Penerapan Algoritma Branch and Bound untuk Menentukan Strategi dengan Jumlah Giliran Minimum pada Board Game Splendor

Muhammad Faiz Alfada Dharma - 13524097
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: faizalfadha@gmail.com, 13524097@std.stei.itb.ac.id

Abstract*—Splendor merupakan board game strategi yang melibatkan pengambilan token, pembelian kartu development, perolehan bonus permanen, dan pengumpulan prestige point. Banyaknya kemungkinan urutan pembelian kartu membuat penentuan strategi menjadi persoalan optimasi. Makalah ini membahas penerapan algoritma Branch and Bound untuk menentukan urutan pembelian kartu development dengan estimasi jumlah giliran minimum. Permainan dimodelkan sebagai pohon ruang status, dengan setiap state merepresentasikan urutan kartu yang telah dibeli, total prestige point, bonus permanen, kartu yang tersisa, dan estimasi jumlah giliran. Fungsi lower bound digunakan untuk memperkirakan batas bawah jumlah giliran yang diperlukan, sehingga cabang yang tidak menjanjikan dapat dipangkas. Berdasarkan pengujian pada 12 kartu development yang terbuka, diperoleh strategi C5, C4, C2, C1 dengan total 16 prestige point dan estimasi 16 giliran. Hasil ini menunjukkan bahwa Branch and Bound dapat digunakan untuk menentukan strategi pembelian kartu secara sistematis pada model Splendor yang disederhanakan.

Keywords*—Branch and Bound; Splendor; optimisasi; pohon ruang status; strategi permainan

I. Pendahuluan

Permainan *board game* tidak hanya bergantung pada keberuntungan, tetapi juga pada kemampuan pemain dalam mengambil keputusan yang optimal pada setiap giliran. Setiap pilihan yang dilakukan pemain dapat memengaruhi kondisi permainan pada giliran berikutnya sehingga penentuan strategi terbaik dapat dipandang sebagai suatu permasalahan komputasional. *Board game* dapat dimodelkan sebagai ruang pencarian yang terdiri atas berbagai kemungkinan keadaan dan aksi.

Splendor merupakan salah satu *board game* dimana pemain mengumpulkan token, membeli kartu development, memperoleh bonus permanen, dan mengumpulkan prestige point [5]. Pemain dinyatakan menang apabila berhasil mencapai sejumlah prestige point tertentu. Untuk mencapai kondisi tersebut, pemain perlu menentukan aksi yang tepat, seperti mengambil token, membeli kartu, atau melakukan reservasi kartu. Banyaknya kombinasi aksi yang mungkin menyebabkan pencarian strategi terbaik menjadi tidak sederhana.

Salah satu pendekatan yang dapat digunakan untuk menyelesaikan permasalahan optimasi strategi tersebut adalah Algoritma Branch and Bound [1]. Algoritma ini bekerja dengan membangun pohon pencarian dari berbagai kemungkinan keputusan dan kemudian memangkas cabang yang tidak menjanjikan berdasarkan suatu fungsi batas. Dengan cara ini, algoritma tidak perlu mengeksplorasi seluruh kemungkinan aksi, tetapi tetap berusaha untuk bisa menemukan solusi optimal berdasarkan kriteria yang telah ditentukan.

Dalam makalah ini, permainan Splendor digunakan sebagai studi kasus penerapan algoritma Branch and Bound. Fokus utama makalah ini adalah memodelkan permainan Splendor sebagai permasalahan optimasi untuk menentukan strategi yang dapat mencapai kemenangan dengan jumlah giliran minimum. Setiap keadaan permainan direpresentasikan berdasarkan urutan kartu yang telah dibeli, total prestige point, bonus permanen yang dimiliki, kartu yang belum dibeli, dan estimasi jumlah giliran yang telah digunakan. Dari representasi tersebut, Algoritma Branch and Bound digunakan untuk mengeksplorasi kemungkinan aksi dan memangkas cabang pencarian yang tidak dapat menghasilkan solusi lebih baik dibandingkan solusi terbaik sementara.

Secara lebih khusus, makalah ini membahas beberapa pertanyaan berikut:

1. Bagaimana kondisi permainan Splendor dapat direpresentasikan sebagai state dalam ruang pencarian?
2. Bagaimana aksi pemain dalam Splendor dapat dimodelkan sebagai cabang dalam pohon pencarian?
3. Bagaimana algoritma Branch and Bound dapat digunakan untuk memangkas strategi yang tidak menjanjikan?
4. Bagaimana strategi dengan jumlah giliran minimum untuk mencapai prestige point yang diinginkan dapat ditentukan menggunakan model tersebut?

II. Landasan Teori

A. Permainan Splendor

Splendor merupakan board game strategi untuk 2 sampai 4 pemain. Dalam permainan ini, pemain mengumpulkan token

permata untuk membeli kartu development. Kartu yang dibeli dapat memberikan bonus permanen dan prestige point. Bonus permanen tersebut mengurangi biaya pembelian kartu berikutnya dan dapat membantu pemain memperoleh *noble tiles* [5].

Fig 1 Gambaran Permainan Splendor

Pada setiap giliran, pemain dapat melakukan satu aksi, seperti mengambil token, membeli kartu development, atau melakukan reservasi kartu. Tujuan utama permainan Splendor adalah memperoleh prestige point sebanyak mungkin. Permainan memasuki akhir ketika salah satu pemain mencapai 15 prestige point. Prestige point dapat diperoleh dari kartu development maupun *noble tiles* [5]. Pemain dengan prestige point tertinggi pada akhir permainan menjadi pemenang

B. Komponen Permainan Splendor

Komponen utama dalam Splendor terdiri atas token permata, token emas, kartu development, dan noble tiles [5].

Fig 2 Komponen Permainan Splendor

- Token permata terdiri atas lima warna, yaitu putih, biru, hijau, merah, dan hitam. Token emas berfungsi sebagai joker yang dapat menggantikan token warna apa pun ketika pemain membeli kartu [5].

