# Pemanfaatan Algoritma Greedy dalam Menentukan Pembelian Pemain di Permainan Football Manager

*Shandy 13516097*
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
*Shandy.gunawan@rocketmail.com*

***Abstract*—Algoritma *greedy* mampu digunakan di berbagai persoalan, termasuk permainan komputer. Algoritma *greedy* dapat digunakan untuk melakukan pemilihan pemain dari daftar pemain yang diincar dalam permainan Football Manager. Algoritma *greedy* menawarkan pendekatan-pendekatan yang berbeda dengan hasil yang berbeda pula sehingga memberikan opsi bagi pemain untuk mengelola kualitas dan anggaran pembelian.**

***Kata kunci*—greedy, permainan, pemilihan, pemain**

## I. PENDAHULUAN

Sepakbola adalah salah satu olahraga yang sangat digemari di dunia, termasuk di Indonesia. Sepakbola dimainkan oleh dua tim yang beranggotakan masing-masing 11 orang pemain di lapangan dengan menggunakan bola yang umumnya terbuat dari bahan kulit. Tujuan dari permainan sepakbola yakni untuk memasukkan bola ke gawang lawan atau yang biasa disebut mencetak gol sebanyak-banyaknya. Tim yang mencetak gol lebih banyak adalah tim yang memenangkan pertandingan. Jika kedua tim berkedudukan imbang hingga waktu berakhir, maka dapat dilakukan perpanjangan waktu yang dilanjutkan adu penalti, penalty, atau penggunaan gol tandang bergantung pada format penyelenggaraan kejuaraan. Lama permainan sepak bola normal adalah 2x45 menit, dengan istirahat 15 menit di antara kedua babak. Jika kedudukan berakhir imbang, maka diadakan perpanjangan waktu selama 2x15 menit. Selain itu, di akhir setiap babak, wasit umumnya menambahkan beberapa waktu tambahan untuk menggantikan waktu yang hilang akibat cedera atau pergantian pemain yang disebut sebagai *injury time*.

Pemain sepakbola di dalam tim mempunyai perannya masing-masing, seperti penjaga gawang berperan untuk menjaga gawang tim agar tidak kebobolan, penyerang yang berperan untuk mencetak gol ke gawang tim lawan. dll. Oleh karena itu, seorang pemain sepakbola dikatakan sebagai pemain yang bagus jika mampu melaksanakan tugasnya dengan baik. Akan tetapi, setiap tim pasti mempunyai satu atau lebih pemain yang tidak mampu melakukan tugasnya dengan baik atau tidak melakukan tugasnya sesuai ekspektasi. Hal ini tentunya menjadi permasalah bagi tim yang sudah membayar gaji pemainnya itu. Oleh karena itu, terciptalah sistem jual-beli pemain antar klub sepakbola atau yang biasa disebut sebagai transfer pemain. Transfer pemain umumnya dilakukan oleh para petinggi klub atau oleh pelatih sendiri. Transfer pemain biasanya dilakukan untuk memperbaiki kekurangan tim, meningkatkan finansial klub, atau mengusir pemain yang berperilakuan tidak baik atau tidak memenuhi ekspektasi. Transfer pemain merupakan interaksi resmi antara dua atau lebih klub sepakbola serta klub dengan pemain sehingga uang, kontrak, dan surat resmi diperlukan dalam melakukan transfer

*Gambar 1.1 Perjanjian transfer pemain*

Penggunaan uang dalam melakukan transfer tentu saja menyulitkan berbagai klub karena setiap klub sepakbola mempunyai kondisi finansial yang berbeda-beda. Oleh karena itu, setiap klub sepakbola menggunakan *scout* untuk mencari pemain atau anak-anak dengan bakat sepakbola yang mumpuni dan tidak membebankan klub secara finansial.

Proses pencarian bakat dan pembelian pemain tentunya tidak mudah, banyak faktor yang harus diperhatikan seperti usia, asal dan kewarganegaraan, performansi, harga, dll. Akan tetapi, performansi dan harga umumnya menjadi komponen yang sangat diperhatikan dalam membeli sebuah pemain. Setiap klub tentunya menginginkan pemain dengan performansi yang bagus dan berharga murah.

Pada makalah ini, penulis akan membahas proses pemilihan pemain menggunakan kedua komponen tersebut, yakni performansi dan harga untuk menentukan pemain yang layak untuk dibeli oleh suatu klub berdasarkan kondisi finansial. Proses yang digunakan menggunakan algoritma yang popular di ilmu komputer, yakni algoritma *greedy*.

