Aplikasi Algoritma *Greedy* pada AI dalam Permainan Age of Empire II

Arie Pratama Sutiono/13509007
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13509007@std.stei.itb.ac.id

***Abstrak*—Pada makalah ini akan dibahas mengenai penggunaan salah satu metode *strategi algoritmik* pada salah satu permaian yang pernah terkenal sekitar tahun 1999, yaitu *Age of Empire II*. Permainan yang berbasis strategi ini dapat dimainkan oleh beberapa pemain, dan pemain tersebut dapat berupa manusia maupun AI (*Artificial Intelligence*). Metode strategi algoritmik yang dibahas disini adalah *algoritma Greedy* yang akan diimplementasikan pada AI yang ada pada permainan tersebut. Algoritma *Greedy* yang akan dicoba diaplikasikan untuk AI disini adalah *Greedy* dengan membangun kastil secepat mungkin dan *Greedy* dengan membangun *wonderi* secepat mungkin.**

Kata kunci— Strategi Algoritmik, Age of Empire II, Artificial Intelligence, algoritma Greedy, kastil, wonder.

I. PENDAHULUAN

Berbagai masalah muncul dalam kehidupan sehari – hari kita seiring berkembangnya teknologi. Tuntutan pemecahan masalah dengan cara otomatisasi semakin besar. Perkembangan pesat ilmu pengetahuan dan teknologi salah satunya dapat dirasakan di bidang digital, khususnya di bidang teknologi informasi.

Permasalahan yang dihadapi manusia saat ini menuntut untuk pencarian solusi yang lebih cepat dan mangkus. Teknologi informasi kini digunakan secara luas untuk pemecahan masalah hampir di semua bidang pengetahuan lain karena kasus yang telah disebutkan diatas. Teknologi informasi menawarkan pemecahan masalah yang lebih cepat dan mangkus. Pemecahan masalah dengan kompleksitas tinggi dengan teknologi informasi menggunakan algoritma yang sesuai dengan kebutuhan dan komputer sebagai alat untuk implementasi algortima tersebut. Pemecahan masalah dengan algoritma berarti menspesifikasikan urutan langkah – langkah apa saja yang harus dilakukan suatu komputer untuk memecahkan masalah tertentu dan komputer yang akan melakukan perhitungannya.

Perkembangan ilmu pengetahuan dan teknologi ternayata tidak hanya semata – mata berguna untuk pekerjaan yang bersifat "akademis" saja, namun juga berguna untuk hiburan multimedia. Salah satu bentuk hiburan yang akan dibahas pada makalah ini adalah mengenai permainan atau yang lebih akrab dikenal dengan sebutan *game*.

Pada awalnya *game* hanya permainan yang dilakukan secara manual oleh satu orang saja yang berkomunikasi dengan komputer dan mengikuti aturan yang diberikan. Kemudian *game* pun berkembang dan terus berkembang, sehingga dapat dimainkan oleh beberapa pemain dan bahkan dapat bermain sendiri dengan pemain – pemain yang sudah diprogram, yang lebih dikenal dengan nama pemain komputer atau AI (*Artificial Intelligence*). AI ini sudah deprogram untuk mengikuti peraturan suatu *game* dan berusaha untuk mengalahkan pemain. AI diciptakan agar pemain merasa tertantang untuk mengalahkan program ini dengan kemampuan yang ia miliki dan membuat suatu *game* menjadi menantang.

Pada makalah ini akan dibahas mengenai pembuatan program AI dengan menggunakan salah satu algoritma yang umum digunakan, yaitu algortima *Greedy*. *Game* yang dibahas disini hanya terbatas pada *game* yang berjudul Age of Empire II.

II. DASAR TEORI

2.1. Strategi Algoritmik

Strategi adalah rencana yang cermat mengenai kegiatan untuk mencapai sasaran khusus (Kamus Besar Bahasa Indonesia, edisi Tahun 1998).