- Kartu development memiliki atribut berupa level kartu, biaya pembelian, bonus permanen, dan prestige point [5]. Level kartu menunjukkan tingkat kesulitan pembelian kartu. Kartu level rendah umumnya memiliki biaya lebih murah, sedangkan kartu level tinggi umumnya memiliki biaya lebih mahal dan dapat memberikan prestige point yang lebih besar. Dalam permainan, terdapat tiga level kartu development. Setiap permainan, terdapat empat kartu yang terbuka dari setiap level, sehingga terdapat 12 kartu development yang terlihat oleh pemain.

- Noble tiles adalah ubin bangsawan yang dapat memberikan prestige point tambahan. Seorang pemain dapat memperoleh noble tile apabila jumlah dan warna bonus permanen yang dimilikinya memenuhi syarat pada noble tile tersebut.

C. Aturan Permainan Splendor

Pada setiap giliran, pemain memilih satu aksi dari beberapa aksi yang tersedia. Aksi pertama adalah mengambil tiga token permata dengan warna berbeda. Aksi kedua adalah mengambil dua token permata dengan warna yang sama. Aksi mengambil dua token dengan warna yang sama hanya dapat dilakukan apabila tersedia minimal empat token dari warna tersebut sebelum token diambil [5].

Aksi ketiga adalah melakukan reservasi satu kartu development dan mengambil satu token emas jika tersedia. Kartu yang direservasi dapat dibeli pada giliran berikutnya. Pemain tidak boleh memiliki lebih dari tiga kartu reservasi.

Aksi keempat adalah membeli satu kartu development. Kartu yang dibeli dapat berasal dari kartu yang terbuka pada papan atau dari kartu yang telah direservasi sebelumnya. Untuk membeli kartu, pemain membayar biaya kartu menggunakan token dan dapat mengurangi biaya tersebut menggunakan bonus permanen yang telah dimiliki.

D. Persoalan Optimasi

Persoalan optimasi adalah persoalan yang bertujuan untuk meminimalkan atau memaksimalkan suatu fungsi objektif tanpa melanggar batasan persoalan [1]. Fungsi objektif adalah nilai yang ingin dioptimalkan, sedangkan batasan adalah syarat yang harus dipenuhi oleh solusi.

Persoalan optimasi dapat berbentuk minimasi atau maksimasi. Pada persoalan minimasi, tujuan pencarian adalah memperoleh nilai fungsi objektif sekecil mungkin, sedangkan pada persoalan maksimasi, tujuan pencarian adalah memperoleh nilai fungsi objektif sebesar mungkin.

E. Pohon Ruang Status

Pohon ruang status adalah struktur pohon yang digunakan untuk merepresentasikan proses pencarian solusi. Setiap simpul pada pohon menyatakan suatu status atau keadaan, sedangkan setiap sisi atau cabang menyatakan keputusan yang menghasilkan status baru [1].

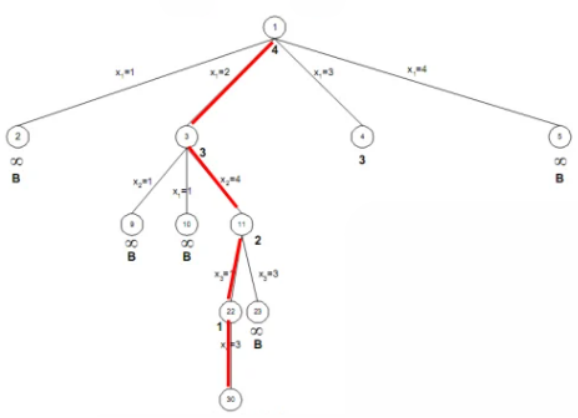

Fig 3 Pembentukan Pohon Ruang Status

Simpul akar menyatakan keadaan awal. Simpul anak menyatakan keadaan yang terbentuk setelah suatu keputusan diambil. Simpul tujuan adalah simpul yang memenuhi kondisi solusi. Pembentukan pohon ruang status dapat dilakukan secara dinamis selama proses pencarian berlangsung.

F. Algoritma Branch and Bound

Branch and Bound adalah algoritma yang digunakan untuk menyelesaikan persoalan optimasi. Algoritma ini membentuk pohon ruang status dan mencari solusi terbaik berdasarkan fungsi objektif yang diberikan [1].

Branch and Bound dapat dipandang sebagai gabungan antara *Breadth First Search* dan *Least-Cost Search*. Pada Breadth First Search biasa, simpul diekspansi berdasarkan urutan pembangkitan. Pada Branch and Bound, simpul yang diekspansi berikutnya dipilih berdasarkan nilai cost terbaik.

Setiap simpul pada Branch and Bound diberi nilai cost. Nilai cost tersebut menyatakan taksiran ongkos terbaik untuk mencapai simpul solusi melalui simpul tersebut. Nilai cost simpul i biasa dinotasikan sebagai $c(i)$ [1].

G. Fungsi Pembatas

Fungsi pembatas atau *bounding function* adalah fungsi yang digunakan untuk menghitung batas nilai terbaik yang masih mungkin dicapai oleh suatu simpul. Nilai *boun*d digunakan untuk menentukan apakah suatu simpul masih layak untuk dieksplorasi atau tidak [1].

Pada banyak persoalan, letak simpul solusi tidak diketahui. Oleh karena itu, nilai cost atau bound dihitung secara heuristik. Nilai *bound* dapat dianggap sebagai batas bawah pada persoalan minimasi atau batas atas pada persoalan maksimasi.

H. Pruning

Pruning adalah proses memangkas atau mematikan simpul yang tidak perlu dieksplorasi lebih lanjut [1]. Pada *Branch and Bound*, *pruning* dilakukan apabila suatu simpul tidak mungkin menghasilkan solusi yang lebih baik daripada solusi terbaik yang telah ditemukan [1]. Sebuah simpul dapat dipangkas jika nilai simpul tersebut tidak lebih baik dari nilai solusi terbaik sejauh ini. Simpul juga dapat dipangkas jika melanggar batasan persoalan atau tidak lagi dapat menghasilkan solusi yang *feasible*.

