Penerapan Algoritma *Greedy* pada Intelegensia Buatan untuk *Transfer* Pemain dalam Permainan Simulasi Sepakbola

A. Thoriq Abrowi Bastari - 13508025
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
if18025@students.if.itb.ac.id

***Abstrak*—Makalah ini akan membahas salah satu penerapan algoritma *greedy* untuk menyelesaikan masalah pada suatu permainan komputer. Masalah yang coba dipecahkan adalah intelegensia buatan untuk *transfer* pemain pada salah satu permainan simulasi sepakbola, yaitu Football Manager. Algoritma ini mungkin bukan merupakan yang terbaik dalam memecahkan masalah ini, namun salah satu yang paling sederhana, mudah diterapkan, dan cukup efektif. Dalam penerapannya, penulis mencoba memperhitungkan beberapa atribut yang dapat mempengaruhi kesuksesan pembelian pemain. Tujuan yang diharapkan adalah agar pemain-pemain yang dibeli adalah yang terbaik, sesuai kebutuhan, dan sesuai dengan anggaran yang dialokasikan.**

***Kata kunci*—algoritma *greedy*, intelegensia buatan, Football Manager, *transfer* pemain**

I. PENDAHULUAN

Seiring dengan perkembangan teknologi, terutama teknologi informasi, permainan komputer semakin populer di seluruh kalangan. Hal itu melahirkan semakin banyak ide-ide baru dan kreatif yang memperkaya jenis-jenis/variasi permainan komputer yang ditawarkan. Salah satu buah ide itu adalah permainan simulasi olahraga sepakbola pada komputer. Pada makalah ini penulis akan membahas salah satu permainan simulasi tersebut, lebih tepatnya adalah permainan simulasi manajer sepakbola yang sangat populer, yaitu Football Manager.

Permainan ini mensimulasikan keterlibatan seorang manajer sepakbola dalam olahraga sepakbola itu sendiri. Pemain dapat merasakan bagaimana menjadi seorang manajer sepakbola, mengatur strategi tim, berhubungan dengan jurnalis-jurnalis pengincar berita, pemain sepakbola, suporter, dan jajaran direksi juga staf-staf dari suatu klub sepakbola. Luasnya lingkup dan banyaknya fitur-fitur yang ditawarkan membuat permainan simulasi ini terasa nyata dan menarik bagi banyak orang yang memang menggemari sepakbola, terutama bagi yang senang mengamati baik dari segi strategi maupun manajemennya.

Gambar 1. Salah satu antarmuka permainan Football Manager

Hampir dalam setiap permainan terdapat intelegensia buatan yang digunakan, begitu pula pada permainan Football Manager. Banyaknya aspek yang harus ditangani oleh manajer tim-tim lain yang tidak kita mainkan (dalam konteks ini intelegensia buatan) membuat penulis tertarik untuk mengamati cara kerja/algoritma yang digunakan. Salah satu aspek yang paling menarik adalah manajemen *transfer*/pembelian pemain yang dilakukan oleh tim-tim lain (intelegensia buatan). Berdasarkan pemikiran dan pengamatan penulis, algoritma sederhana dan cukup efektif yang dapat diterapkan pada aspek tersebut adalah algoritma *greedy*.

II. TEORI DASAR

Dalam persoalan optimasi, algoritma *greedy* adalah salah satu algoritma yang paling populer digunakan, bahkan mungkin yang terpopuler dibanding algoritma-algoritma sejenis. Pembelian pemain dalam permainan simulasi sepakbola ini juga merupakan permasalahan optimasi sehingga algoritma *greedy* dapat diterapkan di sini.

Ide utama algoritma *greedy* adalah "*take what you can get now*". Berarti pada setiap langkah/tahapan, algoritma

ini akan memilih keputusan yang dianggap terbaik pada saat itu tanpa mempedulikan apa yang akan terjadi di langkah-langkah selanjutnya. Dengan begitu, diharapkan keputusan-keputusan yang diambil pada setiap langkah (nilai maksimum lokal) akan membentuk rangkaian keputusan terbaik (nilai maksimum global), atau setidaknya mendekati terbaik. Itu lah penyebab mengapa algoritma ini disebut *greedy*, yang berarti rakus atau tamak.

Pada algoritma *greedy*, terdapat lima elemen utama, yaitu:

1. Himpunan kandidat (C)
   Berisi elemen-elemen kandidat yang nantinya akan membentuk solusi.

2. Himpunan solusi (S)
   Berisi kandidat-kandidat yang terpilih sebagai solusi permasalahan.

3. Fungsi seleksi
   Fungsi untuk memilih kandidat yang paling memungkinkan mencapai solusi optimal. Kandidat yang sudah dipilih pada suatu langkah, tidak pernah dipertimbangkan lagi pada langkah selanjutnya.

