Implementasi Pemrogragraman Dinamis untuk Penentuan Gerakan AI dalam Persegi Bilangan

Penentuan Gerakan AI dalam Turn-Based Game Persegi Bilangan

Davin Prasetya (13514003)
Institut Teknologi Bandung
Sekolah Teknik Elektro dan Informatika
Bandung, Indonesia
13514003@std.stei.itb.ac.id

***Abstract*—Game adalah sesuatu yang terus berkembang dan tidak pernah lepas dari hidup manusia. Dalam perkembangannya, variatif game terus menerus muncul dengan gameplay masing-masing yang unik. Keterbatasan game hanya dapat dimainkan oleh minimal 2 orang pun pudar dengan berkembangnya teknologi. Intelegensi Buatan (AI) adalah pemecah batasan tersebut. AI dirancang untuk dapat berpikir sendiri tanpa masukan manual dari manusia. Permasalahannya, ketidakefektifan AI dalam berpikir dan keterbatasan program AI seringkali membuat sebuah game kurang dapat dinikmati. Dalam makalah ini, akan dibahas sebuah solusi untuk menyelesaikan salah satu masalah AI, yaitu problem pemilihan angka dalam suatu kumpulan bilangan yang kompleks, yakni berupa sebuah persegi dimana pemain maupun AI hanya dapat mengambil nilai dari sebuah bilangan di keliling bangun papan yang nilainya belum diambil. Untuk dapat memperkirakan langkah yang player akan ambil, AI menggunakan perkiraan kedepan dan menilai langkah yang diambil. Akan tetapi, karena banyak sekali langkah yang dapat diambil dan variasi nilai dari tiap langkah, hal ini menjadi masalah dalam pemrograman. Masalah ini dapat diselesaikan dengan implementasi algoritma pemrograman dinamis dengan faktor perbedaan nilai dan perkiraan kedepan(futuristic thinking) AI.**

Keywords—component; strategi game, pemrograman dinamis, persegi bilangan, futuristic thinking, intelegensi buatan

I. PENDAHULUAN

Game bermodel puzzle atau strategi merupakan suatu bentuk game yang sudah umum di masyarakat. Game bermodel seperti ini dapat dimainkan oleh seluruh umur, sehingga perkembangannya pun sangat cepat. Game bermodel puzzle atau strategi memiliki beberapa faktor yang dapat membuat game ini berdaya jual dan menarik pemakai, yakni gameplay atau cara bermain game, penggunaan asset yang sesuai dengan tema permainan, design game yang sesuai dan menarik, dan fitur fitur tambahan lainnya, seperti galeri dan sistem ekonomi. Selain faktor faktor tersebut yang kelihatan dari luar, ada sebuah faktor yang tidak kalah penting, yaitu faktor intelegensi buatan atau yang sering disingkat sebagai AI.

Artificial Intellegence(AI) adalah kemampuan berpikir secara otomatis oleh suatu program. Pola pembuatan pemikiran inilah yang menjadi tantangan setiap programmer game dalam pembuatan game, karena mayoritas game kini sudah menggunakan AI, mulai dari yang simple seperti pacman, mario bros hingga yang kompleks, seperti dalam permainan catur dan puzzle.

Dalam makalah ini, akan dibahas suatu konsep AI dalam suatu sistem game persegi bilangan untuk menentukan langkah pengambilan petak atau nilai dalam persegi bilangan. Konsep AI yang dibahas mengimplementasikan pemikiran kedepan untuk memperkirakan langkah yang pemain akan ambil. Akibatnya, jumlah node pilihan menjadi sangat banyak mengingat dalam pengambilan nilai terdapat 4*panjang sisi dari persegi untuk hanya 1 langkah pengambilan nilai. Artinya terdapat 4*panjang sisi persegi akar dari sebuah tree untuk pemikiran kedepan 1 langkah. Selain itu, pemikiran kedepan dilakukan berkali kali oleh program seiring berjalannya permainanan. Selain faktor banyaknya kemungkinan pengambilan petak oleh pengguna, ada faktor lain yaitu terus berubahnya bentuk petak. Game didesain untuk berhenti ketika terdapat suatu jalur atau path dalam petak yang memisah papan dengan syarat suatu jalur berada pada sisi tertentu terhubung dengan sisi yang bersebrangan dengannya. Akibat dari hal ini, pengambilan keputusan oleh AI menjadi makin sulit karena ada kemungkinan game akan diberhentikan oleh sengaja maupun tidak sengaja ketika AI berada pada posisi kalah atau sebaliknya

Melihat faktor-faktor tersebut, digunakanlah algoritma strategi game yang didukung dengan algoritma pemrograman dinamis yang dapat membantu komputasi dan penyelesaian masalah penentuan langkah AI dalam game bermodel persegi bilangan dan melingkupi permasalahan berubahnya bentuk papan.