III. METODOLOGI

A. Batasan Masalah

Penelitian ini menggunakan model permainan Splendor yang disederhanakan. Model hanya mempertimbangkan satu pemain, sehingga aksi lawan tidak dimodelkan. Kartu yang digunakan adalah 12 kartu development yang sedang terbuka pada papan permainan. Kartu tertutup pada deck dan mekanisme penggantian kartu setelah pembelian tidak dimasukkan ke dalam model.

Aksi reservasi kartu dan *noble tiles* juga tidak dipertimbangkan. Dengan demikian, fokus penelitian adalah menentukan urutan pembelian kartu development yang dapat mencapai minimal 15 prestige point dengan estimasi jumlah giliran minimum.

B. Representasi Kartu

Setiap kartu development direpresentasikan berdasarkan atribut yang diperlukan dalam proses pencarian. Atribut tersebut meliputi kode kartu, biaya pembelian, bonus permanen, dan prestige point. Representasi kartu ditulis sebagai berikut :

$$K = (id, cost, bonus, point)$$

Dengan keterangan:

- id = Kode kartu
- $cost$ = Biaya pembelian kartu berdasarkan warna token
- $bonus$ = Warna bonus permanen yang diperoleh setelah kartu dibeli
- $point$ = Prestige point dari kartu

C. Representasi State pada Pohon Ruang Status

Setiap simpul pada pohon ruang status merepresentasikan satu state. State berisi informasi mengenai urutan kartu yang telah dibeli, total prestige point, bonus permanen yang dimiliki, kartu yang belum dibeli, dan estimasi jumlah giliran yang telah digunakan. State direpresentasikan sebagai :

$$S = (path, P, B, R, d)$$

Dengan keterangan:

- $path$ = Urutan kartu yang telah dibeli
- P = Total prestige point
- B = Bonus permanen pemain
- R = Himpunan kartu yang belum dibeli
- d = Estimasi jumlah giliran

State awal adalah kondisi ketika belum ada kartu yang dibeli, prestige point bernilai 0, bonus permanen masih kosong, dan seluruh 12 kartu masih tersedia.

D. Perhitungan Estimasi Giliran

Estimasi jumlah giliran digunakan untuk menghitung biaya dari suatu urutan pembelian kartu. Biaya kartu dikurangi terlebih dahulu oleh bonus permanen yang dimiliki pemain.

$$effective\_cost = max(0, cost - bonus)$$

Setelah itu, total dari *effective_cost* dijumlahkan. Karena pemain dapat mengambil hingga tiga token dalam satu giliran, estimasi giliran untuk mengumpulkan token dihitung sebagai :

$$token\_turn = ciel\left(\frac{\sum effective\_cost}{3}\right)$$

Proses pembelian kartu juga membutuhkan satu giliran tambahan, sehingga estimasi tambahan giliran untuk membeli kartu adalah :

$$additional\_turn = token\_turn + 1$$

Jika suatu simpul memiliki nilai d (estimasi jumlah giliran), maka setelah membeli kartu baru:

$$d' = d + additional\_turn$$

Nilai *additional_turn* ditambahkan ke jumlah giliran pada state sebelumnya.

E. *Fungsi Bound*

Fungsi *bound* digunakan untuk memperkirakan batas bawah jumlah giliran yang masih dibutuhkan untuk mencapai 15 prestige point. Nilai *lower bound* dihitung berdasarkan jumlah minimum kartu tambahan yang masih diperlukan untuk mencapai target point.

Pertama, dihitung sisa point yang dibutuhkan:

$$remaining\_point = 15 - P$$

Kemudian prestige point kartu-kartu yang tersisa diurutkan dari yang terbesar ke yang terkecil. Jika jumlah kartu minimum yang diperlukan untuk memenuhi r*emaining_point* dinyatakan sebagai k, maka :

$$lower\_bound = d + k$$

Jika nilai *lower_bound* lebih besar atau sama dengan solusi terbaik sementara, maka cabang tersebut tidak perlu dieksplorasi lebih lanjut.

F. *Langkah-langkah Algoritma Branch and Bound*

Langkah-langkah algoritma Branch and Bound yang digunakan adalah sebagai berikut:

- Membentuk simpul akar dari state awal.
- Menghitung nilai *lower_bound* dari simpul akar.
- Menghitung nilai *priority* untuk simpul akar.
- Memasukkan simpul akar ke dalam priority queue.
- Mengambil simpul dengan nilai *priority* terkecil.
- Jika terdapat beberapa simpul dengan nilai *priority* yang sama, simpul dengan nilai d lebih kecil diprioritaskan. Jika masih sama, simpul dengan nilai P lebih besar diprioritaskan.
- Jika simpul sudah mencapai minimal 15 prestige point, maka solusi diperbarui.
- Jika belum mencapai target, bangkitkan anak-anak simpul berdasarkan kartu yang masih tersedia.
- Hitung estimasi giliran, *lower_bound*, dan *priority* setiap anak.
- Pangkas atau *pruning* anak yang tidak menjanjikan.
- Ulangi proses sampai priority queue kosong atau tidak ada simpul yang lebih baik dari solusi terbaik sementara.

Nilai *priority* dihitung dengan rumus :

$$priority = lower\_bound - P$$

IV. Hasil dan Eksperimen

A. *Data Pengujian*

Fig 4 Data Pengujian Algoritma Branch and Bound

Pengujian dilakukan menggunakan kondisi papan Splendor pada gambar. Kartu yang digunakan adalah 12 kartu development yang sedang terbuka pada papan permainan. Kartu tersebut terdiri atas 4 kartu level 3, 4 kartu level 2, dan 4 kartu level 1. *Noble tiles* yang berada pada baris paling atas tidak dimasukkan ke dalam pengujian.