## II. DASAR TEORI

### *A. Algoritma Exhaustive Search*

Algoritma *Exhaustive Search* adalah sebuah algoritma *problem-solving* yang sangat umum. Algoritma ini mengenumerasi semua kemungkinan kandidat solusi dan memeriksa apakah masing-masing kandidat solusi memenuhi

persyaratan solusi dari persoalan yang diajukan. Sebagai contoh, algoritma *Exhaustive Search* untuk mencari pembagi dari bilangan asli *n* akan mengenumerasi semua bilangan dari 1 hingga *n*. *Exhaustive Search* sangat mudah untuk diimplementasikan dan solusi pasti ditemukan jika persoalan mempunyai solusi. Akan tetapi, *Exhaustive search* membutuhkan sumber daya yang sangat besar, berupa waktu dan memori untuk mengenumerasi dan mengevaluasi semua kemungkinan kandidat solusi. Oleh karena itu, para *programmer* umumnya mencari algoritma-algoritma alternatif dari algoritma *Exhaustive Search*.

### *B. Algoritma Greedy*

Kita seringkali menghadapi persoalan optimasi di kehidupan sehari-hari. Persoalan optimasi sendiri adalah persoalan yang diselesaikan dengan mencari solusi optimum dari persoalan tersebut, berupa maksimasi (*maximization*) atau minimasi (*minimization*). Pada kehidupan sehari-hari, contoh dari persoalan optimasi yakni memilih buah terberat dari kumpulan buah untuk dibeli, membeli roti tawar yang mempunyai tanggal ekspirasi lebih jauh dibandingkan roti tawar lainnya, dll.

Algoritma *greedy* adalah algoritma pemecahan persoalan optimasi. Algoritma *greedy* membentuk solusi pada setiap langkahnya dan menggunakan metode heuristik dengan menganggap solusi yang tersebut sebagai solusi optimal atau disebut juga sebagai optimum lokal (*local optimum*) dengan harapan bahwa optimum lokal tersebut akan mencapai ke optimum global (*global optimum*) sehingga algoritma greedy dapat ditulis dengan langkah berikut:

1. Mengambil pilihan terbaik (optimum lokal) yang dapat diperoleh pada setiap langkah tanpa memperhatikan konsekuensi ke depan.
2. Berharap bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global.

Algoritma *greedy* juga mempunyai elemen-elemen yang membantu terbentuknya optimum lokal pada setiap langkah. Elemen-elemen tersebut antara lain.

- Himpunan kandidat (C) yang berisi elemen-elemen kandidata yang digunakan sebagai solusi.
- Himpunan solusi (S) yang berisi kandidat-kandidat dari himpunan kandidat yang sudah terpilih sebagai solusi permasalahan.
- Fungsi seleksi untuk memilih kandidat yang paling memungkinkan untuk mencapai solusi optimal.
- Fungsi kelayakan yang memeriksa jika kandidat yang sedang diperiksa mampu memberikan solusi yang layak, yakni penambahkan kandidat ke dalam himpunan solusi tidak akan melanggar aturan yang ada.
- Fungsi objektif yang berfungsi untuk mengoptimasi solusi.

Dalam penggunaannya, algoritma *greedy* melibatkan pencarian sebuah himpunan bagian, *S*, dari himpunan kandidat, *C*; yang dalam ini, *S* harus memenuhi beberapa kriteria yang telah ditentukan oleh fungsi seleksi dan fungsi kelayakan, yaitu menyatakan suatu solusi dan *S* dioptimisasi oleh fungsi objektif.

Algoritma *greedy* tidak selalu memberikan solusi yang terbaik. Hal ini disebabkan oleh beberapa hal, antara lain.

- Algoritma *greedy* tidak beroperasi secara menyeluruh terhadap semua alternative solusi yang ada. Fungsi seleksi dan fungsi kelayakan akan mengabaikan kandidat solusi jika solusi diterima untuk fungsi seleksi dan jika solusi ditolak untuk fungsi kelayakan.
- Fungsi seleksi dapat berbeda atau tidak optimal sehingga turut menghasilkan solusi yang tidak optimal.