Suatu rencana dapat tediri dari suatu metode atau teknik yang digunakan untuk mencapai sasaran khusus tersebut.

Makna dari kata "metode" dan "teknik" memang relative sama, namun konteks penggunaannya kadang berbeda. Metode adalah cara kerja yang bersistem untuk memudahkan pelaksanaan suatu kegiatan guna mencapai tujuan yang ditentukan, sedangkan teknik adalah cara mengerjakan sesuatu.

Sedangkan pengertian algoritma sendiri adalah urutan langkah – langkah untuk memecahkan suatu masalah. Definisi lain dari algoritma yaitu deretan langkah – langkah komputasi yang mentransformasikan data masukan menjada keluaran [COR92].

Pendefinisian Strategi algoritmik dengan melihat definisi – definisi diatas adalah sebagai berikut:

> **Strategi algoritmik** adalah *kumpulan metode atau teknik untuk memecahkan masalah guna mencapai tujuan yang ditentukan, yang dalam hal ini deskripsi metode atau teknik tersebut dinyatakan dalam suatu urutan langkah-langkah penyelesaian.*

Secara umum, strategi pemecahan masalah dapat

dikelompokan sebagai berikut:

1. Strategi solusi langsung (*direct solution strategies*).
   Beberaoa metode yang termasuk dalam strategi ini:
   - Algoritma *Brute Force*
   - Algoritma *Greedy*
2. Strategi berbasis pencarian pada ruang status (*state – space base strategies*).
   Beberapa metode yang termasuk dalam strategi ini:
   - Algoritma *Backtracking*
   - Algoritma *Branch and Bound*

3. Strategi solusi atas – bawah (*top – down solution strategies*).
   Metode yang termasuk dalam strategi ini adalah algoritma *Divide and Conquer*.

4. Strategi solusi bawah – atas (*bottom – up solution strategies*).
   Metode yang termasuk dalam strategi ini adalah *Dynamic Programming*.

2.2. Algoritma Greedy

Algoritma *Greedy* mungkin merukana metode yang paling populer untuk memecahkan masalah optimasi. Algoritma ini bersifat sederhana dan *straightforward* dalam pemecahan masalah. Algoritma *Greedy* membentuk solusi langkah per langkah. Terdapat banyak pilihan yang perlu dieksplorasi pada setiap langkah solusi, oleh karena itu pada setiap langkah harus diambil keputusan yang terbaik dalam menentukan pilihan.

Pendekatan yang digunakan dalam algoritma *greedy* adalah dengan membuat keputusan yang mungkin akan memberikan perolehan terbaik, yaitu dengan membuat keputusan yang optimum lokal pada setiap langkah dengan harapan bahwa langkah – langkah yang diambil tersebut dapat mengarah ke solusi optimum global, yaitu solusi paling optimal untuk permasalahan yang diatasi.

Pada umumnya algoritma *greedy* terdiri dari elemen – elemen berikut ini:

1. Himpunan kandidat, C.
   Himpunan ini berisi elemen – elemen pembentuk solusi.
2. Himpunan solusi, S.
   Berisi kandidat – kandidat yang terpilih sebagai solusi persoalan. Himpunan bagian adalah himpunan bagian dari himpunan kandidat.
3. Fungsi seleksi.
   Fungsi yang pada setiap langkah memilih kandidat yang paling memungkinkan mencapai solusi optimal.
4. Fungsi kelayakan
   Fungsi yang memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak, yaitu kandidat tersebut bersama – sama dengan himpunan solusi yang sudah terbentuk tidak melanggar kendala (*constraint*) yang ada. Kandidat yang layak dimasukkan ke himpunan solusi, sedangkan kandidat yang tidak layak tidak pernah dipertimbangkan lagi.
5. Fungsi obyektif
   Fungsi yang memaksimumkan atau meminimumkan nilai solusi.

