Penerapan Algoritma Branch and Bound sebagai Strategi Permainan Othello

Difa Habiba Rahman (13518098)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13518098@std.stei.itb.ac.id

Permainan Othello merupakan permainan papan berbasis strategi dengan persoalan optimasi yang disertai batasan aturan dalam penentuan skornya. Untuk menyelesaikan persoalan optimasi, algoritma* branch and bound *adalah salah satu algoritma yang dapat digunakan untuk mencari langkah terbaik. Algoritma* branch and bound *membangkitkan dan memeriksa setiap kemungkinan langkah yang dapat diambil, lalu memangkas kemungkinan yang tidak mengarah ke solusi terbaik. Pemangkasan dilakukan dengan memeriksa hasil status dari langkah tersebut dengan batasan yang ada. Jika status melanggar batasan, langkah tersebut dibuang. Langkah yang dapat dipilih kemudian diurutkan sesuai biaya yang didefinisikan dalam perancangan. Dalam persoalan Othello, algoritma* branch and bound *dapat membantu menemukan penyelesaian, namun algoritma ini bukan algoritma paling optimal yang dapat digunakan

Kata Kunci—Branch and bound, optimasi, Othello

I. PENDAHULUAN

Permainan papan merupakan jenis permainan tradisional yang popular sejak abad sebelum masehi. Jenis permainan ini memiliki beragam variasi dan dapat dimainkan oleh kalangan umum tanpa memandang kalangan. Di zaman modern ini pun, masih banyak permainan papan yang sering dimainkan, bahkan dibuat dalam wujud aplikasi dalam program sehingga lebih praktis, mudah dimainkan kapanpun, dan mudah melibatkan orang lain baik yang dekat maupun jauh. Salah satu permainan yang paling terkenal merupakan permainan Othello.

Gambar 1. Permainan Othello
Sumber: https://board-games-galore.fandom.com/wiki/Othello

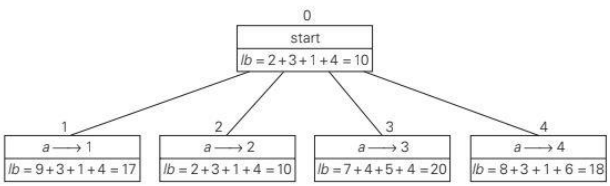
Permainan papan, termasuk Othello, biasanya memiliki tujuan dan aturan dalam memainkannya. Permainan papan seperti Othello juga membutuhkan strategi dalam menyelesaikan dan memenangkannya. Strategi yang digunakan harus dicocokkan dengan jenis dan tujuan dari permainan.

Dalam permainan Othello yang dimainkan oleh dua orang, tiap pemain harus berlomba-lomba mendapatkan keping warna terbanyak dengan cara meletakkan keping berwarnanya setiap giliran. Pemain yang memenangkan permainan ini adalah pemain yang memiliki keping terbanyak di akhir permainan atau berhasil menghabisi warna keping lawan sebelum papan penuh Namun, dalam memilih tempat untuk menyimpan kepingan, pemain juga harus memikirkan tempat paling strategis supaya kepingannya tidak mudah direbut orang lain. Oleh karena itu, strategi untuk mendapatkan poin lebih banyak tidak cukup untuk menentukan langkahnya.

Persoalan dalam permainan Othello dapat digolongkan ke dalam permasalahan optimasi. Dalam ilmu strategi algoritma, terdapat beberapa strategi yang dapat dipilih untuk permsalahan ini. Beberapa strategi di antaranya adalah *brute force, greedy, branch and bound,* dan *A**. Dari strategi-strategi tersebut, algoritma *branch and bound* dapat digunakan untuk memilih langkah dalam menyimpan keping sekaligus memerhatikan tempat strategis pada papan. Algoritma ini dapat dimanfaatkan untuk memprediksi langkah mana yang terbaik untuk diambil dengan prinsip tersebut.