II. Algoritma Pemrograman Dinamis

A. Penjelasan Singkat Algoritma Pemrograman Dinamis

Algoritma Pemrograman Dinamis atau Dynamic Programming adalah metode pemecahan masalah dengan cara menguraikan solusi yang kompleks menjadi beberapa tingkatan atau stage. Setiap stage tersebut kemudian dapat dipandang dari serangkaian keputusan yang saling berkaitan satu sama lain. Istilah pemrograman dinamis muncul karena penggunaan tabel yang ukurannya dapat terus berubah.

Pemrograman dinamis dapat terbagi menjadi 2 sudut pandang, yakni pemrograman dinamis Top-Down atau pemrograman dinamis mundur dan pemgrograman dinamis Bottom-Up atau pemrograman dinamis maju.

Karakteristik dari persoalan yang dapat diselesaikan dengan pemrograman dinamis

1. Jumlah pilihan yang mungkin berhingga
2. Solusi dapat dibangun secara bertahap
3. Solusi yang dibangun sebelumnya digunakan untuk penyelesaian solusi berikutnya
4. Pembatasan pilihan yang harus dipertimbangkan dapat dilakukan

B. Cara Kerja Algoritma Pemrograman Dinamis

Dalam algoritma pemrograman dinamis seperti yang sudah dijelaskan sebelumnya, terdapat pembagian masalah yang dibentuk kedalam solusi-solusi/stage yang saling berkaitan. Untuk dapat membentuk stage yang optimal, terdapat beberapa langkah pembentukan program dinamis:

1. Karakteristikkan struktur solusi yang optimal
2. Pendefinisian fungsi rekursif nilai solusi yang optimal
3. Penghitungan nilai rekursif sesuai pendekatan sudut pandang, Top-Down atau Bottom-Up
4. Konstruksi ulang solusi yang optimal hasil komputasi program

C. Prinsip Optimalitas

Prinsip optimalitas adalah suatu pendekatan yang membentuk algoritma pemrograman dinamis dalam menyusun rangkaian keputusan. Prinsip optimalitas berbunyi "Jika solusi total optimal, maka bagian solusi sampai tahap ke-k juga optimal". Maksud dari prinsip ini adalah jika kita bekerja dari tahap k dan menuju ke tahap k+1, kita dapat menggunakan nilai optimal dari tahap k sebagai basis tanpa harus mengulang langkah mencari nilai optimal dari awal hingga ke tahap k.

Bentuk dasar dari prinsip optimalitas dapat digambarkan menggunakan formula berikut:

$$f(X_k) = f(X_{k-1}) + C(k)$$

Dengan keterangan:

- $f(X_k)$ adalah fungsi optimal dari tahap ke k
- $C(k)$ adalah fungsi biaya/cost/nilai untuk menuju ke tahap k dari tahap sebelumnya

D. Perbedaan Algoritma Pemrograman Dinamis dengan Algoritma Greedy

Dalam algoritma pemrograman dinamis, dikenal sebuah nilai optimal dari suatu langkah. Hal ini tidaklah berbeda dengan greedy yang selalu mengambil nilai yang terbesar/terkecil. Perbedaan kedua algoritma ini ada pada 2 hal. Hal pertama yang membedakan kedua algoritma adalah jenis dari keputusan yang dihasilkan dari kedua algoritma. Algoritma greedy menghasilkan keputusan yang merupakan local optimum value atau nilai optimum lokal, artinya hasil keputusan dari algoritma Greedy belum tentu merupakan nilai optimum secara keseluruhan sistem. Sedangkan algoritma pemrograman dinamis menghasilkan total optimum value atau nilai optimum secara keseluruhan. Hal kedua yang membedakannya terletak pada jumlah rangkaian keputusan yang dipertimbangkan. Algoritma greedy akan menghapus semua keputusan lainnya yang nilainya lebih kecil dari optimum value yang ia dapatkan. Sedangkan algoritma pemrograman dinamis akan menyimpan rangkaian tersebut dan menjadikannya sebagai faktor penentu hingga solusi optimum keseluruhan ditemukan.

III. Futuristic Thinking

A. Penjelasan Singkat Futuristic Thinking

Futuristic Thinking adalah cara pemikiran AI untuk menentukan gerakannya melalui faktor simulasi pergerakan player yang dikomputasi. AI akan membuat semua kemungkinan pergerakan player dalam beberapa step kedepan sesuai dengan kebutuhan game. Dari semua kemungkinan tersebut, akan dihitung nilai value masing-masing kemungkinan yang kemudian akan dibandingkan dengan pergerakan AI sendiri.