Salah satu contoh pemecahan masalah dengan algoritma *greedy* adalah masalah penjadwalan. Berikut ini adalah contoh masalah penjadwalan yang diselesaikan dengan algoritma *greedy*:

John adalah seorang yang amat energetic, dan pada suatu pagi setelah sarapan ia membuat daftar aktifitas yang mungkin dilakukannya seperti berikut:

ID	Scheduled Activity	Time Span
1	Debug the room	Monday, 10:00 PM - Tuesday, 1:00 AM
2	Enjoy a trip to Hawaii	Tuesday, 6:00 AM - Saturday, 10:00 PM
3	Win the Chess Championship	Tuesday, 11:00 AM - Tuesday, 9:00 PM
4	Attend the Rock Concert	Tuesday, 7:00 PM - Tuesday, 11:00 PM
5	Win the Starcraft Tournament	Wednesday, 3:00 PM - Thursday, 3:00 PM
6	Have some paintball fun	Thursday, 10:00 AM - Thursday, 4:00 PM
7	Participate in the TopCoder Single Round Match	Saturday, 12:00 PM - Saturday, 2:00 PM
8	Take a shower	Saturday, 8:30 PM - Saturday 8:45 PM
9	Organize a Slumber Party	Saturday, 9:00 PM - Sunday, 6:00 AM
10	Participate in an "All you can eat" and "All you can drink" contest	Saturday, 9:01 PM - Saturday, 11:59 PM

John ingin agar ia melakukan aktifitas diatas sebanyak mungkin dengan waktu yang terbatas. Jika kita ingin membuat daftar lengkap semua pilihan yang mungkin maka aka nada 2^N pilihan dan mempertimbangkan pilihan sebanyak itu akan membutuhkan waktu yang cukup lama. Maka strategi yang kita pilih adalah memilih aktivitas dengan waktu yang paling sedikit dan tidak bertabrakan dengan aktivitas lainnya. Dengan menghitung lama waktu yang diperlukan untuk setiap aktivitas berdasarkan tabel diatas maka didapatkan kesimpulan sebagai berikut:

- Mandi (15 menit)

- Berpartisipasi pada pertandingan topcoder (2 jam)
- Berpartisipasi pada kontes "All you can eat" dan "All you can drink" (2 jam 58 menit)
- Melakukan debugging (3 jam)
- Menghadiri konser rock (4 jam)
- Bermain paintball (6 jam)

2.3. PC Game

PC *game* adalah video *game* yang dimainkan pada komputer personal, bukan pada mesin *arcade*. PC *game* telah berevolusi dari grafik dan *gameplay* yang sederhana, seperti *Spacewar!* ke *game – game* pada masa kini yang lebih canggih.

2.4. AI

Kecerdasan Buatan (bahasa Inggris: *Artificial Intelligence* atau *AI*) didefinisikan sebagai kecerdasan *entitas* buatan. Sistem seperti ini umumnya dianggap komputer. Kecerdasan diciptakan dan dimasukkan ke dalam suatu mesin (komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, permainan komputer (*games*), *logika fuzzy*, jaringan syaraf tiruan dan robotika.

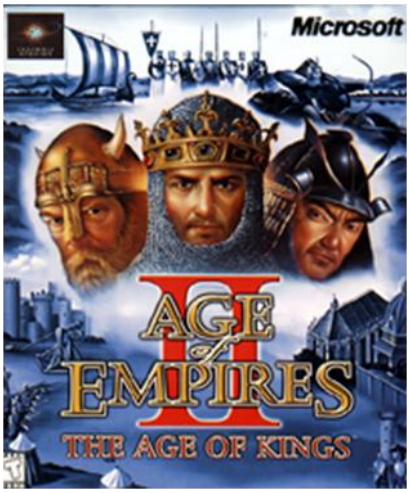
Banyak hal yang kelihatannya sulit untuk kecerdasan manusia, tetapi untuk Informatika relatif tidak bermasalah. Seperti contoh: mentransformasikan persamaan, menyelesaikan persamaan integral, membuat permainan catur atau Backgammon. Di sisi lain, hal yang bagi manusia kelihatannya menuntut sedikit kecerdasan, sampai sekarang masih sulit untuk direalisasikan dalam Informatika. Seperti contoh: Pengenalan Obyek/Muka, bermain sepak bola.

