Penerapan Algoritma *Greedy* dan *Breadth First Search* pada Permainan Kartu *Sevens*

Kharis Isriyanto 13514064
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13514064@std.stei.itb.ac.id

***Abstract*—Persoalan optimasi adalah sesuatu yang dapat kita temui dalam kehidupan kita sehari-hari. Kita dituntut untuk memilih sesuatu yang memberikan hasil paling optimal. Algoritma *greedy* yang berarti serakah membantu kita dalam menyelesaikan persoalan optimasi. Algoritma BFS dapat membantu dalam pencarian solusi dengan mengunjungi simpul dalam graf. Dalam makalah ini akan dibahas bagaimana algoritma *greedy* dan algoritma BFS membantu kita menemukan solusi optimal dalam permainan kartu *Sevens* yang bertujuan untuk mengeluarkan kartu sebanyak-banyaknya.**

Keywords—greedy; BFS; greedy best first search; sevens;

I. Pendahuluan

Permainan kartu *Sevens* merupakan permainan kartu yang mudah dimengerti dan menyenangkan. Meskipun mudah dimainkan permainan ini cukup menantang jika kita mengejar kemenangan karena kita bermain melawan orang-orang lain yang pastinya ingin menang juga. Dibutuhkan strategi untuk dapat mengeluarkan semua kartu yang kita miliki sekaligus menghambat lawan mengeluarkan semua kartunya. Pada permainan ini ada algoritma yang dapat membantu kita untuk menang. Algoritma tersebut adalah algoritma *greedy* dan *breadth first search*. Kedua algoritma ini berperan dalam pengembangan pohon status yang akan dibahas pada bab-bab berikutnya.

Gambar 1 - Permainan *Sevens*

II. Dasar Teori

A. Algoritma Greedy

Algoritma *greedy* adalah salah satu algoritma yang sering dipakai dan merupakan salah satu algoritma yang paling populer dalam menyelesaikan masalah optimasi.[1] Masalah optimasi tidak hanya menuntut solusi yang dapat menyelesaikan masalah tersebut, tetapi juga menuntut solusi yang menghasilkan solusi paling optimal dalam penyelesaiannya. Beberapa contoh masalah optimasi yang dapat kita temui dalam kehidupan sehari-hari contohnya adalah memilih barang berkualitas dengan batasan harga tertentu, pencarian jalur paling dekat dari tempat asal menuju tempat tujuan, dan masih banyak lagi. Dalam masalah-masalah tersebut kita mempunyai sejumlah kendala (*constraint*) dan fungsi optimasi. Contohnya dalam pemilihan barang kita mempunyai kendala barang yang dapat dibawa dan fungsi optimasi keuntungan yang kita dapat dari barang tersebut.

Algoritma *greedy* memecahkan masalah langkah demi langkah. Pada setiap langkah algoritma *greedy* mengambil pilihan terbaik yang dapat diambil pada langkah tersebut. Algoritma *greedy* tidak peduli dengan konsekuensi dari langkah tersebutdi masa depan karena algoritma *greedy* mempunyai prinsip "*take what you can get now*" atau ambil apa yang bisa didapat sekarang. Pada setiap langkah tersebut algoritma *greedy* mengasumsikan pilihan terbaik saat itu akan memberi hasil terbaik pada akhirnya. Setiap langkah akan menghasilkan hasil optimum lokal yang diasumsikan akan menghasilkan optimum global yang merupakan solusi dari permasalahan optimasi tersebut.

Penyelesaian permasalahan optimasi menggunakan algoritma *greedy* mempunyai elemen-elemen sebagai berikut:

1. Himpunan kandidat C yang berisi elemen-elemen pembentuk solusi
2. Himpunan solusi S yang berisi kandidat-kandidat terpilih sebagai solusi persoalan.

3. Fungsi seleksi yaitu fungsi yang memilih kandidat paling memungkinkan mencapai solusi optimal dalam setiap langkah.
4. Fungsi kelayakan (*feasible*) yang memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak, yaitu kandidat tersebut bersama himpunan solusi yang sudah ada tidak melanggar kendala yang diberikan.
5. Fungsi obyektif yaitu fungsi yang memaksimumkan atau meminimumkan nilai solusi.