B. Penyusunan Futuristic Thinking

Dalam futuristic thinking yang sudah dijelaskan sebelumnya, terdapat tahapan dalam menentukan faktor-faktor yang perlu disimulasikan oleh komputasi AI. Berikut langkah-langkah dalam menyusun konsep futuristic thinking dengan baik:

1. Analisis faktor penentu yang merujuk pada alasan pengambilan keputusan
2. Analisis fungsi batasan dalam pembuatan simulasi
3. Komputasikan setiap langkah yang mungkin dilakukan oleh player
4. Proses data yang sudah dikomputasi dengan algoritma yang cocok dan sesuai dengan kebutuhan

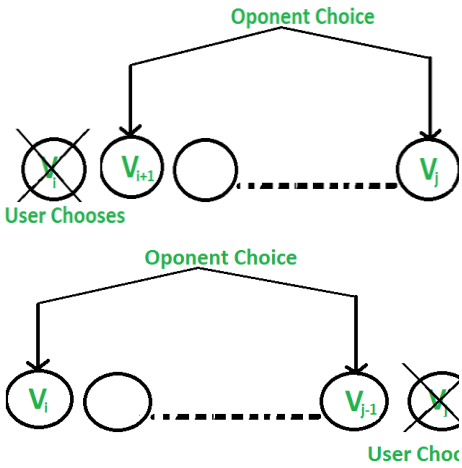

Fig 1. Contoh Futuristic Thinking 2 Kemungkinan
Source :
http://www.geeksforgeeks.org/dynamic-programming-set-31-optimal-strategy-for-a-game/

IV. Persegi Bilangan

A. *Penjelasan Singkat Persegi Bilangan*

Game Persegi Bilangan adalah turn-based game strategi dimana player akan berhadapan dengan sebuah AI yang disusun dalam game. Tujuan dari game ini adalah mengumpulkan nilai lebih besar dari AI yang diperoleh dari pengambilan nilai dari petak. Petak akan terdiri dari beberapa pilihan ukuran sesuai yang diinginkan pemain dengan syarat panjang petak adalah bilangan genap non negatif demi keseimbangan urutan memilih.

Papan Persegi Bilangan							
25	5	17	2	6	20	15	26
31	39	58	13	40	41	14	1
56	22	12	53	62	27	60	43
8	55	30	59	24	61	10	52
57	32	64	7	29	19	42	18
21	4	47	33	28	45	35	44
38	48	23	50	36	3	63	11
9	37	49	16	34	51	20	46

Fig 2.Contoh Papan Persegi Bilangan
Area hijau menandakan petak boleh diambil
Area Krem menandakan petak belum boleh diambil

B. *Sistem Permainan*

Game berbasis turn-based, setiap giliran tiap pemain hanya boleh mengambil 1 petak. Petak yang diberbolehkan untuk diambil adalah petak yang merupakan keliling dari area petak yang masih memiliki nilai atau belum terambil. Petak yang sudah diambil akan bernilai 0 atau tidak bisa diambil lagi. Setiap giliran pemain diwajibkan untuk mengambil, tidak diperbolehkan untuk skip. Pemain yang mengambil petak tertentu akan mendapat nilai sebanyak point yang tercantum di petak tersebut.

Papan Persegi Bilangan							
25		17	2	6	20	15	26
31			13	40	41	14	1
56	22	12	53	62	27	60	43
8	55	30		24	61	10	52
	32			29	19	42	
21	4		33	28	45	35	44
38	48				3	63	11
9	37	49	16		51	20	46

Fig 3. Contoh Papan Persegi Bilangan Setelah 13 Langkah
Area hijau menandakan petak boleh diambil
Area Krem menandakan petak belum boleh diambil
Area Abu-abu menandakan petak sudah diambil

Game akan diberhentikan apabila seluruh petak sudah terambil atau terdapat suatu jalur petak terambil yang menghubungkan kedua sisi yang berseberangan. Permainan mungkin diberhentikan meskipun banyaknya petak yang diambil oleh pemain berbeda, hal ini dapat diakibatkan oleh langkah sengaja ataupun tidak sengaja dari pemain yang memaksa game diberhentikan dengan cara membuat jalur yang menghubungkan dua sisi bersebrangan. Pada akhir dari permainan, nilai kumulatif pemain dan AI akan dihitung. Pemain yang menang adalah pemain yang memiliki point lebih tinggi dari pemain satunya.