Walaupun AI memiliki konotasi fiksi ilmiah yang kuat, AI membentuk cabang yang sangat penting pada ilmu komputer, berhubungan dengan perilaku, pembelajaran dan adaptasi yang cerdas dalam sebuah mesin. Penelitian dalam AI menyangkut pembuatan mesin untuk mengotomatisasikan tugas-tugas yang membutuhkan perilaku cerdas. Termasuk contohnya adalah pengendalian, perencanaan dan penjadwalan, kemampuan untuk menjawab diagnosa dan pertanyaan pelanggan, serta pengenalan tulisan tangan, suara dan wajah. Hal-hal seperti itu telah menjadi disiplin ilmu tersendiri, yang memusatkan perhatian pada penyediaan solusi masalah kehidupan yang nyata. Sistem AI sekarang ini sering digunakan dalam bidang ekonomi, obat-obatan, teknik dan militer, seperti yang telah dibangun dalam beberapa aplikasi perangkat lunak komputer rumah dan video *game*.

III. APLIKASI ALGORITMA *GREEDY* PADA AI DI *GAME* AGE OF EMPIRE II

3.1. *Tentang game Age of Empire II*

Age of Empires II seri The Age of Kings (sering dibaca Age of Empires II saja atau AOE2) adalah sebuah permainan bertipe Real Time Strategy, yang merupakan sekuel kedua dari seri Age of Empires.



Game ini diluncurkan pada 1999, dan menjadi *game* yang laris. Berikut ini adalah spesifikasi *game* ini :

Age of Empires II: The Age of Kings	
Pengembang	Ensemble Studios
Penerbit	Microsoft
Version	2.0a
Sistem operasi	Microsoft Windows, PlayStation 2, Apple Macintosh
Tanggal rilis	30 September, 1999
Kategori permainan	Real-time strategy
Mode permainan	Multiplayer melalui IPX, TCP/IP,Modem, *Game*Spy Arcade, atau *Game*Ranger
Batas umur pemain	ELSPA: 3+ ESRB: T (Remaja)
Media	CD (1)
Kebutuhan sistem	Pentium 166 MHz CPU, 32 MBRAM, 200 MB HD

Alat permainan	keyboard, mouse

Dalam permainan ini, pemain mengendalikan sebuah masyarakat untuk mencapai kemenangan. Pemain dapat mengendalikan penduduk untuk mendirikan atau memperbaiki bangunan atau mengumpulkan sumber daya yang diperlukan untuk pembangunan dan pembuatan pasukan. Pemain juga dapat mengendalikan pasukan untuk berpindah, membuat formasi, berpatroli ataupun menyerang musuh. Penduduk dan pasukan dihasilkan oleh bangunan tertentu.

Dalam permainan ini terdapat empat jenis sumber daya, yaitu kayu,makanan, emas maupun batu. Kayu dibutuhkan untuk mendirikan bangunan, mendirikan ladang (sumber makanan), dan melatih pemanah. Makanan dan emas dibutuhkan untuk melatih penduduk maupun pasukan, serta menemukan sebuah teknologi. Batu diperlukan untuk membangun sarana pertahanan dari batu seperti benteng, menara, dan tembok.

Permainan dimulai pada zaman kegelapan dan berakhir pada zamanimperial, yang merupakan awal dari renaisans. Keseluruhan terdiri dari empat zaman. Semakin maju suatu masyarakat, maka mereka mampu melatih pasukan atau menemukan teknologi yang lebih modern dan beragam. Untuk maju dari suatu zaman ke zaman lain, dibutuhkan sejumlah besar sumberdaya.