II. TEORI DASAR

A. *Algoritma Branch and Bound*



Gambar 2. Contoh Penerapan *Branch and Bound*
Sumber: *Introduction to The Design & Analysis of Algorithms, 3rd Edition,* 2012

Algoritma *branch and bound* adalah algoritma yang digunakan untuk persoalan optimisasi [6]. Algoritma *branch and bound* merupakan gabungan dari prinsip pencarian dengan BFS dan *least cost search* (atau *greatest cost search* jika yang dibutuhkan merupakan hasil maksimal). Dengan algoritma *branch and bound*, dilakukan minimalisasi atau maksimalisasi dari suatu fungsi objektif dengan hasil yang tidak melanggar batasan persoalan. Jika sebuah kemungkinan langkah yang dapat diambil ternyata tidak mengarah ke solusi, langkah tersebut disingkirkan dari kemungkinan solusi dan tidak diperiksa lebih lanjut.

Untuk menyatakan langkah yang akan diambil, tiap kemungkinan langkah yang dapat diambil dengan algoritma *branch and bound* ini memiliki biaya (*cost*) berdasarkan persoalannya. Nilai biaya tersebut biasanya merupakan nilai taksiran. Biaya tersebut secara umum digambarkan dalam persamaan

$$\hat{c}(i) = \hat{g}(i) + \hat{h}(i) \qquad (1)$$

dengan $\hat{c}(i)$ merupakan ongkos untuk simpul i, $\hat{g}(i)$ merupakan ongkos mencapai simpul i dari akar, dan $\hat{h}(i)$ merupakan ongkos untuk mencapai simpul tujuan dari simpul i.

Persoalan beserta status-status yang dapat diambil ketika menjalankan algoritma *branch and bound* biasanya diatur dalam bentuk pohon. Pohon tersebut disebut sebagai ruang status. Tiap simpul yang ada pada pohon tersebut menggambarkan sebuah status. Sebuah status memperlihatkan keadaan persoalan yang sedang diselesaikan setelah melalui langkah tertentu. Tiap cabang menandakan transisi dari sebuat status ke status lain setelah mengambil sebuah langkah. Tiap langkah memiliki nilai biaya. Dalam memeriksa sebuah simpul, dibangkitkan anak-anak dari simpul tersebut. Pemeriksaan kemudian dilanjutkan pada anak simpul, dimulai dari yang memiliki biaya terkecil (atau terbesar).

Secara umum, berikut adalah langkah-langkah melakukan algoritma *branch and bound.*

1) Memasukkan simpul akar ke antrian pemeriksaan.

2) Mengambil simpul pertama dalam antrian pemeriksaan.

3) Jika simpul tersebut adalah simpul solusi (*goal*), pencarian diberhentikan dan simpul ditetapkan sebagai solusi. Jika bukan, pencarian dilanjutkan.

4) Jika simpul memiliki anak yang dapat dibangkitkan, pencarian dilakjutkan ke langkah 6). Jika tidak, antrian diperiksa di langkah 5).

5) Jika antrian tidak kosong, pencarian dilanjutkan dengan melakukan langkah 2). Jika antrian kosong, pencarian dilakukan dan dinyatakan bahwa tidak ada solusi untuk permasalahan tersebut.

6) Membangkitkan anak-anak dari simpul yang sedang diperiksa.

7) Menghitung nilai biaya dari setiap anak simpul tersebut.

8) Memasukkan tiap anak simpul ke dalam antrian mengurut berdasarkan biayanya. Jika terdapat simpul yang tidak memungkinkan untuk dijadikan solusi atau tidak mengarah ke solusi berdasarkan nilai biayanya, simpul tersebut tidak akan diperiksa lagi.

9) Pencarian dilanjutkan ke langkah 2).

B. Penghitungan Heuristik

Untuk melakukan algoritma *branch and bound*, perlu dicari nilai biaya dari tiap simpul yang dibangkitkan dalam pohon ruang status. Penghitungan biaya dapat ditentukan sesuai dengan permasalahan yang diselesaikan. Untuk mengoptimalkan penyelesaian dapat dimasukkan perhitungan heuristik.