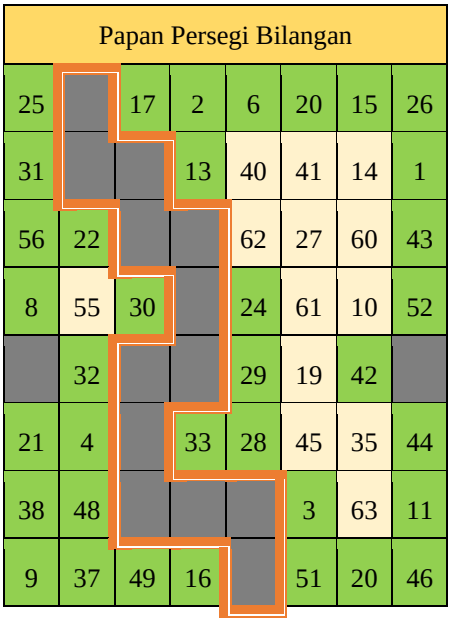

Fig 4. Contoh Kasus Permainan Diberhentikan
Area hijau menandakan petak boleh diambil
Area Krem menandakan petak belum boleh diambil
Area Abu-abu menandakan petak sudah diambil

C. Algoritma AI Game Persegi Bilangan

AI lawan pada game ini akan menerapkan algoritma pemrograman dinamik dan futuristic thinking. Algoritma yang terlebih dahulu diimplementasikan dalam AI adalah futuristic thinking. Pertama, program membuat tabel sebanyak keliling papan. Tabel menandkan simulasi pengambilan AI untuk petak tersebut. Setiap tabel berisikan kemungkinan tindakan yang akan player ambil setelah AI mengambil suatu petak. Setiap nilai dalam tabel memiliki 2 attribut, attribut yakni nilai petak (int tileValue) dan nilai harapan pemain(int expectedWinValue).

Tile Choice	Value	Optimum Value	Optimum Route
1	V_1 playerSimulation [] T_1	$F(V_1,T_1)$	i*
2	V_2 playerSimulation [] T_2	$F(V_2,T_2)$	i*
⋮	⋮	⋮	⋮
n	V_n playerSimulation [] T_n	$F(V_n,T_n)$	i*

Table 1. Bentuk Tabel Decision AI

```
class playerSimulation{
       int tileValue;
       int expectedWinValue;
   };
```

ExpectedWinValue disini adalah simulasi AI sebagai player, artinya AI akan mensimulasikan gerakan selanjutnya seperti ia adalah player yang bergerak setelahnya. Attribut ini diperlukan karena menurut psikologis, dalam merencanakan strategi manusia mayoritas hanya dapat berpikir 1 langkah kedepan. Untuk mendapat nilai expectedWinValue, digunakan algoritma Greedy yaitu mencari selisih terbesar dari nilai yang pemain ambil dan lawan akan ambil. Teori yang digunakan adalah strategi greedy dimana anggapannya lawan akan bermain mengambil nilai yang besar. Sehingga dapat disusun algoritma getExpectedWinValue dengan cara

$$F(X_n) = Max\{V_p - \max((X_{n-1}, p)\}$$

$F(X_n)$ adalah expectedWinValue dengan Xn tabel papan, menghasilkan nilai perbedaan terbesar antara nilai petak player dengan nilai petak lawan.

V_p adalah nilai petak yang akan diambil player

$\max((X_{n-1}, p)$ adalah nilai maximum yang akan lawan ambil menurut algoritma Greedy.

Setelah semua nilai value dari TileChoice terpenuhi, sudut pandang kita kembalikan ke sudut pandang AI. AI memerlukan nilai optimum yang disimpan ke currentValue. Nilai optimum dapat diperoleh dari kalkulasi value tile (V_n) dengan playerSimulation (T_n). Dari persoalan ini, algoritma paling cocok yang dapat digunakan adalah algoritma pemrograman dinamik. Fungsi optimum dari algoritma ini dapat dinyatakan nilai kalkulasinya dengan cara

Basis :

$$F(V_1, T_1) = Max\{V_1 - T_1.expectedWinValue\}$$

Rekurens :

$$F(V_n, T_n) = Max\{F(V_{n-1}, T_i) + V_1 - T_1.expectedWinValue\}$$

Rekurensi algoritma pemrograman dinamik akan terus dipanggil hingga permainan selesai. Adapun kasus khusus dimana permainan secara paksa diberhentikan karena terdapat jalur yang menghubungkan dua sisi bersebrangan. Deteksi adanya jalur yang menghubungkan dua sisi bersebrangan dapat dilakukan dengan mudah, yaitu dengan pencarian adjensi dari tile yang sudah dilewati. Pencarian adjensi bertujuan untuk mencari tile lain yang juga sudah dilewati di sekeliling tile. Arah yang termasuk dalam keliling adalah sistem 4 kemungkinan jalan. Pencarian berlanjut hingga semua adjensi sudah dilewati atau sebuah akar dari pencarian mencapai index yang bersebrangan dari index awal.