Dengan elemen-elemen tersebut algoritma *greedy* memiliki skema umum sebagai berikut:

1. inisialisasi S dengan kosong
2. pilih sebuah kandidat dengan fungsi seleksi dari C
3. kurangi C dengan kandidat yang sudah dipilih
4. periksa apakah kandidat yang dipilih tersebut bersama dengan himpunan solusi membentuk solusi yang layak atau tidak. Jika layak masukkan kandidat tersebut ke himpunan solusi. Jika tidak, maka buang kandidat tersebut
5. periksa apakah himpunan solusi sudah memberikan solusi yang lengkap. Jika belum ulangi lagi dari langkah 2

Salah satu contoh persoalan optimasi adalah masalah penukaran uang. Permasalahan ini akan diselesaikan dengan algoritma *greedy*. Diberikan sekumpulan koin dari berbagai pecahan dan kita diminta untuk menentukan jumlah minimum koin yang dperlukan untuk ditukar dengan uang senilai A. Sebagai contoh, koin-koin yang tersedia mempunyai nilai 1, 5, 10, dan 25. Jika kita ingin menukar uang senilai 32, maka ada beberapa kombinasi untuk menukar uang tersebut di antaranya 5 koin (3 buah koin 10 dan 2 buah koin 1), 8 koin (6 buah koin 5 dan 2 buah koin 1), 32 koin (32 buah koin 1), dan masih banyak kemungkinan yang lainnya. Akan tetapi kita dapat melihat bahwa solusi optimal dapat tercapai jika kita memilih 1 buah koin 25, 1 buah koin 5, dan 2 buah koin 1 sehingga hanya dibutuhkan 4 koin. Solusi optimal tersebut dapat ditemukan dengan algoritma *greedy*.

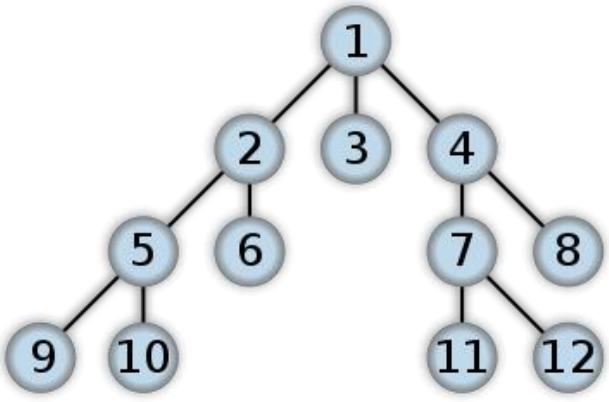
Pada persoalan penukaran uang kita memiliki himpunan kandidat koin-koin yang tersedia dalam berbagai pecahan. Dalam persoalan ini diasumsikan jumlah koin sangat banyak. Himpunan solusi S adalah himpunan koin yang jika nilai-nilainya ditambahkan mempunyai total sama dengan nilai A, yaitu nilai uang yang akan ditukar. Himpunan solusi S disebut layak jika nilai koin di dalamnya tidak melebihi A. Fungsi obyektif adalah meminimumkan jumlah koin di dalam S. Fungsi seleksinya adalah memilih kandidat yang nilainya paling besar dari himpunan kandidat yang tersisa. Hal ini dilakukan agar didapatkan jumlah koin di dalam himpunan solusi sesedikit mungkin.

Untuk menyelesaikan persoalan tersebut pertama-tama kita akan memilih koin dengan nilai paling besar yaitu 25. Karena 25 masih lebih kecil dibanding 32, maka 25 dapat dimasukkan sebagai solusi karena masih layak. Pada iterasi selanjutnya kita akan memilih koin yang terbesar lagi. Akan tetapi jika kita memilih koin 25 atau 10, maka solusi yang kita dapat akan tidak layak karena jika kandidat ditambahkan dengan himpunan solusi, nilai himpunan solusi akan lebih besar dari 32. Jadi kita akan memilih koin 5. Koin ini masih layak karena 30 masih kurang dari 32. Karena nilai himpunan solusi kita masih kurang dari 32, maka kita akan melakukan iterasi lagi. Koin 5 tidak masuk himpunan solusi lagi karena akan menyebabkan ketidaklayakan himpunan solusi. Oleh karena itu kita akan memilih koin 1 dan pada iterasi selanjutnya kita juga memilih koin 1. Karena total nilai pada himpunan solusi sudah sama dengan 32, maka iterasi dihentikan dan didapatkan koin-koin solusi adalah satu buah koin 25, satu buah koin 5, dan 2 buah koin 1. Solusi tersebut merupakan solusi optimal karena dibutuhkan 4 koin.