4. Fungsi kelayakan
   Memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak, yakni kandidat tersebut bersama-sama dengan himpunan solusi yang sudah terbentuk tidak melanggar kendala (*constraint*) yang ada. Kandidat yang layak dimasukkan ke dalam himpunan solusi, sedangkan kandidat yang tidak layak, dibuang dan tidak pernah dipertimbangkan lagi.

5. Fungsi obyektif
   Fungsi yang memaksimumkan atau meminimumkan nilai solusi.

Secara umum algoritma *greedy* dapat dirumuskan sebagai berikut:

1. Inisialisasi S dengan nilai kosong.

2. Pilih sebuah kandidat dengan fungsi seleksi dari C.

3. Buang kandidat yang telah terpilih pada langkah ke-2 dari C.

4. Periksa apakah kandidat yang terpilih tersebut bersama dengan himpunan solusi membentuk solusi yang layak menggunakan fungsi kelayakan. Jika ya, masukkan kandidat tersebut ke dalam himpunan solusi. Jika tidak, jangan dimasukkan ke himpunan solusi dan tidak perlu dipertimbangkan lagi.

5. Periksa apakah himpunan solusi sudah lengkap, misalnya sudah memenuhi jumlah tertentu. Jika tidak, lakukan lagi dari langkah ke-2. Jika ya, hentikan penelusuran.

Berikut adalah *pseudo-code* dari algoritma *greedy*:

```
function greedy(input C:
himpunan_kandidat)→ himpunan_kandidat
{ Mengembalikan solusi dari persoalan
optimasi dengan algoritma greedy
Masukan: himpunan kandidat C
Keluaran: himpunan solusi yang bertipe
himpunan_kandidat
}
Deklarasi
x : kandidat
S : himpunan_kandidat
Algoritma:
S ← {} { inisialisasi S dengan kosong
}
while (not SOLUSI(S)) and (C ≠ {} ) do
    x ← SELEKSI(C) { pilih sebuah
kandidat dari C}
    C ← C - {x} { elemen himpunan
kandidat berkurang satu }
    if LAYAK(S ∪ {x}) then
        S ← S ∪ {x}
    endif
endwhile
{SOLUSI(S) or C = {} }
if SOLUSI(S) then
    return S
else
    write('tidak ada solusi')
endif
```

III. PENERAPAN ALGORITMA *GREEDY*

A. Penjelasan Umum Algoritma

Penerapan algoritma *greedy* pada intelegensia buatan pembelian pemain di permainan Football Manager ini sebenarnya tidak sesederhana yang dibayangkan. Kesulitan utama adalah dalam menentukan fungsi seleksi dan juga fungsi obyektif yang digunakan. Sesungguhnya nilai apa yang perlu dimaksimalkan/diminimumkan untuk melakukan pembelian pemain yang terbaik? Hal ini akan sangat relatif dan banyak variasinya. Namun berdasarkan pengalaman dan pengamatan penulis, *transfer* pemain yang baik adalah memperoleh pemain-pemain terbaik yang ada, sesuai kebutuhan tim, dan dengan biaya yang serendah-rendahnya. Untuk itu, dalam penerapan algoritma ini dibutuhkan masukan berupa pemain berposisi apa yang diperlukan dan juga biaya maksimal yang dialokasikan untuk pembelian pemain.

Agar memperoleh himpunan solusi yang lebih mendekati solusi terbaik akan dilakukan dua tahap

pembentukan solusi. Tahap pertama adalah tahap pembuatan *shortlist*. *Shortlist* adalah daftar pemain-pemain dari semua posisi yang telah diseleksi oleh fungsi seleksi *shortlist*. Tujuan dari tahap ini adalah untuk mempersempit lingkup dan membatasi jumlah pemain yang akan ditawar/dibeli. Untuk membuat *shortlist* ini akan dijalankan algoritma *greedy* beberapa kali sesuai dengan jumlah posisi berbeda yang dibutuhkan. Himpunan-himpunan solusi dari setiap penelusuran itu lalu akan digabungkan untuk membentuk *shortlist*. Misalnya suatu tim membutuhkan pemain-pemain berposisi bek tengah, penyerang, dan sayap kanan, maka akan dilakukan tiga kali pembentukan himpunan solusi, yang masing-masing himpunan solusi itu hanya mengandung pemain yang bermain di salah satu dari tiga posisi yang diminta. Untuk setiap posisi hanya akan diambil lima pemain.