Terdapat dua mode permainan, *campaign*(kampanye), maupun *random map* (peta acak). Pada mode *campaign* pemain memainkan seorang pemimpin untuk memenangkan perang-perang bersejarah. Perang bersejarah yang dapat dimainkan antara lain Salahuddin Ayyubi diPerang Salib, penaklukan oleh Jenghis Khan, dan lain-lain. *Random map* adalah permainan standar dimana pemain berusaha melawan komputeruntuk mencapai beberapa target yang disebut dengan syarat kemenangan.

Dalam permainan ini juga terdapat aspek diplomasi. Pemain dapat berdagan, bersekutu, maupun menyatakn perang terhadap pemain lain atau komputer.

3.2. *Ai yang Menggunakan Algoritma Greedy dengan Waktu paling Minimaluntuk Membangun Kastil*

Dalam permainan Age of Empire II ini pemain dituntu saling menyerang agar dapat memenangkan permainan . pemain dengan penyerangan dan pertahanan yang kuat dan dapat bertahan sampai permainan berakhir adalah pemain yang akan menjadi pemenang.

Untuk dapat membentuk pertahanan atau penyerangan yang kuat atau bahkan kedua – duanya, maka pemain harus membangun bangunan militer yang dapat memproduksi kekuatan militer. Khusus untuk pertahanan disediakan bangunan – bangunan yang dapat menunjang pertahanan dengan menyerang pasukan musuh yang terdekat. Bangunan tersebut berupa menara (*tower*), kastil (*castle*), dan pusat kota (*town center*). Menggunakan bangunan sebagai pertahanan merupakan cara efektif, karena selain dapat bertahan dari serangan pasukan manusia, bangunan ini dapat menambah kekuatan serangnya dengan memasukan orang ke dalam bangunan tersebut.

Kastil adalah salah satu bangunan yang efektif, karena dapat digunakan untuk bertahan dan menyerang. Bangunan ini dapat digunakan untuk bertahan karena menyerang pasukan musuh yang terdekat, dan digunakan untuk menyerang karena dapat memproduksi kekuatan militer yang istimewa. Kekuatan militer yang dihasilkan oleh bangunan kastil ini unik untuk setiap bangsa yang digunakan oleh pemain dan pada umumnya merupakan pasukan yang terkuat yang dimiliki oleh suatu bangsa, selain itu kastil juga dapat memproduksi pasukan yang khusus untuk menghancurkan bangunan yang sulit untuk dihancurkan dengan pasukan manusia.

Penulis menggunakan program *aieditor* yang merupakan framework untuk menulis script AI pada *game* Age of Empire II. File yang sudah dibuat kemudian disimpan dengan ekstensi ".per" yang menandakan bahwa file tersebut adalah file yang berisi aturan untuk AI. File tersebut disimpan ke folder bernama "Ai" di dalam folder tempat *game* Age of Empire II berada. Selain file dengan ekstensi ".per" tersebut, pembuat *script* AI juga membutuhkan file berekstensi ".ai" yang bernama sama dengan file ".per" yang dibuat, hal ini dimaksudkan agar AI yang sudah dibuat dikenal oleh *game* Age of Empire II.

Untuk mencapai suatu kemenangan dalam permainan ini, khususnya pada mode *Random map* maka dibutuhkan kecepatan dan ketepatan dalam pengaturan sumber daya dan dalam melakukan pembangunan. Salah satu strategi untuk memperoleh penyerangan sekaligus pertahanan yang kuat adalah dengan mambangun kastil secepat mungkin. Alasan melakukan hal ini seperti yang sudah dijelaskan sebelumnya, yaitu karena bangunan kastil dapat memberikan sumbangan yang besar bagi pertahanan dan penyerangan suatu pemain.

Berikut ini adalah penguraian elemen – elemen *greedy* untuk algoritma pembangunan kastil dengan cepat:

1. Himpunan kandidat, C.
   = Himpunan strategi.
2. Himpunan solusi, S.
   = Himpunan strategi untuk menang.
3. Fungsi seleksi.
   = Pilih strategi untuk membuat kastil secepat mungkin.
4. Fungsi kelayakan
   = Memeriksa apakah suatu aturan AI dapat membuat kastil dengan waktu yang lebih cepat daripada pemain manusia pada umumnya.
5. Fungsi obyektif
   = Waktu untuk membangun satu kastil seminimum mungkin atau lebih cepat daripada pemain pada umumnya.