B. Algoritma BFS (Breadth First Search)

Breadth First Search (BFS) atau algoritma pencarian melebar adalah salah satu algoritma traversal di dalam graf selain algoritma pencarian mendalam atau *Depth First Search (DFS)*.[1] Seperti namanya, algoritma BFS mengunjungi simpul-simpul secara melebar. Misalkan simpul pertama yang kita kunjungi adalah simpul v. Selanjutnya, semua simpul yang bertetangga dengan v kita kunjungi. Setelah itu semua simpul yang belum dikunjungi dan bertetangga dengan simpul-simpul yang telah dikunjungi tadi dikunjungi dan begitu seterusnya hingga semua simpul sudah dikunjungi.

Agar lebih jelas kita akan membahas BFS pada graf pada gambar 2. Graf yang berbentuk pohon tersebut dimulai dari simpul 1. Dari simpul 1 kita mengunjungi simpul-simpul yang bertetangga dengannya yaitu simpul 2, 3, dan 4. Pada iterasi berikutnya kita akan mengunjungi simpul lain yang bertetangga dengan simpul sebelumnya dan belum pernah dikunjungi. Simpul-simpul tersebut adalah simpul 5, 6, 7, dan 8 karena simpul 5 dan 6 bertetangga dengan simpul 2 dan simpul 7 dan 8 bertetangga dengan simpul 4. Adapun simpul 3 tidak mempunyai tetangga yang belum dikunjungi. Pada iterasi berikutnya simpul 9, 10, 11, dan 12 dikunjungi. Karena semua simpul sudah dikunjungi maka algoritma tersebut selesai.



Gambar 2 - Graf BFS
Sumber: https://en.wikipedia.org/wiki/Breadth-first_search

III. DASAR PERMAINAN SEVENS

Permainan kartu *Sevens* dikenal juga dengan nama *Laying Out Sevens, Fan Tan, Card Dominoes, Crazy Sevens,* atau *Parliament.*[2] Permainan ini dimainkan dengan satu dek kartu remi yang berisi 52 kartu dengan 4 simbol yaitu sekop(*spade*), hati(*heart*), keriting(*club*), dan tahu(*diamond*), juga 13 nilai pada setiap simbolnya dimulai dari 2 s/d 10, *jack, queen, king,* dan *ace*. *Sevens* dapat dimainkan oleh 3-8 orang, tetapi biasanya permainan ini dimainkan oleh 4 orang.

Untuk memulai permainan, kartu dibagikan oleh *dealer*(orang yang membagikan kartu) kepada 4 orang secara bergantian sampai semua kartu habis. Sekarang setiap orang sudah memiliki 13 kartu di tangannya. Ada beberapa versi untuk memulai permainan. Pada versi *Fan Tan* pemain yang mendapat giliran pertama adalah pemain di sebelah kiri *dealer*.[2] Pada versi lainnya pemain pertama yang berjalan adalah pemain yang mempunyai kartu 7 tahu (*seven of diamonds*) dan pemain tersebut harus mengeluarkan kartu 7 keriting terlebih dahulu,[3] atau 7 sekop (*seven of spades*) keluar terlebih dahulu.

Setelah pemain pertama berjalan, pemain kedua dan seterusnya akan mendapat giliran untuk mengeluarkan kartu. Pemain dapat mengeluarkan kartu 7 apapun yang akan membuat barisan baru. Pemain juga dapat mengeluarkan kartu berurutan naik atau turun berdasarkan kartu yang ada pada papan permainan dan harus sesuai simbolnya. Kartu yang berurutan turun adalah dari 6 sampai dengan *ace*. Kartu yang berurutan naik adalah dari 8 sampai dengan *king*. Sebagai contoh jika di meja terdapat kartu 7 sekop maka kita dapat mengeluarkan 6 sekop pada satu sisi kartu sekop, atau kartu 8 sekop pada sisi lainnya, atau kartu 7 lainnya dengan membuat barisan baru. Misalkan pemain kedua mengeluarkan kartu 6 sekop. Pemain ketiga dapat mengeluarkan 5 sekop atau 8 sekop, atau kartu 7 lainnya jika ada. Jika tidak ada kartu yang dapat dikeluarkan, maka pemain tersebut dilewat gilirannya dan berlanjut ke pemain berikutnya.