Perhitungan secara heuristik adalah perhitungan yang dirancang untuk menyelesaikan persoalan dengan lebih cepat dan lebih efisien dari metode biasa [4]. Algoritma heuristik sering digunakan untuk menyelesaikan persoalan NP atau persoalan yang membutuhkan keputusan. Perhitungan heuristic dapat memberikan solusi secara terpisah atau menjadi dasar perhitungan dalam algoritma untuk optimasi.

III. PERMAINAN OTHELLO

A. Sejarah Permainan Othello

Permainan Othello, dikenal juga dengan permainan Reversi, adalah sebuah permainan papan klasik berukuran 8x8 petak. Terdapat beberapa sumber berbeda yang menerangkan sejarah dari permainan ini. Menurut sebuah versi, [1] ada dua tempat yang disebut sebagai asal-muasal dari permainan Othello. Tempat pertama adalah Tionghoa. Othello di Tionghoa dikenal dengan nama Fan Mian. Namun, lebih banyak sumber menyebutkan bahwa Othello diciptakan dan dipatenkan oleh orang Inggris bernama Lewis Waterman dan John W. Mollet pada tahun 1888[2]. Permainan ini mulai mendunia hingga Perang Dunia I. Saat itu, Othello masih dikenal dengan nama Reversi. Setelah Perang Dunia I, permainan tersebut tidak lagi popular.

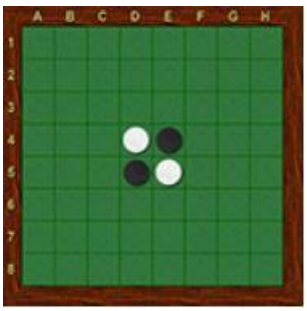
Pada tahun 1968, permainan Reversi dibuat ulang di Jepang dan diberi nama Othello. Pada tahun 1973, permainan tersebut secara resmi dirilis [5]. Konsep dan peraturan dari Othello ini dirancang oleh Goro Hasegawa yang berasal dari Jepang.

B. Peraturan dan Tata Cara Bermain Othello

Othello adalah permainan dua orang yang dimainkan menggunakan sebuah papan berukuran 8x8 petak dan 64 buah kepingan dengan warna berbeda di sisi-sisinya, yaitu hitam dan putih. Keping-keping tersebut dapat diletakkan di atas papan dan dapat dibalik warnanya. Tujuan dari permainan ini adalah untuk mendapatkan sebanyak-banyaknya keping dengan warna yang sama dengan warna yang dimiliki oleh pemain.

Sebelum permainan dimulai, kedua pemain terlebih dahulu menentukan warna. Cara penentuan warna ini dilakukan sesuai kesepakatan pemain. Kemudian, papan disiapkan dengan empat keping pertama disimpan di empat petak

ditengah papan, sementara 60 keping lainnya dibagi sama banyak kepada masing-masing pemain.



Gambar 3. Pengaturan Awal Permainan Othello
Sumber: https://www.ultraboardgames.com/othello/game-rules.php

Pemain dengan warna putih mengambil giliran pertama. Dalam satu giliran pemain, pemain harus menaruh satu keping di atas papan. Keping hanya boleh ditaruh di tempat sedemikian rupa sehingga keping yang baru ditaruh dapat 'menjepit' keping lawan dengan keping lain dengan warna pemain. Pemain dapat 'menjepit' keping pemain lawan secara vertikal, diagonal, atau horizontal. Pemain dapat menjepit keping lawan dari lebih dari satu arah dalam satu waktu. Keping lawan yang 'terjepit' kemudian dibalik menjadi warna milik pemain. Jika tidak ada tempat untuk pemain meletakkan kepingnya sesuai aturan, maka gilirannya dilewati.

Gambar 4. Contoh Langkah dalam Othello
Sumber: https://www.ultraboardgames.com/othello/game-rules.php

Permainan berakhir ketika salah satu dari kondisi berikut terpenuhi.

1) Papan sudah penuh karena seluruh keping telah diletakkan.