Setiap kartu diberi kode C1 sampai C12. Penomoran dilakukan dari kiri ke kanan dan dari atas ke bawah pada bagian kartu development.

TABLE I. Tabel Data Pengujian

Kode	Level	Point	Bonus	W	U	G	R	K
C1	3	4	G	3	6	3	0	0
C2	3	5	W	3	0	0	0	7
C3	3	3	R	3	5	3	0	3
C4	3	4	K	0	0	0	7	0
C5	2	3	U	0	6	0	0	0
C6	2	1	R	2	0	0	2	3

C7	2	1	G	3	0	2	3	0
C8	2	2	G	4	2	0	0	1
C9	1	1	W	0	0	4	0	0
C10	1	0	G	2	1	0	0	0
C11	1	0	K	0	0	2	1	0
C12	1	0	W	0	3	0	0	0

Keterangan Tabel :

- Kolom W, U, G, R, dan K menunjukkan jumlah biaya token permata untuk setiap kartu. W adalah putih, U adalah biru, G adalah hijau, R adalah merah, dan K adalah hitam.
- Kolom bonus adalah bonus permanen yang diberikan oleh setiap kartu

Target pengujian adalah mencapai minimal 15 prestige point dengan estimasi jumlah giliran minimum.

B. Penerapan Representasi Kartu

Setiap kartu direpresentasikan sebagai :

$$K = (id, cost, bonus, point)$$

Sebagai contoh, kartu C5 dapat direpresentasikan sebagai :

$$K = (C5, 6U, U, 3)$$

Representasi kartu C5 dapat dibaca sebagai kartu yang membutuhkan 6 token biru untuk dibeli, memberikan 1 bonus biru (U) setelah dibeli, dan memberikan 3 prestige point.

C. Penerapan Representasi State Awal

State pada setiap simpul direpresentasikan sebagai :

$$S = (path, P, B, R, d)$$

Sebagai contoh untuk state awal dapat direpresentasikan sebagai :

$S0 = ([\,], 0, (0,0,0,0,0), \{C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11, C12\}, 0)$

Artinya, pada state awal pemain belum membeli kartu apa pun, prestige point masih 0, belum memiliki bonus permanen, seluruh 12 kartu masih tersedia, dan estimasi jumlah giliran masih 0.

Selanjutnya adalah menghitung nilai *lower_bound* untuk simpul akar. Berdasarkan representasi state awal dengan target prestige point adalah 15, dapat dihitung *remaining_point.*

$$remaining\_point = 15 - P = 15 - 0 = 15$$

Untuk menghitung k, kartu-kartu yang tersisa diurutkan berdasarkan prestige point terbesar. Kartu dengan poin terbesar adalah *C2* = 5 point, *C1* = 4 point, *C4* = 4 point, dan *C3* = 3 point. Jika hanya menggunakan tiga kartu terbesar, maka poin maksimum yang bisa didapat hanya 13, kurang dari 15. Maka dari itu tiga kartu belum cukup untuk mencapai target. Jika ditambahkan satu kartu berikutnya, yaitu kartu bernilai 3 point, maka total poin yang didapatkan adalah 16. Karena 16 sudah mencapai target, maka jumlah minimum kartu tambahan yang diperlukan atau k adalah 4. Sehingga diperoleh nilai *lower_bound* untuk state awal dan *priority* adalah :

$$lower\_bound = d + k = 0 + 4 = 4$$
$$priority = lower\_bound - P = 4 - 0 = 4$$

D. Ekspansi State Awal

State S0 belum mencapai 15 prestige point. Oleh karena itu, algoritma akan membangkitkan anak-anak state berdasarkan seluruh kartu yang masih tersedia. Karena terdapat 12 kartu, maka terdapat 12 anak state yang dibangkitkan.

TABLE II. TABEL EKSPANSI STATE AWAL

State	*path*	*P*	*B*	*d*	*lower_bound*
S1	[C5]	3	(0,1,0,0,0)	3	6
S2	[C10]	0	(0,0,1,0,0)	2	6
S3	[C11]	0	(0,0,0,0,1)	2	6
S4	[C12]	0	(1,0,0,0,0)	2	6
S5	[C4]	4	(0,0,0,0,1)	4	7
S6	[C8]	2	(0,0,1,0,0)	4	7
S7	[C9]	1	(1,0,0,0,0)	3	7
S8	[C2]	5	(1,0,0,0,0)	5	8
S9	[C1]	4	(0,0,1,0,0)	5	8
S10	[C6]	1	(0,0,0,1,0)	4	8
S11	[C7]	1	(0,0,1,0,0)	4	8
S12	[C3]	3	(0,0,0,1,0)	6	9

Setelah seluruh anak state dibangkitkan, semua state tersebut dimasukkan ke priority queue. State dengan *priority* terkecil diprioritaskan untuk diekspansi.

State S1, S2, S3, dan S4 memiliki *lower_bound* yang sama, yaitu 6. S1 dengan path [C5] dipilih untuk diekspansi terlebih dahulu karena memiliki nilai *priority* lebih kecil. Berikut adalah contoh perhitungan untuk state S1 yang dibentuk dengan membeli kartu *C5* :

Kartu C5 memiliki nilai $K = (C5, 6U, U, 3)$. Pada state awal, pemain belum memiliki bonus permanen. Maka bonus biru pemain adalah 0.