Setelah *shortlist* terbentuk, barulah dilakukan tahap yang kedua, yaitu penawaran pemain. Tahap ini juga diimplementasikan menggunakan algoritma *greedy*. Pemain yang menjadi hasil dari seleksi pada himpunan kandidat (*shortlist*) akan langsung diberi penawaran untuk *transfer*. Apabila penawaran diterima dan pembelian sukses maka jumlah uang yang dipakai akan dikurangi dari total uang yang dialokasikan, selain itu posisi dari pemain tersebut akan dibuang dari daftar posisi yang dicari. Sementara apabila penawaran ditolak akan dilakukan seleksi lagi. Kandidat yang sudah diseleksi dibuang dari *shortlist* dan tidak dipertimbangkan lagi.

Penelusuran akan berakhir untuk tiga kondisi. Kondisi pertama adalah apabila uang yang dialokasikan telah habis. Kondisi kedua adalah jika seluruh posisi yang diincar pertama kali telah diisi oleh pemain yang baru dibeli, yang berarti daftar posisi yang dicari telah kosong. Kondisi terakhir adalah apabila himpunan kandidat telah kosong.

B. Penjelasan Fungsi Seleksi

Pemilihan fungsi seleksi adalah elemen terpenting algoritma *greedy*. Fungsi inilah yang akan menentukan tingkat keberhasilan algoritma. Semakin baik fungsi seleksi yang digunakan akan semakin dekat pula himpunan solusi yang diperoleh dengan solusi ideal. Di sini penulis memakai tiga jenis fungsi seleksi untuk menentukan nilai maksimal dari setiap pemain, yaitu:

1. *Greedy by Ability*
   Fungsi seleksi ini mempertimbangkan satu hal/atribut saja, yaitu Ability. Ability adalah suatu nilai yang merepresentasikan kemampuan rata-rata seorang pemain.

2. *Greedy by Potential*
   Fungsi seleksi ini memperhitungkan dua atribut, yaitu Potential Ability dan umur pemain. Nilainya dapat diperoleh dengan membagi Potential Ability dengan atribut Age. Semakin muda umurnya akan semakin baik nilainya, dan semakin besar Potential Ability-nya akan semakin baik pula nilainya.

3. *Greedy by Average*
   Fungsi seleksi ini mempertimbangkan dua atribut, yaitu Ability dan Value. Ability telah dijelaskan di atas, sementara Value adalah harga pasaran suatu pemain yang menunjukkan rata-rata biaya yang dibutuhkan untuk membelinya. Nilai yang diperhitungkan adalah Ability dibagi dengan Value.

C. Elemen-Elemen Algoritma Greedy

Terdapat dua set elemen dari algoritma *greedy* yang digunakan karena memang dilakukan dua tahap penggunaan algoritma *greedy* yang berbeda. Berikut adalah elemen-elemen dari algoritma *greedy* pada tahap pertama (pembentukan *shortlist*):

1. Himpunan kandidat (C): Seluruh pemain sepakbola pada basis data dengan posisi tertentu yang terkontrak oleh tim lain.

2. Himpunan solusi (S): Lima pemain yang telah melalui seleksi dan kelayakan.

3. Fungsi seleksi: Salah satu dari tiga fungsi yang dijelaskan bagian sebelumnya, yaitu *greedy by ability*, *greedy by potential*, atau *greedy by average*.

4. Fungsi kelayakan: Memeriksa apakah pemain yang telah terpilih dari hasil seleksi memiliki nilai Value di atas batas alokasi biaya.

5. Fungsi obyektif: *Shortlist* yang dihasilkan berisi pemain-pemain terbaik sesuai fungsi seleksi masing-masing

Sementara elemen-elemen dari algoritma *greedy* pada tahap kedua adalah sebagai berikut:

1. Himpunan kandidat (C): *Shortlist* yang dihasilkan di tahap sebelumnya.

2. Himpunan solusi (S): Pemain-pemain yang berhasil dibeli.

3. Fungsi seleksi: Salah satu dari tiga fungsi yang dijelaskan bagian sebelumnya, yaitu *greedy by ability*, *greedy by potential*, atau *greedy by average*.

4. Fungsi kelayakan: Memeriksa apakah pemain yang telah terpilih dari hasil seleksi memiliki nilai Value di atas batas alokasi biaya. Apabila ya, dicek lagi apakah tawaran pembelian yang dilakukan sukses.

5. Fungsi obyektif: Pemain-pemain yang dibeli adalah pemain-pemain terbaik sesuai fungsi

seleksi masing-masing

D. Pseudo-Code

Untuk lebih bisa memahami algoritma *greedy* yang diterapkan, disediakan kode yang ditulis dalam bahasa C++ sebagai berikut:

Header fungsi-fungsi seleksi:

```
Player maxAbility(vector<Player>
candidates);
//Mengambil nilai maksimal dari kandidat
pemain berdasarkan Ability

Player maxPotential(vector<Player>
candidates);
//Mengambil nilai maksimal dari kandidat
pemain berdasarkan Potential/Age

Player maxAverage(vector<Player>
candidates);
//Mengambil nilai maksimal dari kandidat
pemain berdasarkan Ability/Value
```

Algoritma yang digunakan untuk mencari nilai maksimum dari setiap fungsi adalah algoritma *brute force* biasa.