2) Seluruh keping di papan memiliki warna yang sama sehingga tidak ada pemain yang dapat menjepit keping pemain lainnya.

IV. Implementasi Branch and Bound dalam Othello

Algoritma *branck and bound* dipilih untuk menyelesaikan persoalan Othello ini. Hel tersebut disebabkan *branch and bound* merupakan salah satu algoritma yang digunakan untuk menyelesaikan persoalan optimasi. Dengan algoritma *branch and bound*, dapat ditentukan biaya tiap status permainan dengan memperhatikan poin kepingan yang dapat diperoleh dalam satu langkah serta posisi tempat kepingan dapat ditaruh. Nilai perhitungan biaya tersebut dapat membantu dalam menentukan solusi dalam permainan. Selain itu, algoritma ini juga dipilih disebabkan tiap giliran pemain, pasti diperlukan *bounding* atau pemangkasan simpul yang takkan dikunjungi karena tidak mengarah ke solusi maupun nilai terbaik untuk pemain. Algoritma *branch and bound* dapat melakukan pemangkasan simpul dengan memerhatikan biaya simpul dan batasan atau *constraints* yang berlaku dalam permainan.

Dalam menyelesaikan persoalan Othello dengan *branch and bound*, status yang diwakili oleh setiap simpulnya merupakan keadaan papan Othello setiap giliran sebuah pemain. Dimpul akar dari persoalan ini merupakan papan dengan empat buah keping yang disimpan di awal permainan sebelum dimulai. Kemudian, tiap cabang akan mewakili keputusan atau langkah yang dapat diambil oleh seorang pemain dalam satu gilirannya.

Terdapat batasan yang harus diperhatikan sebelum membangkitkan langkah yang dapat diambil oleh pemain. Pertama, langkah yang diambil harus dipastikan berada di atas papan. Kedua, langkah yang diambil dapat mengubah minimal satu keping lawan di atas papan menjadi warna milik pemain. Konsekuensi dari batasan kedua berarti keping tidak dapat disimpan jauh dari keping-keping yang sudah ada di atas papan pada giliran sebelumnya. Jika kedua batasan ini tidak dipenuhi sebuah kemungkinan langkah, maka langkah tersebut tidak dibangkitkan di dalam pohon status.

Setelah membangkitkan sebuah simpul, perlu dihitung biaya dari simpul tersebut. Karena tujuan dari permainan ini adalah mendapatkan sebanyak-banyaknya poin, maka lebih mudah untuk mengambil sebuah simpul dengan biaya terbesar dari nilai fungsi biaya yang perlu ditetapkan. Nilai biaya sebuah simpul dipengaruhi oleh banyaknya keping musuh yang dapat diubah dalam satu langkah serta posisi tempat yang akan disimpan keping. Semakin banyak keping musuh yang dapat diubah menjadi warna sendiri, semakin baik. Semakin strategis tempat yang dipilih, semakin sulit lawan untuk merebut kembali keping tersebut.

Karena tempat mempengaruhi nilai biaya, maka tempat paling strategis harus memiliki prioritas tertinggi dari langkah lain yang tidak berada dalam tempat strategis. Tempat strategis yang dimaksud di sini merupakan petak-petak pada papan yang berada di pinggir sehingga sulit untuk keping yang ada dalam petak tersebut untuk dijepit oleh keping lawan. Namun, petak-petak sekitarnya memiliki kelemahan karena dapat memberikan lawan kesempatan untuk merebut petak tersebut sehingga nanti pemain akan kesulitan untuk mengambil alih kembali.

Berdasarkan deskripsi tersebut, berikut adalah salah satu pemberian poin tempat pada petah-petak yang berada di atas papan. Petak-petak yang berada di ujung diberi nilai lebih karena keping yang disimpan di petak tersebut sulit untuk direbut lawan. Sementara itu, petak yang mengelilingi ujung diberi nilai lebih kecil karena petak-petak tersebut harus dihindari. Dengan menghindari petak-petak tersebut, lawan

akan kesulitan mendapatkan kesempatan untuk menyimpan keping di ujung-ujung papan. Gambar 5 merupakan contoh aplikasi tersebut.