- Biaya efektif *C5* :

$$effective\_cost = max(0, cost - bonus)$$
$$effective\_cost = max(0, 6 - 0)$$
$$effective\_cost = 6$$

- Estimasi tambahan giliran untuk mengambil token :

$$token\_turn = ceil\left(\frac{\sum effective\_cost}{3}\right)$$
$$token\_turn = ceil\left(\frac{6}{3}\right) = 2$$
$$additional\_turn = token\_turn + 1$$
$$additional\_turn = 2 + 1 = 3$$

- Nilai *d* baru :

$$d' = d + additional\_turn$$
$$d' = 0 + 3 = 3$$

- Nilai *P* dan *B* baru :

$$P = 0 + 3 = 3$$
$$B = (0, 1, 0, 0, 0)$$

- Kartu yang belum dibeli :

$$R = \{C1, C2, C3, C4, C6, C7, C8, C9, C10, C11, C12\}$$

Berdasarkan perhitungan untuk State S1, diperoleh state S1 adalah :

$$S1 = ([C5], 3, (0, 1, 0, 0, 0), \{C1, C2, C3, C4, C6, C7, C8, C9, C10, C11, C12\}, 3)$$

Selanjutnya adalah dihitung *lower_bound*. Sisa prestige point yang dibutuhkan :

$$remaining\_point = 15 - P = 15 - 3 = 12$$

Kartu tersisa dengan point terbesar adalah C2, C1, dan C4 dengan total poin adalah 13. Karena tiga kartu tersebut sudah cukup untuk mencapai *remaining_point*, maka kartu tambahan yang diperlukan atau *k* adalah 3. Sehingga diperoleh nilai *lower_bound* dan *priority* untuk state S1 adalah :

$$lower\_bound = d + k = 3 + 3 = 6$$
$$priority = lower\_bound - P = 6 - 3 = 3$$

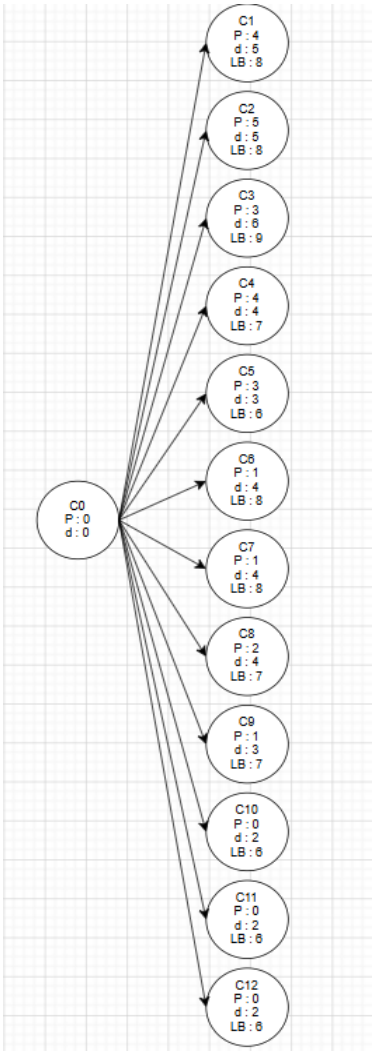

Fig 5 Ekspansi State Awal

E. Ekspansi State S1

State S1 belum mencapai 15 prestige point, sehingga anak-anak state dibangkitkan dari seluruh kartu yang masih tersedia dalam *R,* himpunan kartu yang belum dibeli.

TABLE III. TABEL EKSPANSI STATE S1

State	*path*	*P*	*B*	*d*	*lower_bound*
S13	[C5, C10]	3	(0,1,1,0,0)	5	8
S14	[C5, C11]	3	(0,1,0,0,1)	5	8
S15	[C5, C12]	3	(1,1,0,0,0)	5	8
S16	[C5, C4]	7	(0,1,0,0,1)	7	9
S17	[C5, C8]	5	(0,1,1,0,0)	6	9
S18	[C5, C9]	4	(1,1,0,0,0)	6	9
S19	[C5, C2]	8	(1,1,0,0,0)	8	10

S20	[C5, C1]	7	(0,1,1,0,0)	8	10
S21	[C5, C6]	4	(0,1,0,1,0)	7	10
S22	[C5, C7]	4	(0,1,1,0,0)	7	10
S23	[C5, C3]	6	(0,1,0,1,0)	9	11

State S16 dan S19 memiliki nilai *priority* yang sama, yaitu 2. Karena nilai *priority* sama, digunakan aturan tambahan yaitu memilih state dengan nilai *d* lebih kecil. State S16 memiliki *d* = 7, sedangkan S19 memiliki *d* = 8, sehingga S16 dipilih untuk diekspansi. Berikut adalah contoh perhitungan untuk state S16 yang dibentuk dengan membeli kartu *C4* setelah *C5* :

Kartu *C4* memiliki nilai $K = (C4, 7R, K, 4)$. Pemain memiliki 1 bonus biru. Bonus biru tidak mengurangi biaya C4 karena *C4* membutuhkan token merah. Maka bonus merah pemain adalah 0.

- Biaya efektif *C4* :

$$effective\_cost = max(0, cost - bonus)$$
$$effective\_cost = max(0, 7 - 0)$$
$$effective\_cost = 7$$

- Estimasi tambahan giliran untuk mengambil token :

$$token\_turn = ceil\left(\frac{\sum effective\_cost}{3}\right)$$
$$token\_turn = ceil\left(\frac{7}{3}\right) = 3$$
$$additional\_turn = token\_turn + 1$$
$$additional\_turn = 3 + 1 = 4$$

- Nilai *d* baru :

$$d' = d + additional\_turn$$
$$d' = 3 + 4 = 7$$

- Nilai *P* dan *B* baru :

$$P = 3 + 4 = 7$$
$$B = (0, 1, 0, 0, 1)$$

- Kartu yang belum dibeli :

$$R = \{C1, C2, C3, C6, C7, C8, C9, C10, C11, C12\}$$

Berdasarkan perhitungan untuk State S16, diperoleh state S16 adalah :

$$S16 = ([C5, C4], 7, (0, 1, 0, 0, 1), \{C1, C2, C3, C6, C7, , C8, C9, C10, C11, C12\}, 7)$$

Selanjutnya adalah dihitung *lower_bound*. Sisa prestige point yang dibutuhkan :

$$remaining\_point = 15 - P = 15 - 7 = 8$$

Kartu tersisa dengan point terbesar adalah C2 dan C1 dengan total poin adalah 9. Karena kedua kartu tersebut sudah cukup untuk mencapai *remaining_point*, maka kartu tambahan yang diperlukan atau *k* adalah 2. Sehingga diperoleh nilai *lower_bound* dan *priority* untuk state S16 adalah :