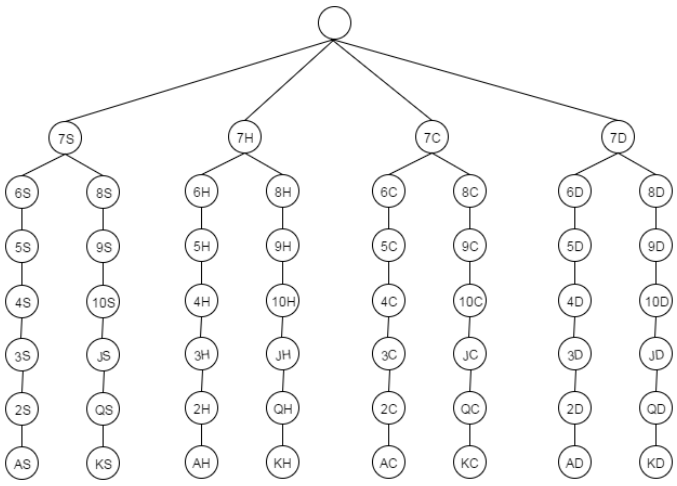
Tujuan dari permainan ini adalah pemain yang pertama kali menghabiskan kartunya akan menang. Pada versi *Fan Tan* pemain memiliki beberapa *chip*. Jika pemain tidak dapat mengeluarkan kartu, maka pemain tersebut memberikan satu *chip* kepada *dealer*.[2] Pemain yang menang akan mendapat *chip* yang dikumpulkan oleh *dealer*. Pada versi lainnya skor akan dihitung dengan kartu sisa yang ada pada tangan tiga orang lain yang kalah. Skor tersebut akan dijumlahkan pada ronde-ronde berikutnya. Pemain yang menang adalah pemain dengan ronde terendah pada akhir permainan.

Pada pembahasan di makalah ini penulis menggunakan aplikasi berbasis *mobile* yang berjalan di sistem aplikasi *android*. Aplikasi tersebut bernama *Sevens Playing Cards Game* dan diluncurkan di *Google Play* oleh Edroot. Seperti sudah dijelaskan sebelumnya ada banyak versi dalam permainan *Sevens* ini. Kita akan memakai versi yang dipakai oleh aplikasi ini di mana permainan dimainkan oleh 4 pemain. Pemilik kartu 7 sekop mulai terlebih dahulu dan harus mengeluarkan kartu tersebut. Permainan dilakukan dalam beberapa ronde dan skor akan dihitung berdasarkan kartu yang masih ada di tangan. Pemenang adalah pemegang skor terendah pada akhir permainan. Jadi kita dituntut untuk mengeluarkan semua kartu atau sebanyak-banyaknya yang bisa dikeluarkan, dan mencegah pemain lain mengeluarkan kartunya. Pada aplikasi ini jika pemain memegang kartu *king* pada akhir ronde, skor yang diperoleh akan berbeda dengan pemain yang memegang kartu *ace*. Akan tetapi hal ini tidak akan terlalu berpengaruh dalam makalah ini sehingga tidak diperlukan penjelasan lebih lanjut.

IV. PEMBAHASAN

Setiap pemain dalam permainan kartu *Sevens*, seperti yang sudah dibahas sebelumnya, mempunyai giliran dalam mengeluarkan kartu yang dimilikinya. Setiap pemain mempunyai 13 kartu sehingga setiap pemain memiliki minimal 13 giliran untuk mengeluarkan semua kartunya jika tidak ada giliran yang dilewat (*pass*). Dalam setiap giliran tersebut setiap pemain memiliki pilihan dalam mengeluarkan kartu, entah satu kartu, dua kartu, lebih, atau bahkan tidak ada kartu sama sekali yang dapat keluar. Untuk menentukan kartu mana yang akan dikeluarkan kita akan memakai algoritma *greedy* dan BFS.

