40,04
Body plate	Aachen dark brigandine	18,29	52,69	7,00	7,00	13,41
Body garment	Dark combat jacket	11,73	68,95	6,00	6,00	21,63
Gloves	Leather gloves	3,16	97,16	0,50	0,40	50,16
Legs	Tight black hose	5,77	93,26	1,00	0,80	48,00
Boots	Quiet dark shoes	4,62	90,76	2,00	1,60	43,17
Shield	Bouche shield	100,00	100,00	6,00	7,80	78,10
Total		**222,88**	**693,58**	**28,60**	**29,10**	**373,03**

Tabel 6. Loadout optimal hasil pemrograman dinamis

Hasil pada Tabel 6 menunjukkan bahwa total berat loadout adalah 28,60 dan total penalti stamina adalah 29,10. Kedua nilai tersebut masih berada di bawah batas yang ditentukan, yaitu (W_{max} = 30) dan (T_{max} = 35). Total utility yang diperoleh adalah 373,03. Dengan demikian, kombinasi tersebut merupakan solusi optimal berdasarkan dataset, fungsi utilitas, dan batasan eksperimen yang digunakan.

Pemrograman dinamis tidak memilih item terbaik secara lokal pada setiap kategori secara mutlak. Sebagai contoh, Dark Saxon gambeson memiliki utility individual sedikit lebih tinggi daripada Dark combat jacket, tetapi item tersebut memiliki berat 10,00 dan penalti stamina 8,00. Jika Dark Saxon gambeson digunakan bersama item terbaik lain, total berat loadout akan melewati batas berat maksimum. Oleh karena itu, algoritma memilih Dark combat jacket karena memberikan kombinasi global yang lebih feasible dan tetap memiliki utility cukup tinggi.

E. Pembahasan Loadout Optimal

Loadout optimal pada Tabel 6 memperlihatkan pola pemilihan yang seimbang antara kebutuhan combat dan stealth. Algoritma memilih St. George’s sword sebagai senjata karena item ini memiliki combat score tertinggi di antara senjata yang diuji, yaitu 74,91, serta stealth score maksimum. Meskipun beratnya sedikit lebih besar daripada Stinger, selisih

combat score yang cukup besar membuat St. George's sword tetap memberikan utility tertinggi pada kategori weapon.

Pada kategori head, Padded coif dipilih karena memiliki stealth score tinggi dan penalti yang relatif kecil. Item seperti Bell-shaped kettle hat dan Arching bascinet memiliki combat score lebih baik, tetapi penalti stealth dan stamina yang lebih besar menyebabkan utility totalnya lebih rendah. Hal ini menunjukkan bahwa untuk kategori kepala, model lebih mengutamakan perlengkapan yang tidak terlalu mencolok dan tidak terlalu bising daripada perlengkapan logam yang lebih berat.

Pada kategori body plate, Aachen dark brigandine dipilih sebagai kompromi antara combat dan stealth. Nurembergian cuirass dan Milanese brigandine memiliki combat score yang sedikit lebih tinggi, tetapi stealth score-nya lebih rendah. Aachen dark brigandine memiliki stealth score 52,69, jauh lebih baik daripada Nurembergian cuirass yang hanya 14,71. Walaupun beratnya lebih besar, kontribusi stealth dari Aachen dark brigandine membuat utility akhirnya menjadi lebih baik dalam skenario hybrid.

Pada kategori body garment, algoritma memilih Dark combat jacket. Secara utility individual, Dark Saxon gambeson sebenarnya sedikit lebih tinggi. Namun, Dark Saxon gambeson memiliki berat 10,00, sedangkan Dark combat jacket hanya 6,00. Jika item dengan berat lebih besar tersebut digunakan, kombinasi dengan item lain menjadi lebih sulit memenuhi batas ($W_{max} = 30$). Keputusan ini menunjukkan kelebihan pemrograman dinamis: algoritma tidak hanya melihat nilai item secara individual, tetapi juga mempertimbangkan dampaknya terhadap kombinasi total.