### *C. Penggunaan Exhaustive Search pada Persoalan Knapsack*

Persoalan Knapsack adalah salah satu persoalan optimasi yang terkenal. Persoalan Knapsack meminta kita untuk mencari solusi optimal jika terdapat benda-benda dengan berat dan nilai yang harus dimasukkan ke dalam suatu tempat yang hanya mampu menampung dengan kapasitas maksimal tertentu. Aturan dari peraturan Knapsack yakni jumlah berat benda-benda yang dimasukkan harus kurang dari atau sama dengan kapasitas tempat sembari mendapatkan nilai yang setinggi mungkin. Persoalan Knapsack dapat diselesaikan oleh algoritma *exhaustive search*.

Langkah-langkah algoritma *exhaustive search* dalam menemukan solusi persoalan Knapsack adalah sebagai berikut.

1. Enumerasikan semua himpunan bagian dari himpunan dengan *n* objek
2. Hitung toal keuntungan dari setiap himpunan bagian dari langkah 1
3. Pilih himpunan bagian yang memberikan total keuntungan terbesar

Solusi persoalan direpresentasikan dengan X = {x1, x2, …, xn} dengan xi = 1, jika objek ke-i dipilih dan xi = 0 jika tidak.

Sebagai contoh, persoalan Knapsack kapasitas sebesar 16.

| Objek | Bobot | Profit ($) |
|---|---|---|
| 1 | 2 | 20 |
| 2 | 5 | 30 |
| 3 | 10 | 50 |
| 4 | 5 | 10 |

*Gambar 2.1 Contoh penyelesaian persoalan Knapsack dengan algoritma Exhaustive Search*

Langkah-langkah pencarian solusi Knapsack secara *exhaustive search* dirangkum dalam tabel berikut ini.

| Himpunan Bagian | Total Bobot | Total keuntungan |
|---|---|---|
| {} | 0 | 0 |
| {1} | 2 | 20 |
| {2} | 5 | 30 |
| {3} | 10 | 50 |
| {4} | 5 | 10 |
| {1, 2} | 7 | 50 |
| {1, 3} | 12 | 70 |
| {1, 4} | 7 | 30 |
| **{2, 3}** | **15** | **80** |
| {2, 4} | 10 | 40 |
| {3, 4} | 15 | 60 |
| {1, 2, 3} | 17 | tidak layak |
| {1, 2, 4} | 12 | 60 |
| {1, 3, 4} | 17 | tidak layak |
| {2, 3, 4} | 20 | tidak layak |
| {1, 2, 3, 4} | 22 | tidak layak |

*Gambar 2.2 Contoh enumerasi exhaustive search*

Himpunan bagian objek yang memberikan keuntungan maksimum adalah {2,3} dengan total keuntungan adalah 80 sehingga solusi adalah X = {0, 1, 1, 0}. Banyaknya himpunan

bagian dari sebuah himpunan dengan $n$ elemen adalah $2^n$ dan waktu untuk menghitung total botok objek yang dipilih adalah O($n$) sehingga kompleksitas waktu dari algoritma *exhaustive search* untuk persoalan Knapsack adalah O($n.2^n$).

### *D. Penggunaan Algoritma Greedy pada Persoalan Knapsack*

Persoalan Knapsack dapat diselesaikan oleh algoritma *greedy* dengan berbagai pendekatan.

Pendekatan pertama yakni mengutamakan nilai dari benda-benda yang dimasukkan (*greedy by profit*). Pada setiap langkah, pilih objek yang mempunyai keuntungan terbesar. Pendekatan ini mencoba memaksimumkan keuntungan dengan memilih objek yang paling menguntungkan terlebih dahulu.

Pendekatan kedua yakni mengutamakan berat dari benda-benda yang dimasukkan (*greedy by weight*). Pada setiap langkah, pilih objek yang mempunyai berat teringan. Pendekatan ini mencoba memaksimumkan keuntungan dengan memasukkan sebanyak mungkin objek ke dalam tempat penampungan.

Pendekatan ketiga yakni mengutamakan densitas dari benda-benda yang dimasukkan (*greedy by density*). Pada setiap langkah, tempat penampungan diisi dengan objek yang memilih nilai/berat terbesar. Pendekatan ini mencoba memaksmumkan keuntungan dengan memilih objek yang mempunyai keuntungan per unit berat terbesar.