Untuk dapat membangun kastil, seorang pemain harus dapat mencapai *castle age*. Pada umumnya permainan

dimulai dari *dark age* dan dibutuhkan peningkatan teknologi sebanyak dua tingkat untuk mencapai *castle age*. Agar dapat mencapai level teknologi yang dibutuhkan, maka pemain harus menentukan jumlah penduduk yang digunakan untuk mengumpulkan sumber daya yang dibutuhkan dengan tepat. Untuk AI yang akan membangun kastil dengan cepat, berikut ini adalah potongan program untuk AI agar dapat melanjutkan level teknologi dari *dark age* ke *feudal age* yang merupakan level teknologi kedua dengan cepat:

```
; Phase 0 - initial dark age food
gathering
(defrule
(goal GOAL-PHASE 0)
=>
(set-strategic-number sn-percent-
civilian-gatherers 65)
(set-strategic-number sn-percent-
civilian-builders 35)
(set-strategic-number sn-percent-
civilian-explorers 0)
(set-strategic-number sn-cap-civilian-
gatherers NUM-TOTAL-POPULATION)
(set-strategic-number sn-cap-civilian-
builders 1)
(set-strategic-number sn-cap-civilian-
explorers 0)
(set-strategic-number sn-food-gatherer-
percentage 100)
(set-strategic-number sn-wood-gatherer-
percentage 0)
(set-strategic-number sn-gold-gatherer-
percentage 0)
(set-strategic-number sn-stone-gatherer-
percentage 0)
)
```

Potongan program ini membuat aturan untuk AI agar dapat mencapai level teknologi berikutnya dengan cepat, yaitu dengan mengalokasikan semua penduduk untuk mendapatkan sumber daya makanan, dan persentase penduduk yang bertugas untuk membuat bangunan adalah 35% dari jumlah penduduk yang dimiliki, sedangkan yang bertugas untuk mengambil sumber daya adalah 65%.

Jumlah makanan yang dibutuhkan untuk mencapai *feudal age* berjumlah 500 makanan. Jika AI sudah mencapai angka tersebut, maka ia akan segera meningkatkan level teknologi.

Ketika sudah mencapai *feudal age* maka ia akan melanjutkan untuk dapat mencapai *castle age* agar dapat membangun kastil. Berikut ini adalah aturan ai dalam hal pengalokasian penduduk ketika berada di *dark age* dan *feudal age* untuk pengumpulan sumber daya kayu dan makanan:

```
; Phase 1 - dark age food and wood
gathering
(defrule
(goal GOAL-PHASE 1)
=>
(set-strategic-number sn-percent-
civilian-gatherers 65)
(set-strategic-number sn-percent-
civilian-builders 35)
(set-strategic-number sn-percent-
civilian-explorers 0)
(set-strategic-number sn-cap-civilian-
gatherers NUM-TOTAL-POPULATION)
(set-strategic-number sn-cap-civilian-
builders 2)
(set-strategic-number sn-cap-civilian-
explorers 0)
(set-strategic-number sn-food-gatherer-
percentage PERCENT-FOOD-GATHERER)
(set-strategic-number sn-wood-gatherer-
percentage PERCENT-WOOD-GATHERER)
(set-strategic-number sn-gold-gatherer-
percentage 0)
(set-strategic-number sn-stone-gatherer-
percentage 0)
)
```

Pada aturan ini, penduduk dialokasikan untuk fokus pada pencarian sumber daya makanan dan kayu. Berikut definisi konstanta yang dipakai untuk kode diatas:

```
(defconst PERCENT-FOOD-GATHERER 60)
(defconst PERCENT-WOOD-GATHERER 40)
```