15	0	10	10	10	10	0	15
0	0	5	5	5	5	0	0
10	5	5	5	5	5	5	10
10	5	5	5	5	5	5	10
10	5	5	5	5	5	5	10
10	5	5	5	5	5	5	10
0	0	5	5	5	5	0	0
15	0	10	10	10	10	0	15

Gambar 5. Contoh Pemberian Poin pada Petak Othello
Sumber: Dokumentasi Pribadi

Setelah menentukan poin tersebut, fungsi biaya sebuah simpul dapat didefinisikan. Fungsi biaya memiliki bentuk seperti persamaan (1), dengan ketentuan sebagai berikut.

1) Nilai $\hat{h}(i)$ didapatkan dari jumlah keping musuh yang dapat diperoleh jika mengambil langkah tersebut ditambah poin tempat yang telah ditentukan, seperti pada gambar 5.

2) Nilai $\hat{g}(i)$ didapatkan dari poin yang sudah didapatkan pemain pada giliran tersebut ditambah giliran dalam permainan.

Strategi ini merupakan strategi yang dapat diterapkan oleh seorang pemain dan dalam status yang dapat berubah-ubah. Seperti misalnya, pada giliran musuh, poin pemain bisa berkurang karena musuh menyimpan kepingnya dan menjepit keping milik pemain. Selain itu, strategi ini tidak mempengaruhi keputusan lawan dalam mengambil langkah. Oleh karena itu, perlu dilakukan penyesuaian sebagai berikut.

1) Di akhir setiap giliran, perlu dilakukan *update* terhadap angka giliran yang sedang berjalan dan nilai biaya dari simpul pada giliran berikutnya.

2) Simpul yang tidak diambil pada sebuah giliran langsung dipangkas saat itu juga.

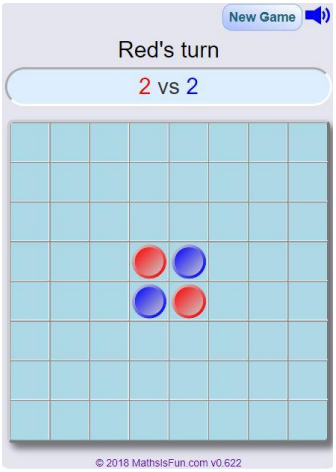
3) Pada giliran lawan, semua simpul yang dapat dibangkitkan dalam pohon memiliki nilai biaya yang sama sehingga simpul dapat diambil secara acak dan tidak dipandang berbeda. Langkah yang diambil merupakan langkah yang dipilih oleh lawan.

V. Hasil Percobaan

A. *Teknis Percobaan*

Untuk menguji kinerja dari rancangan algoritma ini, digunakan alat berupa permainan Othello atau Reversi yang tersedia daring, salah satunya pada [7]. Pada permainan tersebut, digunakan warna merah dan biru sebagai pengganti warna ikonik Othello yaitu putih dan merah. Pemain putih pada permainan memegang keping merah dan melakukan giliran pertama.

Permainan daring digunakan untuk mensimulasikan permainan yang dapat terjadi dan mendapatkan sampel dari reaksi mesin yang menjadi lawan dalam bertanding untuk dimasukkan ke perhitungan dalam algoritma. Mesin lawan menggunakan algoritma tertentu yang juga mempertimbangkan langkah sesuai dengan masukan dari lawannya.



Gambar 6. Permainan Othello Daring
Sumber: https://www.mathsisfun.com/games/reversi.html

Dalam mengimplementasikan algoritma *branch and bound*, untuk membantu proses evaluasi nilai biaya pada tiap status, dibuat sebuah program yang menerima masukan berubah peletakkan keping di papan kemudian mengeluarkan opsi bagi pemain. Opsi tersebut berupa posisi yang dapat digunakan untuk menyimpan keping baru beserta nilai $\hat{g}(i)$, $\hat{h}(i)$, dan total biaya yang didapatkan dari sebuah opsi langkah. Keluaran tersebut dituliskan di layar dalam format ((x, y), $\hat{g}$, $\hat{h}$, biaya), dengan (x, y) indeks tempat keping sebaiknya diletakkan.