Dalam penyelesaian permainan ini akan dibuat suatu pohon ruang status yang menggambarkan status permainan dalam setiap giliran. Pohon tersebut ditunjukkan pada gambar 3 di bawah ini. Setiap simpul menyatakan kartu apa yang sedang berada di ujun dan dapat ditambahkan oleh kartu lain. Simbol dalam simpul yang dipakai adalah dua karakter di mana karakter pertama adalah nilai kartu dan karakter kedua adalah lambang kartu dalam bahasa Inggris. Sebagai contoh kartu 6 hati akan dituliskan sebagai 6H, kartu raja sekop akan dituliskan sebagai KS, dst. Lambang hati akan dituliskan sebagai H (*heart*), sekop sebagai S (*spade*), keriting sebagai C (*club*), dan tahu sebagai D (*diamond*). Kartu *jack, queen, king,* dan *ace* akan dituliskan berurutan sebagai J, Q, K, dan A. Khusus untuk kartu 10 digunakan tiga karakter yaitu 10S, 10D, 10C, dan 10H.



Gambar 3 - Pohon ruang status permainan *Sevens*

Pada pohon tersebut simpul yang diekspan pertama kali adalah simpul akar yang menghidupkan empat simpul di bawahnya yaitu 7H, 7S, 7C, 7D. Simpul yang hidup menandakan simpul tersebut dapat dikunjungi dan kartu yang

diwakili simpul tersebut dapat dikeluarkan. Akan tetapi karena pada permainan ini pemain yang mendapat giliran pertama adalah pemain dengan kartu 7 sekop dan harus mengeluarkan kartu tersebut maka ada pengecualian. Jadi sebagai contoh saat pemain pertama mengeluarkan 7 sekop, maka simpul di bawahnya yaitu 6 sekop dan 8 sekop dapat dikeluarkan. Pemain selanjutnya dapat mengeluarkan 7H, 7C, 7D, 6S, atau 8S, yaitu simpul-simpul yang hidup.

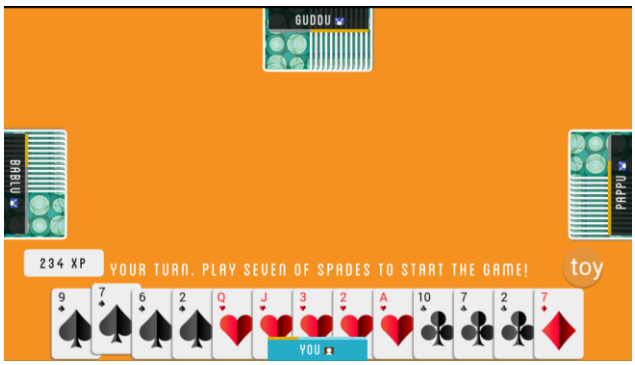
Selanjutnya kita akan memberi bobot setiap simpul untuk menentukan *greedy* yang akan digunakan. Simpul yang perlu diberi bobot hanya simpul-simpul yang kartunya kita miliki di tangan. Hal ini dikarenakan kita hanya dapat mengunjungi simpul-simpul yang kartunya ada di tangan yaitu saat kita mengeluarkan kartu tersebut. Bobot suatu simpul ditentukan oleh jumlah kartu yang ada setelah kartu tersebut dikeluarkan atau dengan kata lain jumlah simpul di bawahnya yang harus dikunjungi. Agar lebih efektif nilai tersebut dikurangi satu per empat dari jumlah simpul lawan di bawahnya. Pengurangan ini dilakukan karena kita juga harus memperhitungkan jumlah kartu lawan yang dapat keluar jika kita mengeluarkan suatu kartu. Angka ¼ didapat dengan pertimbangan ada empat pemain sehingga jumlah kartu masing-masing pemain adalah ¼ dari kartu seluruhnya. Oleh karena kartu di tangan kita lebih diprioritaskan untuk keluar daripada kartu di tangan lawan, maka nilai kartu di tangan lawan dikalikan ¼. Jadi nilai suatu simpul adalah jumlah simpul di bawahnya yang harus dikunjungi dikurangi satu per empat dari jumlah simpul yang melambangkan kartu lawan di bawah simpul tersebut.