$$lower\_bound = d + k = 7 + 2 = 9$$
$$priority = lower\_bound - P = 9 - 7 = 2$$

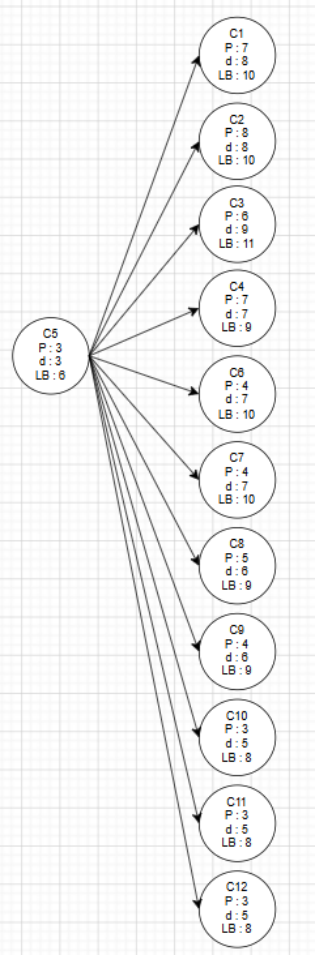

Fig 6 Ekspansi State S1

F. Ekspansi State S16

State S16 belum mencapai 15 prestige point, sehingga anak-anak state dibangkitkan dari seluruh kartu yang masih tersedia dalam *R*.

TABLE IV. TABEL EKSPANSI STATE S16

State	*path*	*P*	*B*	*d*	*lower_bound*
S24	[C5, C4, C10]	7	(0,1,1,0,1)	9	11
S25	[C5, C4, C11]	7	(0,1,0,0,2)	9	11
S26	[C5, C4, C12]	7	(1,1,0,0,1)	9	11
S27	[C5, C4, C2]	12	(1,1,0,0,1)	11	12
S28	[C5, C4, C8]	9	(0,1,1,0,1)	10	12

S29	[C5, C4, C6]	8	(0,1,0,1,1)	10	12
S30	[C5, C4, C9]	8	(1,1,0,0,1)	10	12
S31	[C5, C4, C1]	11	(0,1,1,0,1)	12	13
S32	[C5, C4, C3]	10	(0,1,0,1,1)	12	13
S33	[C5, C4, C7]	8	(0,1,1,0,1)	11	13

Pada jalur solusi, state S27 dengan path [C5, C4, C2] dipilih karena menghasilkan P = 12 dengan nilai *lower_bound* = 12 sehingga nilai *priority* adalah 0. Berikut adalah contoh perhitungan untuk state S27 yang dibentuk dengan membeli kartu *C2* setelah *C5* dan *C4*:

Kartu C2 memiliki nilai $K = (C2,\ 3W + 7K,\ W, 5)$. Pemain memiliki 1 bonus hitam. Maka biaya hitam berkurang 1. Pemain belum memiliki bonus putih.

- Biaya efektif putih (W) :

$$effective\_cost = max\,(0,\ cost - bonus)$$
$$effective\_cost = max\,(0, 3 - 0)$$
$$effective\_cost = 3$$

- Biaya efektif hitam (K) :

$$effective\_cost = max\,(0,\ cost - bonus)$$
$$effective\_cost = max\,(0, 7 - 1)$$
$$effective\_cost = 6$$

- Estimasi tambahan giliran untuk mengambil token :

$$token\_turn = ciel\,(\frac{\sum effective\_cost}{3})$$
$$token\_turn = ciel\,(\frac{6+3}{3}) = 3$$
$$additional\_turn = token\_turn + 1$$
$$additional\_turn = 3 + 1 = 4$$

- Nilai *d* baru :

$$d' = d + additional\_turn$$
$$d' = 7 + 4 = 11$$

- Nilai *P* dan *B* baru :

$$P = 7 + 5 = 12$$
$$B = (1, 1, 0, 0, 1)$$

- Kartu yang belum dibeli :

$$R = \{C1,\ C3,\ C6,\ C7,\ C8,\ C9,\ C10,\ C11,\ C12\}$$

Berdasarkan perhitungan untuk State S27, diperoleh state S27 adalah :

$S27 = (\,[C5, C4, C2],\ 12,\ (1, 1, 0, 0, 1),\ \{C1,\ C3, C6,\ C7,\ , C8,\ C9,\ C10,\ C11,\ C12\},\ 11)$

Selanjutnya adalah dihitung *lower_bound*. Sisa prestige point yang dibutuhkan :

$$remaining\_point = 15 - P = 15 - 12 = 3$$

Kartu tersisa dengan point terbesar adalah C1 dengan total poin adalah 4 atau dengan C3 dengan poin 3. Satu kartu tambahan sudah cukup, sehingga nilai *k* adalah 1 untuk mencapai target. Sehingga diperoleh nilai *lower_bound* dan *priority* untuk state S27 adalah :

$$lower\_bound = d + k = 11 + 1 = 12$$
$$priority = lower\_bound - P = 12 - 12 = 0$$

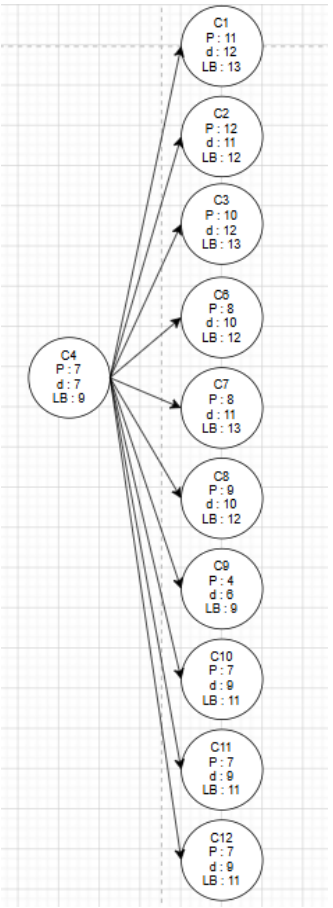

Fig 7 Ekspansi State S16

G. *Ekspansi State S27*

State S27 belum mencapai 15 prestige point, sehingga anak-anak state dibangkitkan dari seluruh kartu yang masih tersedia dalam R.