Mula-mula, program memiliki tampilan dan status persis seperti awal mula permainan Othello. Pada tampilan papan, angka 1 menandai petak yang diduduki keping milik pemain pertama atau pemain dengan warna putih, sedangkan angka 2 menandai petak yang diduduki keping milik pemain kedua atau pemain dengan warna hitam.

```
D:\STEI_2018\IF\Smt04\Stima\Makalah>python app.py
    0 1 2 3 4 5 6 7
0  | 0 0 0 0 0 0 0 0 |
1  | 0 0 0 0 0 0 0 0 |
2  | 0 0 0 0 0 0 0 0 |
3  | 0 0 0 1 2 0 0 0 |
4  | 0 0 0 2 1 0 0 0 |
5  | 0 0 0 0 0 0 0 0 |
6  | 0 0 0 0 0 0 0 0 |
7  | 0 0 0 0 0 0 0 0 |
Turn  1
Possible Moves:
((2, 4), 2, 6, 8)
((4, 2), 2, 6, 8)
((3, 5), 2, 6, 8)
((5, 3), 2, 6, 8)

Masukkan petak dengan format (i,j) atau -5,-5 untuk skip.
```

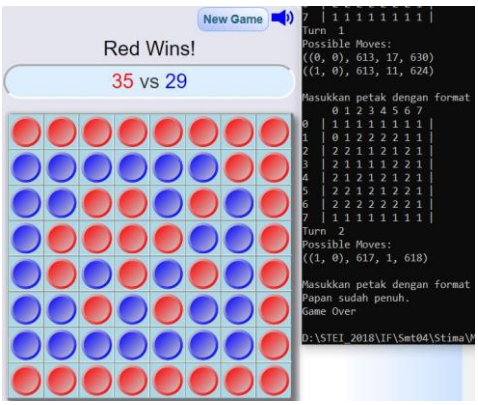
Gambar 7. Program Penghitung Biaya Algoritma *Branch and Bound*

Sumber: Dokumentasi Pribadi

Tiap giliran, program akan memeriksa keadaan papan terlebih dahulu dan menghitung gerakan yang dapat dilakukan oleh pemain yang sedang memegang giliran. Jika tidak ada gerakan yang dapat dilakukan pemain tersebut, program akan langsung lanjut ke giliran berikutnya. Bila ada, program akan membangkitkan kemungkinan langkah yang lolos dari batasan yang diberlakukan. Setelah itu, program akan menghitung biaya dari masing-masing langkah dan menampilkan langkah tersebut terurut menurun berdasarkan biayanya. Program kemudian akan meminta masukan. Rangkaian langkah tersebut terus dilakukan hingga permainan selesai atau tidak dapat dilanjutkan karena warna keping sama semua. Pada tiap langkah dalam pengujian, diambil petak yang pertama muncul setelah diperiksa oleh program. Proses dan Hasil Pengujian

Pengujian dilakukan sebanyak tiga kali untuk melihat apakah perancangan algoritma *branch and bound* dalam strategi ini efektif untuk menyelesaikan persoalan Othello dan mendapatkan nilai optimal dalam permainan. Tiap pengujian dilakukan menggunakan tingkat kesulitan berbeda yang disediakan oleh permainan daring. Berikut adalah hasil yang didapatkan dari tiap pengujian.

1) Pengujian pertama dilakukan melawan komputer dengan tingkat kesulitan mudah. Hasil pengujian pertama menghasilkan kondisi terakhir sebagai berikut.