Pada kategori gloves, legs, dan boots, algoritma cenderung memilih item ringan dengan stealth score tinggi. Leather gloves dipilih karena memiliki stealth score 97,16 dengan berat hanya 0,50. Tight black hose dipilih karena memiliki stealth score 93,26 dan penalti sangat rendah. Quiet dark shoes dipilih karena memiliki stealth score sedikit lebih tinggi daripada Silent shoes, sehingga memberikan utility terbaik pada kategori boots. Ketiga item ini membantu mengimbangi penggunaan body plate dan shield yang memiliki penalti lebih besar.

Pada kategori shield, Bouche shield tetap dipilih karena berdasarkan dataset yang digunakan, item ini memiliki combat score dan stealth score maksimum. Selain itu, total berat dan penalti stamina loadout masih berada di bawah batas setelah Bouche shield dimasukkan. Opsi no shield tidak dipilih karena pengurangan berat dan stamina dari tidak menggunakan shield tidak mampu menggantikan utility besar yang diberikan oleh Bouche shield. Namun, hasil ini harus dipahami sesuai model dataset, karena nilai stealth shield pada dataset bernilai sangat baik akibat visibility, conspicuousness, dan noise yang bernilai rendah.

Secara keseluruhan, loadout optimal yang diperoleh bukan loadout combat murni dan bukan loadout stealth murni. Loadout ini memakai senjata dan shield dengan kontribusi combat tinggi, tetapi tetap mempertahankan item stealth ringan pada head, gloves, legs, dan boots. Kombinasi tersebut menunjukkan bahwa model pemrograman dinamis mampu menangani trade-off antaritem melalui batas total berat dan penalti stamina. Solusi optimal yang dihasilkan adalah loadout terbaik terhadap dataset 32 item, rumus estimasi, bobot utilitas, dan batas eksperimen yang digunakan, bukan klaim sebagai loadout terbaik absolut dalam seluruh kondisi permainan Kingdom Come: Deliverance.

V. KESIMPULAN

Berdasarkan hasil studi yang telah dilakukan, permasalahan pemilihan loadout combat–stealth dalam Kingdom Come: Deliverance dapat dimodelkan sebagai variasi multiple-choice knapsack problem. Setiap kategori perlengkapan direpresentasikan sebagai kelompok pilihan, sedangkan setiap item memiliki nilai utilitas yang dihitung dari skor combat, skor stealth, penalti berat, dan penalti stamina. Dengan pemodelan tersebut, pemrograman dinamis dapat digunakan untuk memilih kombinasi item terbaik secara sistematis berdasarkan batas berat dan batas penalti stamina yang telah ditentukan.

Pengujian dilakukan pada dataset 32 item yang terbagi ke dalam delapan kategori perlengkapan, yaitu weapon, head, body plate, body garment, gloves, legs, boots, dan shield. Setelah proses normalisasi dan perhitungan utilitas, algoritma menghasilkan loadout optimal yang terdiri atas St. George's sword, Padded coif, Aachen dark brigandine, Dark combat jacket, Leather gloves, Tight black hose, Quiet dark shoes, dan Bouche shield. Loadout tersebut memiliki total utility sebesar 373,03, total berat sebesar 28,60, dan total penalti stamina sebesar 29,10, sehingga masih memenuhi batas eksperimen yang digunakan, yaitu berat maksimum 30 dan penalti stamina maksimum 35.

Hasil optimasi menunjukkan bahwa pemrograman dinamis tidak hanya memilih item dengan nilai combat tertinggi atau stealth tertinggi secara lokal, tetapi mempertimbangkan pengaruh setiap item terhadap kombinasi loadout secara keseluruhan. Beberapa item dengan utility individual tinggi tidak selalu dipilih karena dapat menyebabkan total berat atau penalti stamina melebihi batas. Dengan demikian, metode ini mampu menangani trade-off antara kebutuhan combat dan stealth dalam pemilihan loadout. Namun, hasil yang diperoleh tetap berlaku dalam ruang lingkup dataset, rumus estimasi, bobot utilitas, dan batas eksperimen yang digunakan, sehingga tidak dapat diklaim sebagai loadout terbaik absolut dalam seluruh kondisi permainan.