> Nilai suatu simpul = jumlah simpul di bawahnya – ¼ x jumlah simpul lawan di bawahnya

Contohnya jika seorang pemain mempunyai kartu 7 hati, 4 hati, 5 hati, 8 hati, 9 hati, 10 hati , *queen* hati, dan kartu sisanya adalah kartu bukan hati maka bobot simpul 7H adalah 5-(7/4) atau 3,25 karena ada 5 simpul di bawahnya yang harus dikunjungi dan ada 7 simpul lawan di bawahnya. Bobot simpul 4H adalah -0,75 karena tidak ada simpul lain di bawahnya yang harus dikunjungi dan ada 3 simpul lawan di bawahnya. Sejalan dengan strategi yang biasa digunakan yaitu menahan kartu yang jika dikeluarkan dapat menguntungkan pemain lain. Dalam hal ini jika kita mengeluarkan 4H, pemain lain bisa mengeluarkan 3H, 2H, dan AH sehingga sebisa mungkin kita jangan mengeluarkan kartu 4 hati.

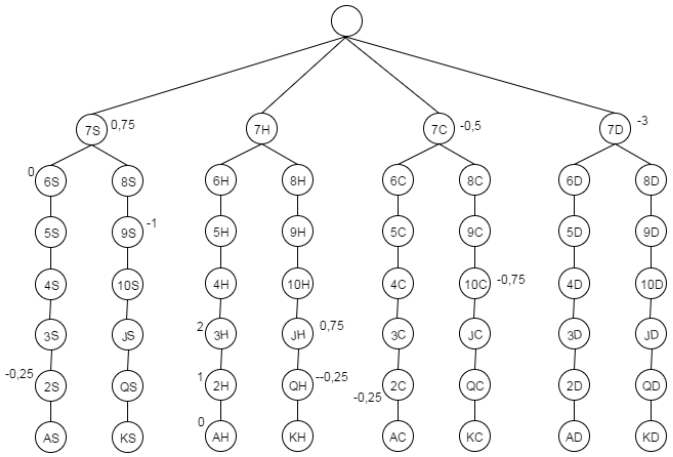
Setelah setiap simpul sudah diberi bobot, maka algoritma *greedy* akan memilih simpul mana yang harus dihidupkan. Pilih simpul dengan nilai terbesar jika ada lebih dari 2 pilihan. Jika hanya ada 1 pilihan maka pilih simpul tersebut. Jika tidak ada pilihan sama sekali maka terpaksa giliran harus dilewat.

Untuk membahas lebih lanjut, kita akan mencoba memainkan satu ronde dengan menggunakan algoritma ini. Pada awalnya kita mendapat susunan kartu seperti pada gambar 4.



Gambar 4 - Putaran pertama

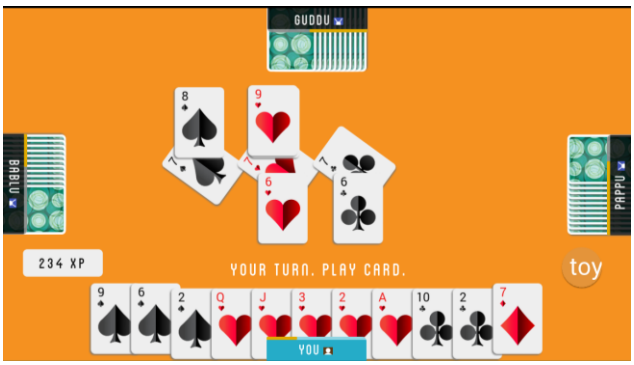
Pohon ruang status beserta bobot simpul yang ada di tangan ditunjukkan di bawah ini.



Gambar 5 - Pohon ruang status permainan *Sevens*

Pada ronde ini kita mendapat 7 sekop sehingga kita harus memulai terlebih dahulu dengan mengeluarkan kartu 7 sekop tersebut. Pada putaran berikutnya kita dapat mengeluarkan kartu 7 keriting atau 7 tahu. Simpul 7C mempunyai bobot lebih tinggi yaitu 2 dibandingkan simpul 7D yang mempunyai bobot -3. Oleh karena itu pada putaran kedua kita mengeluarkan kartu 7 keriting.