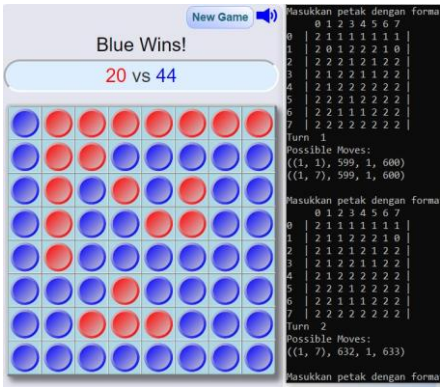
Gambar 8. Hasil Pengujian Pertama

Sumber: Dokumentasi Pribadi

Nilai optimal diraih oleh pemain dengan keping merah, yang merupakan wujud implementasi dari algoritma *branch and bound* yang telah dirancang. Hal tersebut menunjukkan bahwa dalam kasus ini, algoritma *branch and bound* lebih optimal dari algoritma yang digunakan oleh komputer lawan. Tampilan *console* di kanan menunjukkan hasil akhir yang diberikan program.

Dalam pengujian, terdapat pola yang mirip antara penghitungan dari algoritma *branch and bound* maupun langkah yang diberikan komputer lawan. Sebelum lawan memilih langkah, perhitungan dari algoritma *branch and bound* memberikan serangkaian langkah yang mungkin diambil lawan, termasuk langkah yang benar-benar dilakukan oleh lawan. Hal ini menunjukkan bahwa komputer lawan menggunakan algoritma yang mirip, yang mengandalkan penghitungan biaya dalam tiap status.

2) Pengujian kedua dilakukan melawan komputer dengan tingkat kesulitan sedang. Hasil pengujian kedua menghasilkan kondisi terakhir sebagai berikut.



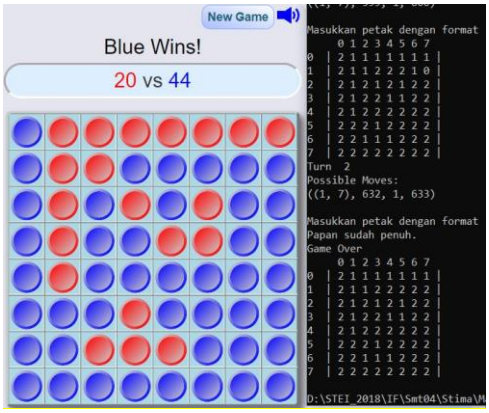
Gambar 9. Hasil Pengujian Kedua

Sumber: Dokumentasi Pribadi

Nilai optimal pada pengujian ini diraih oleh pemain keping biru, yaitu komputer lawan. Perbedaan nilai yang diraih pun cukup jauh. Komputer lawan menggunakan algoritma yang dapat merebut tempat strategis sebelum algoritma *branch and bound* ini. Hal tersebut menunjukkan bahwa ada algoritma lain yang lebih optimal dari *branch and bound* dalam menghadapi persoalan ini.

Di awal pengujian, terdapat pola yang mirip dengan pengujian pertama, baik dari hasil yang diberikan algoritma *branch and bound* maupun langkah yang diambil komputer lawan. Kemudian, pola yang mirip dari komputer lawan pada pengujian pertama terlihat pada pengujian ini. Namun, dalam pengujian ini, komputer lawan meningkatkan prioritas pada tempat strategis, seperti halnya dalam algoritma *branch and bound* yang digunakan.

3) Pengujian ketiga dilakukan melawan komputer dengan tingkat kesulitan tinggi. Hasil pengujian ketiga menghasilkan kondisi terakhir sebagai berikut.



Gambar 10. Hasil Pengujian Ketiga

Sumber: Dokumentasi Pribadi

Nilai optimal pada pengujian ini juga diraih oleh komputer lawan. Perbedaan nilai yang diraih pun cukup jauh. Komputer lawan menggunakan algoritma yang dapat merebut tempat strategis sebelum algoritma *branch and bound* ini. Hasil akhir yang didapatkan pada pengujian ini pun tidak berbeda dari hasil pengujian sebelumnya. Begitu pula dalam proses pengujiannya.