TABLE V. TABEL EKSPANSI STATE S27

State	*path*	*P*	*B*	*d*	*lower_bound*
S34	[C5, C4, C2, C10]	12	(1,1,1,0,1)	13	14
S35	[C5, C4, C2, C11]	12	(1,1,0,0,2)	13	14
S36	[C5, C4, C2, C12]	12	(2,1,0,0,1)	13	14
S37	[C5, C4, C2, C8]	14	(1,1,1,0,1)	14	15

S38	[C5, C4, C2, C6]	13	(1,1,0,1,1)	14	15
S39	[C5, C4, C2, C9]	13	(2,1,0,0,1)	14	15
S40	[C5, C4, C2, C1]	16	(1,1,1,0,1)	16	16
S41	[C5, C4, C2, C3]	15	(1,1,0,1,1)	16	16
S42	[C5, C4, C2, C7]	13	(1,1,1,0,1)	15	16

Pada ekspansi ini ditemukan dua state solusi, yaitu S40 dan S41. S40 memiliki path [C5, C4, C2, C1] dengan P = 16 dan d = 16. Berikut adalah contoh perhitungan untuk state S40 yang dibentuk dengan membeli kartu *C1* setelah *C5, C4*, dan *C2* :

Kartu C1 memiliki nilai $K = (C1,\ 3W + 6U + 3G,\ G, 4)$. Pemain memiliki 1 bonus putih dan 1 bonus biru. Pemain belum memiliki bonus hijau.

- Biaya efektif putih (W) :

$$effective\_cost = max\,(0,\ cost - bonus)$$
$$effective\_cost = max\,(0, 3 - 1)$$
$$effective\_cost = 2$$

- Biaya efektif Biru (U) :

$$effective\_cost = max\,(0,\ cost - bonus)$$
$$effective\_cost = max\,(0, 6 - 1)$$
$$effective\_cost = 5$$

- Biaya efektif hijau (G) :

$$effective\_cost = max\,(0,\ cost - bonus)$$
$$effective\_cost = max\,(0, 3 - 0)$$
$$effective\_cost = 3$$

- Estimasi tambahan giliran untuk mengambil token :

$$token\_turn = ceil\,(\frac{\sum effective\_cost}{3})$$
$$token\_turn = ceil\,(\frac{2+5+3}{3}) = 4$$
$$additional\_turn = token\_turn + 1$$
$$additional\_turn = 4 + 1 = 5$$

- Nilai *d* baru :

$$d' = d + additional\_turn$$
$$d' = 11 + 5 = 16$$

- Nilai *P* dan *B* baru :

$$P = 12 + 4 = 16$$
$$B = (1, 1, 1, 0, 1)$$

- Kartu yang belum dibeli :

$$R = \{C3,\ C6,\ C7,\ C8,\ C9,\ C10,\ C11,\ C12\}$$

Berdasarkan perhitungan untuk State S40, diperoleh state S40 adalah :

$$S40 = (\,[C5, C4, C2, C1],\ 16,\ (1, 1, 1, 0, 1),\ \{C3, C6,\ C7,\ ,C8,\ C9,\ C10,\ C11,\ C12\},\ 16)$$

Karena State S40 memiliki nilai *priority* yang paling kecil dan nilai P sudah lebih besar atau sama dengan 15, maka S40 merupakan solusi. Solusi terbaik sementara diperbarui dengan urutan path [C5, C4, C2, C1] dalam 16 giliran

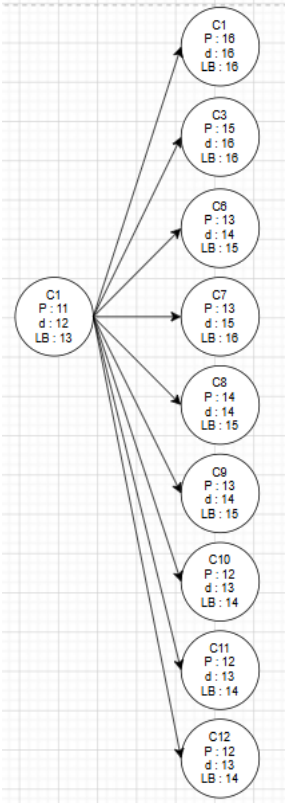

Fig 8 Ekspansi State S27

H. Pruning Simpul

Setelah solusi S40 ditemukan, algoritma melakukan *pruning* terhadap simpul yang tidak dapat menghasilkan solusi lebih baik. Karena jumlah giliran terbaik adalah 16, maka state dengan *lower_bound* lebih besar atau sama dengan 16 tidak perlu dieksplorasi lebih lanjut karena tujuan pencarian adalah menemukan solusi dengan estimasi giliran lebih kecil.

I. Hasil Akhir Algoritma Branch and Bound

Berdasarkan proses Branch and Bound, solusi yang dipilih adalah solusi dengan urutan pembelian kartu C5 → C4 → C2 → C1 dengan total prestige point 16 dalam 16 giliran. Terdapat juga solusi alternatif yaitu dengan urutan pembelian kartu C5 → C4 → C2 → C3 dengan total prestige point 15 dalam 16 giliran. Solusi C5 → C4 → C2 → C1 dipilih karena menghasilkan prestige point lebih besar dengan jumlah giliran yang sama. *Fig 9* menunjukkan hasil akhir dari ekspansi pohon ruang status. Warna biru menunjukkan urutan state yang dipilih, sedangkan warna merah menunjukkan simpul yang tidak menghasilkan solusi, atau yang sudah dilakukan *pruning* / pemangkasan.