Pada putaran ketiga susunan kartunya seperti pada gambar 6. Simpul hidup yang bisa dikunjungi adalah 9S, 6S, 10H, 5H, 8C, 5C, dan 7D. Simpul yang dapat kita kunjungi adalah 9S, 6S, dan 7D. Kita dapat mengeluarkan kartu 9 sekop, 6 sekop, atau 7 keriting. Di antara ketiga kartu tersebut kartu dengan simpul yang mempunyai bobot tertinggi adalah 6 sekop yang mempunyai bobot 0. 9 sekop mempunyai bobot -1 dan 7 mempunyai bobot -3. Jadi kita mengeluarkan kartu 6 sekop.



Gambar 6 - Putaran ketiga

Pada putaran berikutnya ada satu kartu lagi yang dapat kita keluarkan selain 9 sekop dan 7 sekop yaitu *jack* hati. Simpul JH mempunyai bobot 0,75 yang lebih besar dari simpul 7D dan 9S. Jika dilihat jika kita mengeluarkan kartu *jack* hati, kita bisa mengeluarkan kartu *queen* hati, sesuai algoritma yang dipakai kartu *jack* hati atau simpul JH lebih menguntungka. Jadi pada giliran ini kita mengeluarkan kartu *jack* hati dan mengunjungi simpul JH.

Gambar 7 - Putaran keempat

Pada putaran berikutnya ada dua kartu baru lagi yang dapat dikeluarkan yaitu kartu *queen* hati hasil ekspan dari simpul JH dan kartu 10 keriting. Jika dilihat keempat kartu yang dapat dikeluarkan mempunyai bobot negatif yang berarti tidak ada kartu kita yang dapat dikeluarkan setelah kita mengeluarkan suatu kartu tersebut. Akan tetapi tujuan lain kita adalah mencegah kartu lawan keluar. Oleh karena itu kita memilih kartu dengan bobot terbesar karena jika kita mengeluarkan kartu tersebut kita dapat mencegah kartu lawan keluar lebih banyak. Kartu yang kita keluarkan yaitu *queen* hati yang memiliki bobot -0.25.

Gambar 8 - Putaran kelima

Pada putaran berikutnya muncul satu kartu baru yang muncul yaitu 3 hati. Simpul 3H memiliki bobot yang cukup besar yaitu 2 karena di bawah simpul 3H ada simpul 2H dan 1H yang dapat kita kunjungi. Jadi kita keluarkan kartu 3 hati mengingat bobot simpul lain lebih kecil dibanding 3H.

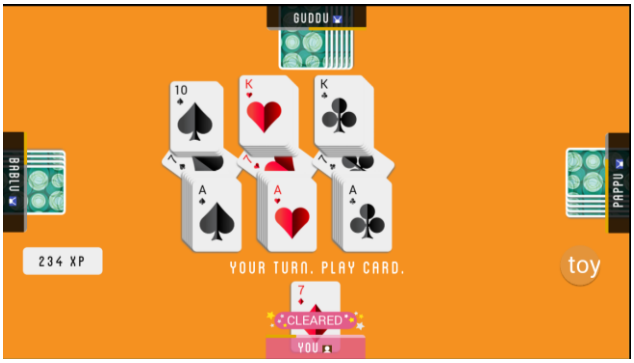
Gambar 9 - Putaran keenam

Pada putaran selanjutnya dapat kita lihat semua kartu yang kita miliki dapat kita kunjungi. Simpul yang hidup adalah 9S, 2S, AH, 10C, 2C, dan 7D di mana semuanya kita miliki. Untuk mencegah kartu pemain lain keluar pilih kartu dengan bobot terbesar. Begitu seterusnya hingga permainan selesai.

Gambar 10 - Putaran kedelapan

Pada putaran terakhir, kartu terakhir yang kita miliki adalah 7 tahu yang mempunyai bobot terendah dari semua kartu yang kita miliki pada awal permainan. Kartu ini strategis karena mencegah kartu yang lain keluar dan dapat kita lihat setelah

kita mengeluarkan kartu 7 tahu maka permainan selesai dan lawan mempunyai cukup banyak sisa kartu yang belum keluar. Jadi algoritma ini selain membantu kita mengeluarkan semua kartu yang kita miliki juga mencegah kartu-kartu pemain lain keluar.