Setelah mengetahui cara pendeteksian kasus khusus, sekarang pertimbangan kasus khusus dalam algoritma dapat dilakukan. Mulai dari pencarian nilai expectedWinValue, dilakukan pengecekan isEnd setelah mensimulasikan pengambilan petak. Jika isEnd bernilai true dan total value dari player melebihi AI, maka expectedWinValue akan di assign dengan nilai INF yang merupakan konstanta int sangat besar

yang tidak mungkin dapat dicapai oleh jumlah nilai tile. Demikian juga dengan algoritma AI bagian pemrograman dinamisnya. Akibat dari perubahan nilai expectedWinValue berupa nilai yang sangat besar, tidak perlu dilakukan kasus khusus dalam penyettingan algoritma. Tetapi masih ada kasus lain sama seperti kasus penentuan expectedWinValue. Dilakukan pengecekan isEnd dan bila bernilai true, dilakukan pengecekan nilai total kedua pihak. Apabila AI lebih tinggi maka AI akan mengeset nilai Optimum Value $F(V_n,T_n)$ dengan nilai INF dan menyelesaikan permainan.

V. Permasalahan

Permasalahan yang mungkin terjadi adalah ketidakseimbangan game karena distribusi pada papan bilangan tidak baik. Papan Bilangan dibilang tidak baik apabila ada suatu nilai pencilan ataupun berkumpulnya nilai yang tidak variatif dalam satu kumpulan tile.

Permasalahan lain adalah faktor pemain. Di dalam algoritma yang dipakai dalam makalah ini adalah asumsi bahwa player optimal dan sanggup berpikir 1 langkah kedepan. Dalam kenyataan nya tidak semua player seperti itu. Akibat dari hal ini adalah ketidakcocokan AI dengan player sehingga game dapat terasa tidak menarik.

Di sisi lain dari permasalahan game secara keseluruhan adalah belum ditesnya game dengan sistem selesai paksa jika jalur yang menghubungkan kedua sisi. Hal ini berakibat pada keseimbangan game yang mungkin tidak seimbang.

Dari sudut pandang algoritma sendiri, algoritma Futuristic Thinking ketika mencari expectedWinValue mungkin masih terdapat ketidakcocokan dengan target pemain. Hal ini harus dianalisa dengan pengetesan kepada massa sehingga penentuan algoritma dapat lebih cocok.

VI. Kesimpulan

Algoritma pemrograman dinamik dapat diterapkan dalam game papan bilangan dan dapat menyelesaikan permasalahan yang ada dalam game bermodel papan bilangan.

Algoritma Futuristic Thinking bersifat fleksibel dalam mensimulasikan player. Fungsi yang terdapat dalam makalah ini merupakan murni pemikiran sehingga masih terbuka luas untuk modifikasi.

Penggunaan algoritma Greedy dalam penentuan expectedWinValue tergolong cukup cocok mengingat waktu komputasi dapat diminimalisasi dengan tidak banyak mengurangi akurasi komputasi.

Game papan bilangan masih terbuka luas untuk pengembangan berupa modifikasi penentuan expectedWinValue yang lebih cocok dan mangkus dan Futuristic Thinking yang sesuai dengan target pengguna.

Ucapan Terima Kasih

Penulis mengucapkan terima kasih kepada Tuhan Yang Maha Esa karena atas bantuan-Nya makalah ini dapat selesai. Selain itu penulis juga mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir, M.T dan Ibu Dr. Nur Ulfa Maulidevi, S.T M.T selaku dosen kuliah Strategi Algoritma Teknik Informatika ITB yang telah memberikan ilmu yang bermanfaat dalam penulisan makalah ini.

Referensi

[1] http://www.ics.uci.edu/~goodrich/teach/cs161/notes/dynamicp.pdf diakses tanggal 8 Mei 2016

[2] https://www.codechef.com/wiki/tutorial-dynamic-programming diakses tanggal 8 Mei 2016

[3] http://www.geeksforgeeks.org/dynamic-programming-set-31-optimal-strategy-for-a-game/ diakses tanggal 8 Mei 2016

[4] Presentasi Program Dinamis (2015).pptx bahan kuliah IF2211:Stratego Algoritma oleh Rinaldi Munir diakses tanggal 8 Mei 2016

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, 9 Mei 2016

Davin Prasetya, 13514003