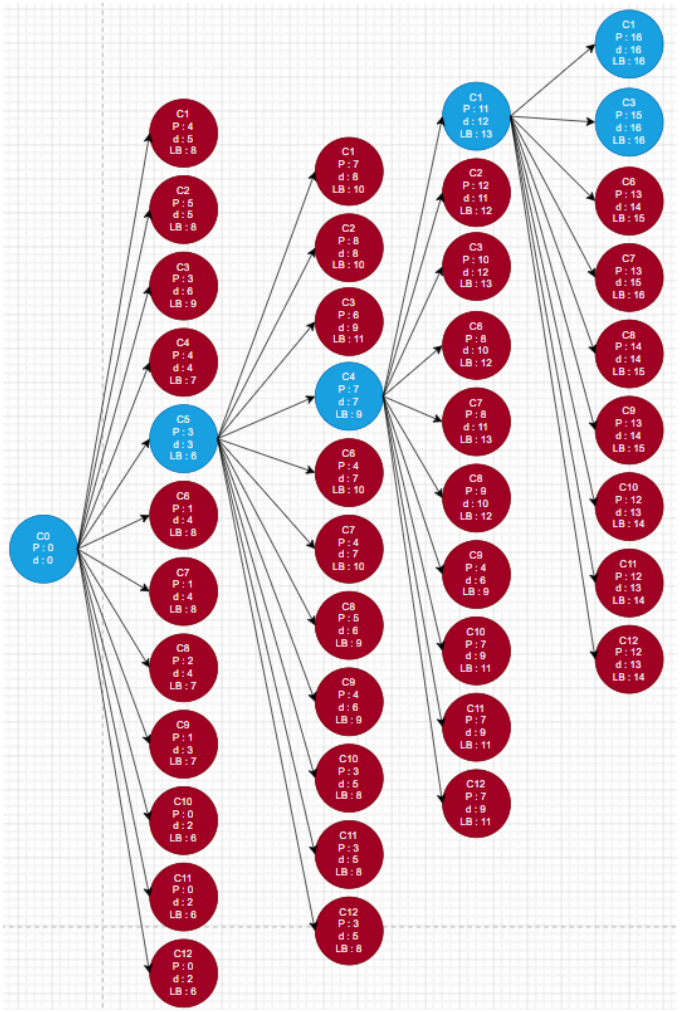

Fig 9 Hasil Akhir Pohon Ruang Status dan Jalur Terbaik

J. *Analisis Hasil Akhir*

Hasil pengujian menunjukkan bahwa algoritma Branch and Bound dapat digunakan untuk mencari urutan pembelian kartu development pada model permainan Splendor yang disederhanakan. Algoritma membentuk pohon ruang status, menghitung nilai *lower_bound*, lalu memilih cabang yang masih berpotensi menghasilkan solusi terbaik.

Solusi yang diperoleh adalah urutan pembelian C5 → C4 → C2 → C1 dengan total 16 prestige point dan estimasi 16 giliran. Terdapat solusi alternatif C5 → C4 → C2 → C3 yang menghasilkan 15 prestige point dengan estimasi giliran yang sama. Solusi pertama dipilih karena memberikan prestige point lebih besar.

Penggunaan *lower_bound* membantu algoritma memangkas cabang yang tidak dapat menghasilkan solusi lebih baik. Dengan demikian, Branch and Bound dapat mengurangi pencarian yang tidak perlu dibandingkan brute force. Namun, hasil ini masih terbatas karena model belum mempertimbangkan aksi lawan, *noble tiles*, reservasi kartu, dan penggantian kartu dari deck.

V. KESIMPULAN

Berdasarkan hasil pengujian, algoritma Branch and Bound dapat diterapkan untuk menentukan urutan pembelian kartu development pada model permainan Splendor yang disederhanakan. Permainan dimodelkan sebagai pohon ruang status, dengan setiap state merepresentasikan urutan kartu yang telah dibeli, total prestige point, bonus permanen, kartu yang tersisa, dan estimasi jumlah giliran.

Hasil pengujian menunjukkan bahwa strategi terbaik yang diperoleh adalah urutan pembelian C5 → C4 → C2 → C1. Urutan tersebut menghasilkan total 16 prestige point dengan estimasi 16 giliran. Penggunaan nilai lower_bound membantu algoritma dalam menilai cabang yang masih menjanjikan dan memangkas cabang yang tidak dapat menghasilkan solusi lebih baik.

Dengan demikian, Branch and Bound dapat digunakan untuk menyelesaikan persoalan optimasi strategi pada Splendor, khususnya dalam menentukan urutan pembelian kartu secara sistematis. Namun, model yang digunakan masih terbatas karena belum mempertimbangkan aksi lawan, reservasi kartu, noble tiles, dan penggantian kartu dari deck.

REFERENSI

[1] Rinaldi, N. U. Maulidevi, and M. L. Khodra, "Algoritma Branch & Bound (Bagian 1)," Bahan Kuliah IF2211 Strategi Algoritma, Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, 2026.

[2] Rinaldi, N. U. Maulidevi, and M. L. Khodra, "Algoritma Branch & Bound (Bagian 2)," Bahan Kuliah IF2211 Strategi Algoritma, Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, 2026.

[3] Rinaldi, N. U. Maulidevi, and M. L. Khodra, "Algoritma Branch & Bound (Bagian 3)," Bahan Kuliah IF2211 Strategi Algoritma, Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, 2026.

[4] R. Munir and M. L. Khoddra, "Algoritma Branch & Bound: Bahan Kuliah IF2211 Strategi Algoritma (Bagian 4)," Bahan Kuliah IF2211 Strategi Algoritma, Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, 2026.

[5] Space Cowboys, "Splendor," Space Cowboys Games. [Online]. Available: Space Cowboys official Splendor page. [diakses: Juni. 18, 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

Muhammad Faiz Alfada Dharma – 13524097