Gambar 11 - Putaran ketiga belas

Permainan di atas adalah salah satu permainan di mana kartu yang kita dapat adalah kartu-kartu ideal sehingga pada setiap giliran kita mempunyai lebih dari satu pilihan kartu untuk dikeluarkan. Jika kita kurang beruntung dan kartu-kartu yang kita dapat mempunyai sedikit pilihan untuk keluar atau bahkan tidak dapat keluar, algoritma ini sedikit tidak berguna karena kita tidak bisa memilih dan hanya mengeluarkan kartu yang bisa dikeluarkan.

Perlu diketahui mengapa tidak digunakan algoritma lain seperti *branch and bound* atau *greedy best first search*. Sekilas algoritma yang digunakan memang mirip karena ada pemilihan simpul dengan jumlah terbesar atau terkecil dan bagaimana simpul tersebut diekspan dan menghasilkan simpul lainnya. Ada beberapa alasan mengapa tidak dipakai algoritma lain. Yang pertama pada algoritma dalam penyelesaian permainan ini simpul-simpul diekspan dan dihidupkan oleh orang-orang yang berbeda bergantung pada gilirannya. Hal ini menyebabkan pilihan dalam mengekspan suatu simpul bergantung pada kartu yang ada pada pemain. Jadi setiap simpul hidup belum tentu dapat dikunjungi oleh pemain jika pemain tersebut tidak mempunyai kartunya. Berbeda dengan *branch and bound* atau *greedy best first search* yang dapat mengekspan semua simpul yang ada.

Alasan yang kedua adalah pemberian nilai setiap simpul akan berbeda pada setiap pemain. Ini berbeda dengan algoritma *greedy best first search* atau *branch and bound* yang nilai heuristiknya atau nilai setiap simpul tidak bergantung pemain. Pada satu pohon semua nilai simpul konsisten dan tidak berubah bergantung pemain. Selain itu tujuan dari algoritma ini bukan untuk menemukan solusi melainkan untuk mengeluarkan semua kartu yang ada di tangan atau dengan kata lain mengunjungi setiap simpul dengan simbol-simbol kartu yang ada pada tangan.

V. Kesimpulan

Algoritma *greedy* bersama-sama dengan algoritma BFS dapat membantu mengambil keputusan pengeluaran kartu dalam permainan *Sevens*. Penetapan bobot simpul berdasarkan jumlah kartu yang harus keluar dan juga mencegah kartu lawan untuk keluar juga sudah cukup berhasil. Meskipun begitu penetapan bobot tersebut masih dapat disempurnakan lagi dengan penambahan faktor lain seperti contohnya perhitungan langkah-langkah dari suatu simpul ke simpul lain di bawahnya yang harus dikunjungi. Selain itu jika ada kartu-kartu berurutan yang dimiliki, contohnya 6 sekop, 5 sekop, dan 4 sekop, seharusnya diprioritaskan 6 sekop dan 5 sekop keluar terlebih dahulu dibanding kartu lain meskipun mempunyai bobot lebih tinggi.

VI. Ucapan Terima Kasih

Penulis mengucap syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmatNya makalah ini dapat selesai tepat pada waktunya. Penulis berterima kasih kepada Bapak Dr. Ir. Rinaldi Munir, M.T. dan Ibu Dr. Nur Ulfa Maulidevi, S.T. M.T. selaku dosen mata kuliah IF2211 Strategi Algoritma. Penulis juga berterimakasih kepada pengembang aplikasi *Sevens Playing Cards Game*, yaitu Edroot yang aplikasinya dipakai dalam pembuatan makalah ini, dan gambar-gambar aplikasinya dicantumkan. Tidak lupa juga penulis berterimakasih kepada orangtua dan teman-teman yang telah mendukung pengerjaan makalah ini.

Referensi

[1] Munir, Rinaldi. *Diktat Kuliah IF 2211 Strategi Algoritma*. Bandung: Penerbit Teknik Informatika Institut Teknologi Bandung.

[2] https://www.pagat.com/domino/sevens.html (diakses pada 7 Mei 2016 pukul 19.00)

[3] http://www.classicgamesandpuzzles.com/Sevens.html (diakses pada 7 Mei 2016 pukul 19.00)

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, 8 Mei 2016

Kharis Isriyanto 13514064