Ketika sampai pada *castle age* maka pengalokasian penduduk yang bertugas untuk mencari sumber daya diubah. Sebuah kastil memerlukan sumber daya batu sebanyak 800 buah, sehingga untuk membangun kastil tersebut dengan cepat maka cara terbaik dan *straightforward* adalah dengan mengalokasikan sebanyak mungkin penduduk yang dimiliki untuk mengambil sumber daya batu. Berikut ini aturan yang menunjukannya:

```
; Phase 3 - stone gathering
(defrule
(goal GOAL-PHASE 3)
=>
(set-strategic-number sn-percent-civilian-gatherers
65)
(set-strategic-number sn-percent-civilian-builders 35)
(set-strategic-number sn-percent-civilian-explorers 0)
(set-strategic-number sn-cap-civilian-gatherers
NUM-TOTAL-POPULATION)
(set-strategic-number sn-cap-civilian-builders 2)
(set-strategic-number sn-cap-civilian-explorers 0)
(set-strategic-number sn-food-gatherer-percentage 0)
(set-strategic-number sn-wood-gatherer-percentage
0)
(set-strategic-number sn-gold-gatherer-percentage 0)
(set-strategic-number sn-stone-gatherer-percentage
100)
)
```

Pada aturan ini ditunjukan bahwa ketika sedang bertransisi ke *castle age* maka penduduk dialokasikan sebanyak mungkin untuk mengambil batu agar dapat membangun castle dengan cepat. Pada aturan diatas dapat

dilihat bahwa penduduk yang ditugaskan untuk mengambil batu adalah 100%. Ketika sumber daya yang diperlukan sudah dipenuhi dan *castle age* sudah tercapai, maka hal selanjutnya yang harus dilakukan adalah membuat kastil tersebut. Pendekatan yang paling cepat dan sederhana untuk membuat kastil tersebut dengan cepat adalah mengalokasikan sebanyak mungkin penduduk yang dipunyai untuk membangun sebuah kastil. Berikut ini aturan yang menunjukannya:

```
; Phase 4 - castle building
(defrule
(goal GOAL-PHASE 4)
=>
(set-strategic-number sn-percent-
civilian-gatherers 5)
(set-strategic-number sn-percent-
civilian-builders 95)
(set-strategic-number sn-percent-
civilian-explorers 0)
(set-strategic-number sn-cap-civilian-
gatherers 0)
(set-strategic-number sn-cap-civilian-
builders NUM-TOTAL-POPULATION)
(set-strategic-number sn-cap-civilian-
explorers 0)
(set-strategic-number sn-food-gatherer-
percentage 100)
(set-strategic-number sn-wood-gatherer-
percentage 0)
(set-strategic-number sn-gold-gatherer-
percentage 0)
(set-strategic-number sn-stone-gatherer-
percentage 0)
)
```

Waktu rata – rata yang dibutuhkan oleh program AI ini agar dapat mencapai tujuan, yaitu membangun kastil adalah 20 menit. Waktu tersebut cukup cepat bila dibandingkan dengan permainan yang dilakukan pemain biasa.

Algoritma *greedy* dengan membangun kastil dengan cepat ini tentu saja bukan solusi yang merupakan solusi optimal global karena algoritma ini memiliki kelemahan – kelemahan. Salah satu kelemahan yang dimiliki algoritma ini adalah tidak memperhitungkan akan adanya serangan dari pemain lawan, sehingga ketika pemain lawan melakukan penyerangan maka pembangunan akan terhambat dan mungkin saja AI ini dapat kalah dari permainan karena tidak adanya kekuatan pertahanan.