Berdasarkan pengamatan tersebut, didapatkan bahwa algoritma *branch and bound* dapat digunakan untuk menyelesaikan persoalan optimasi dalam permainan Othello. Namun, algoritma ini bukan algoritma yang paling efisien. Selama pengujian, terdapat lebih dari satu pilihan dengan biaya terbesar dalam sebuah giliran. Untuk mengoptimalkan langkah, heuristik yang digunakan dalam algoritma perlu dirancang lebih matang agar pemilihan langkah bisa lebih akurat. Heuristik yang memberikan nilai pada tiap petak perlu ditinjau kembali supaya tempat-tempat strategis tersebut diprioritaskan oleh pemain agar keping-kepingnya tidak mudah direbut lawan.

VI. KESIMPULAN

Permainan Othello adalah salah satu permainan papan tradisional yang membutuhkan strategi untuk memenangkannya. Persoalan permainan Othello termasuk dalam persoalan optimasi dengan batasan berupa aturan yang perlu diperhatikan. Salah satu algoritma yang dapat dipakai untuk menyelesaikan persoalan tersebut adalah algoritma *branch and bound* yang dibantu dengan heuristik. Algoritma *branch and bound* terbukti dapat dipakai untuk menyelesaikan persoalan Othello. Namun, algoritma ini bukan algoritma paling optimal jika dibandingkan dengan algoritma lain. Perlu dilakukan peninjauan terhadap penghitungan biaya dan heuristik yang matang jika ingin menggunakan algoritma *branch and bound* untuk persoalan seperti ini.

TAUTAN VIDEO DI YOUTUBE

Topik dan isi makalah ini diulas dan dikemas dalam bahasa yang lebih ringan di tautan berikut ini.

https://youtu.be/BJ02aQOiDH0

UCAPAN TERIMA KASIH

Penulis mengucapkan puji syukur kepada Tuhan Yang Maha Esa karena atas segala rahmat dan berkah-Nya penulis dapat menyelesaikan makalah "Penerapan Algoritma *Branch and Bound* sebagai Strategi Permainan Othello" sebagai tugas mata kuliah IF2211 Strategi Algoritma Semester Genap tahun ajaran 2019/2020 ini. Penulis berterima kasih kepada Dr. Nur Ulfa Maulidevi, S.T., M.Sc. atas segala bimbingan beliau sebagai dosen pengajar mata kuliah Strategi Algoritma. Penulis juga berterima kasih kepada seluruh keluarga, teman, dan rekan-rekan atas dukungan dan arahan yang diberikan pada penulis hingga dapat menjalankan perkuliahan hingga saat ini.

REFERENSI

[1] Archimedes' Lab. *Othello History*. Archimedes' Laboratory. Dikunjungi 3 Mei 2020 dari https://www.archimedes-lab.org/game_othello/othello.html

[2] Author Tanpa Nama. *Reversi*. Cyningstan. http://www.cyningstan.com/game/73/reversi

[3] Galli, Keith. *How to Win at Othello: Corner & Edge Strategies. YouTube*, 6 Agustus 2018, https://www.youtube.com/watch?v=SvxTrjvPrSY

[4] Kenny, Vincent, dkk. (25 Mei 2014). *Heuristic Algorithms.* Dikunjungi 3 Mei 2020 dari https://optimization.mccormick.northwestern.edu/index.php/Heuristic_algorithms

[5] Kikuchi, Daisuke. (23 Juni 2016). *Goro Haseqawa, Inventor Of Board Game Othello, Dies at 83.* The Japan Times. Dikunjungi 3 Mei 2020 dari https://www.japantimes.co.jp/news/2016/06/23/national/goro-hasegawa-inventor-board-game-othello-dies-83/#.Xq6ws6gzZPZ

[6] Munir, Rinaldi. 2020. *Algoritma Branch and Bound.* [Online]. Tersedia di http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2019-2020/stima19-20.htm/. Dikunjungi 3 Mei 2020.

[7] Pierce, Rod. (11 Juli 2018). *Play Reversi*. Math Is Fun. Dikunjungi 4 Mei 2020 dari http://www.mathsisfun.com/games/reversi.html

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, 4 Mei 2020

Difa Habiba Rahman (13518098)