VI. SARAN

Studi ini masih dapat dikembangkan dengan memperluas dataset item yang digunakan. Jumlah item pada studi ini dibatasi sebanyak 32 item, sehingga variasi kombinasi loadout yang diuji masih terbatas. Studi selanjutnya dapat menambahkan lebih banyak item dari kategori senjata, armor, pakaian, dan perlengkapan lain agar hasil optimasi lebih representatif terhadap pilihan perlengkapan yang tersedia dalam Kingdom Come: Deliverance. Selain itu, model juga dapat dikembangkan dengan memasukkan syarat penggunaan item, seperti minimum strength atau kemampuan karakter tertentu, agar pemilihan loadout menjadi lebih mendekati kondisi aktual permainan.

Pengembangan berikutnya juga dapat dilakukan dengan menguji beberapa skenario bobot dan batas eksperimen yang berbeda. Misalnya, skenario combat dapat menggunakan bobot combat yang lebih besar, sedangkan skenario stealth dapat menggunakan bobot stealth yang lebih dominan. Selain itu, batas berat dan penalti stamina dapat divariasikan untuk melihat perubahan loadout optimal pada kondisi yang lebih ketat atau lebih longgar. Dengan pengujian tambahan tersebut, studi tidak hanya menghasilkan satu loadout optimal, tetapi juga dapat menunjukkan bagaimana perubahan prioritas permainan memengaruhi keputusan pemilihan item.

VII. Ucapan Terima Kasih

Dalam penyelesaian tugas makalah mata kuliah Strategi Algoritma ini, penulis berterima kasih kepada Tuhan Yang Maha Esa karena berkat rahmat dan kasih karunia-Nya penulis dapat menyelesaikan makalah ini dengan baik dan tanpa kendala. Penulis juga ingin mengucapkan terima kasih kepada keluarga serta teman-teman yang telah memberikan dukungan sehingga penulis dapat menyelesaikan makalah ini.

Terima kasih juga penulis sampaikan kepada para pengajar mata kuliah IF2211, khususnya Ir. Rila Mandala, M.Eng., Ph.D. selaku dosen pengampu kelas 02 yang telah memberikan ilmu yang bermanfaat dalam penyusunan makalah ini. Harapan penulis, makalah ini dapat menjadi sumber referensi yang berguna, baik bagi para pembelajar yang tertarik dengan bidang ilmu ini maupun sebagai acuan bagi penulis sendiri di masa mendatang.

Lampiran

Tautan file dataset item: https://drive.google.com/file/d/1aybz11GVuqEwqgFra21L3SqDXnNtFR4V/view?usp=sharing

Daftar Pustaka

[1] Deep Silver, “Kingdom Come: Deliverance”, Deep Silver Official Website. [Online]. Tersedia: https://www.deepsilver.com/games/kingdom-come-deliverance. [Diakses: 17 Juni 2026].

[2] Kingdom Come: Deliverance Wiki, “Stats,” Fandom. [Online]. Tersedia: https://kingdom-come-deliverance.fandom.com/wiki/Stats. [Diakses: 17 Juni 2026].

[3] Kingdom Come: Deliverance Wiki, “Stealth,” Fandom. [Online]. Tersedia: https://kingdom-come-deliverance.fandom.com/wiki/Stealth. [Diakses: 17 Juni 2026].

[4] R. Munir, “Program Dinamis (Dynamic Programming) Bagian 1,” Bahan Kuliah IF2211 Strategi Algoritma, Program Studi Teknik Informatika, STEI ITB, 2026. [Online]. Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/25-Program-Dinamis-(2026)-Bagian1.pdf . [Diakses: 17 Juni 2026].

[5] H. Kellerer, U. Pferschy, dan D. Pisinger, Knapsack Problems. Berlin: Springer, 2004.

[6] S. Martello dan P. Toth, Knapsack Problems: Algorithms and Computer Implementations. Chichester: John Wiley & Sons, 1990.

[7] Kingdom Come: Deliverance Wiki, “Weapons,” Fandom. [Online]. Tersedia: https://kingdom-come-deliverance.fandom.com/wiki/Weapons. [Diakses: 17 Juni 2026].

[8] Kingdom Come: Deliverance Wiki, “Module:ItemData/KCD,” Fandom. [Online]. Tersedia: https://kingdom-come-deliverance.fandom.com/wiki/Module:ItemData/KCD. [Diakses: 17 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, 19 Juni 2026

Wildan Abdurrahman Ghazali
13524054