3.3. *Ai yang Menggunakan Algoritma Greedy dengan Waktu paling Minimaluntuk Membangun Wonder*

Algoritma *greedy* ini dapat dibilang lebih baik daripada sebelumnya, karena tujuan yang ingin dicapai oleh AI adalah membangun *wonder* secepat mungkin. Bangunan *wonder* ini adalah salah satu kunci kemenangan dalam permainan, karena jika seorang pemain dapat membuatnya maka akan ada waktu penghitungan mundur sampai pemain menang secara otomatis dengan mempertahankan *wonder*-nya dari serangan musuh – musuh. Strategi ini memungkinkan pemain untuk menang tanpa harus menaklukan semua musuh yang ada, yang mungkin akan memakan waktu yang lebih lama karena pertahanan musuh yang cukup kuat.

Berikut ini adalah penguraian elemen – elemen *greedy* untuk algoritma pembangunan *wonder* dengan cepat:

1. Himpunan kandidat, C.
   = Himpunan strategi.
2. Himpunan solusi, S.
   = Himpunan strategi untuk menang.
3. Fungsi seleksi.
   = Pilih strategi untuk membuat *wonder* secepat mungkin.
4. Fungsi kelayakan
   = Memeriksa apakah suatu aturan AI dapat membuat *wonder* dengan waktu yang lebih cepat daripada pemain manusia pada umumnya.
5. Fungsi obyektif
   = Waktu untuk membangun satu *wonder* seminimum mungkin atau lebih cepat daripada pemain pada umumnya.

Strategi pembangunan *wonder* disini hampir sama dengan strategi pembangunan kastil sebelumnya, yaitu secepat mungkin untuk mencapai level teknologi yang dibutuhkan yaitu *Imperial age* dan kemudian membangun bangunan *wonder* dengan mengalokasikan sebagian besar penduduk untuk mengerjakan bangunan tersebut dengan lebih cepat. Hal ini ditunjukan pada aturan berikut:

```
;------- BUILD WONDER
(defrule
(can-build wonder)
(not (goal 1 50))
=>
(set-goal 1 50)
(set-strategic-number sn-cap-civilian-
explorers 0)
(set-strategic-number sn-food-gatherer-
percentage 0)
(set-strategic-number sn-wood-gatherer-
percentage 0)
(set-strategic-number sn-gold-gatherer-
percentage 0)
(set-strategic-number sn-stone-gatherer-
percentage 0)
(set-strategic-number sn-retask-gather-
amount 0)
(set-strategic-number sn-max-retask-
gather-amount 0)
(set-strategic-number sn-use-by-type-max-
gathering 0)
(set-strategic-number sn-cap-civilian-
builders 100)
(set-strategic-number sn-percent-
civilian-builders 100)
(set-strategic-number sn-percent-
civilian-gatherers 0)
(set-strategic-number sn-random-
placement-factor 0)
)
```

Waktu rata – rata yang dibutuhkan untuk mencapai pembangunan *wonder* ini adalah sekitar 30 menit dan merupakan waktu yang cukup cepat untuk membangun

sebuah *wonder*.

Kelemahan utama dari algoritma ini pun sama dengan algoritma *greedy* dengan membangun kastil dengan cepat, yaitu pertahanan yang lemah. Untuk itu harus ditambahkan dengan pembuatan pasukan untuk bertahan seminimal mungkin, karena dengan melakukan hal tersebut akan memperlambat waktu untuk pembuatan *wonder*.

REFERENSI

[1] http://aok.heaven*game*s.com/blacksmith/showfile.php?fileid=5414 8 Desember 2011.

[2] http://community.topcoder.com/tc?module=Static&d1=tutorials&d2=*greedy*Alg 8 Desember 2011.

[3] http://en.wikipedia.org/wiki/PC_*game* 8 Desember 2011.

[4] http://id.wikipedia.org/wiki/Age_of_Empires_II:_The_Age_of_Kings 8 Desember 2011.

[5] http://id.wikipedia.org/wiki/Kecerdasan_buatan 8 Desember 2011.

[6] Munir, Rinaldi, "Diktat Kuliah IF3051 Strategi Algoritma". Institut Teknologi Bandung. 2009.

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 Desember 2011

Arie Pratama Sutiono/13509007