Membentuk shortlist:

```
vector<Player> getShortlist(vector<Player>
playerCandidates, int greedyMode, int
budget) {
//Membentuk shortlist
	vector<Player> shortlist;
	Player player;
	int count=0;

	while(!playerCandidates.empty() &&
count<5) {
		switch(greedyMode) {
			case 0:
				player =
maxAbility(playerCandidates);
				break;
			case 1:
				player =
maxPotential(playerCandidates);
				break;
			case 2:
				player =
maxAverage(playerCandidates);
		}

		if(player.getValue<=budget)
		{
//cek apabila biaya yang dialokasikan
kurang
		shortlist.push_back(player);
			count++;
		}

//membuang pemain yang telah diambil dari
himpunan kandidat pemain
	playerCandidates.remove(player);
	}
	return shortlist;
}
```

Melakukan penawaran/*transfer* pemain:

```
void playerTransfer(vector<Player>
playerDatabase, int greedyMode, int budget,
vector<int> pos) {
	vector<Player> shortlist;

	for(int i = 0; i<pos.size(); i++) {
//memanggil fungsi getShortlist untuk
mendapatkan shortlist pemain-pemain tiap
posisi yang dibutuhkan

	shortlist.append(getShortlist(player
Database.pos(pos.get(i)), greedyMode,
budget));
	}

	while(!shortlist.empty() && budget>0
&& !pos.empty()) {
		switch(greedyMode) {
			case 0:
				player =
maxAbility(shortlist);
				break;
			case 1:
				player =
maxPotential(shortlist);
				break;
			case 2:
				player =
maxAverage(shortlist);
		}

		if(player.getValue()<=budget)
		{
//cek apabila biaya yang dialokasikan masih
cukup, apabila masih maka lakukan penawaran
			bool offerStatus =
makeOffer(player);

//apabila penawaran diterima dan transfer
berhasil, buang posisi dari vektor posisi
dan kurangi budget
			if(offerStatus==true) {
				budget-
=player.getValue();

	pos.remove(player.pos());
			}
		}

//membuang pemain yang telah diambil dari
himpunan kandidat pemain
	playerCandidates.remove(player);
	}
}
```

Kode yang ditampilkan di atas hanyalah merupakan rancangan kasar dari konsep perancangan yang dilakukan.

IV. ANALISIS

Meskipun penulis tidak mengimplementasikan

algoritma yang digunakan untuk menyelesaikan permasalahan dalam permainan atau simulasi dari permainan ini, namun sudah bisa dipastikan bahwa algoritma ini tidak selalu menghasilkan solusi yang terbaik.

Algoritma *greedy* memang umum dipakai untuk mencari solusi global yang mendekati solusi terbaik dengan beban kalkulasi yang relatif ringan sehingga kegagalan menemukan solusi paling baik bukan berarti algoritma *greedy* tidak cocok untuk diterapkan di sini.

Pada masalah *transfer* pemain pada permainan simulasi sepakbola seperti ini, solusi yang terbaik sangat sulit untuk dicari karena terlalu banyak faktor yang bisa menentukan kesuksesan/keberhasilan seorang pemain di suatu klub. Dengan begitu, kegagalan algoritma *greedy* dalam melakukan pembelian yang menurutnya terbaik tidak akan berakibat terlalu buruk bagi tim yang melakukan pembelian pemain tersebut.

V. KESIMPULAN

Kesimpulan yang dapat diambil dari makalah ini adalah sebagai berikut:

1. Algoritma *greedy* terbukti dapat digunakan untuk menyelesaikan banyak jenis masalah, salah satunya adalah sebagai intelegensia buatan dalam pemecahan masalah *transfer* pemain pada permainan simulasi sepakbola.
2. Penerapan algoritma *greedy* dalam kasus ini tidak selalu dapat menemukan solusi yang terbaik, namun hampir bisa dipastikan kalau solusi yang didapat tersebut mendekati hasil yang paling ideal.

Pengembangan dari algoritma ini bisa dilakukan dengan menambahkan/menggantinya dengan algoritma program dinamis. Namun, dengan begitu kalkulasi yang dilakukan oleh program akan sangat berat dan sangat berpotensi untuk mengurangi kinerja program.

REFERENSI

Rinaldi Munir, *Diktat Kuliah IF2251 Strategi Algoritmik*, Program Studi Teknik Informatika ITB, 2005

http://www.sigames.com.

http://www.footballmanager.com.

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, 7 Desember 2010

A Thoriq Abrowi Bastari (13508025)