Perlu diingat, ketiga pendekatan tersebut tidak selalu menghasilkan solusi yang optimal. Contoh dari penyelesaian persoalan Knapsack dengan algoritma *greedy* sebagai berikut (kapasitas tempat sebesar 16).

| Properti objek | | | | *Greedy by* | | | Solusi |
|---|---|---|---|---|---|---|---|
| *i* | $w_i$ | $p_i$ | $p_i/w_i$ | *profit* | *weight* | *density* | Optimal |
| 1 | 6 | 12 | 2 | 0 | 1 | 0 | 0 |
| 2 | 5 | 15 | 3 | 1 | 1 | 1 | 1 |
| 3 | 10 | 50 | 5 | 1 | 0 | 1 | 1 |
| 4 | 5 | 10 | 2 | 0 | 1 | 0 | 0 |
| Total bobot | | | | 15 | 16 | 15 | 15 |
| Total keuntungan | | | | 65 | 37 | 65 | 65 |

*Gambar 2.3 Contoh pertama penyelesaian persoalan Knapsack dengan Algoritma Greedy*

Berdasarkan gambar 2.3, kita dapat menarik kesimpulan bahwa pendekatan ketiga, yakni *greedy by density* memberikan hasil yang terbaik dibandingkan pendekatan lainnya, bahkan pada gambar diatas solusi yang dihasilkan sama dengan solusi optimal. Akan tetapi, hal ini tidak menjamin bahwa pendekatan ini akan selalu memberikan solusi yang optimal seperti pada gambar dibawah ini (kapasitas tempat sebesar 100).

| Properti objek | | | | *Greedy by* | | | Solusi |
|---|---|---|---|---|---|---|---|
| *i* | $w_i$ | $p_i$ | $p_i/w_i$ | *profit* | *weight* | *density* | Optimal |
| 1 | 100 | 40 | 0,4 | 1 | 0 | 0 | 0 |
| 2 | 50 | 35 | 0,7 | 0 | 0 | 1 | 1 |
| 3 | 45 | 18 | 0,4 | 0 | 1 | 0 | 1 |
| 4 | 20 | 4 | 0,2 | 0 | 1 | 1 | 0 |
| 5 | 10 | 10 | 1,0 | 0 | 1 | 1 | 0 |
| 6 | 5 | 2 | 0,4 | 0 | 1 | 1 | 0 |
| Total bobot | | | | 100 | 80 | 85 | 100 |
| Total keuntungan | | | | 40 | 34 | 51 | 55 |

*Gambar 2,4 Contoh kedua penyelesaian persoalan Knapsack dengan Algoritma Greedy*

Berdasarkan gambar 2.4, ketiga pendekatan *greedy* tidak memberikan solusi yang optimal. Akan tetapi, pendekatan *greedy by density* tetap memberikan solusi yang terbaik dibandingkan kedua pendekatan lainnya.

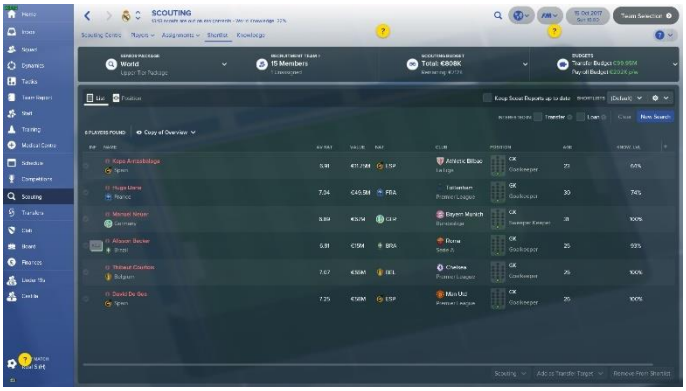
## III. Penggunaan Algoritma Greedy Dalam Menentukan Pembelian Pemain di Dalam Permainan Football Manager

### *A. Antar Muka Football Manager*

Football Manager (FM) adalah permainan simulasi sepakbola yang terkenal. Pemain FM berperan sebagai pelatih sebuah klub sepakbola di dalam permainan sehingga pemain FM turut mengatur taktik, latihan, serta transfer pemain. Edisi Football Manager yang digunakan yakni Football Manager 2018 (FM 2018) yang merupakan edisi terbaru saat tulisan ini ditulis. Antar muka FM terlihat cukup menyulitkan bagi pemula karena banyaknya informasi yang disampaikan. Akan tetapi, terdapat beberapa tombol bantuan yang membantu pemain FM pemula untuk beradaptasi untuk antar muka permainan. Berikut adalah contoh dari antar muka halaman utama dari FM.

*Gambar 3.1 Antar muka halaman utama FM*

Antar muka halaman utama tersebut menunjukkan berbagai informasi penting yang perlu diketahui, seperti pertandingan selanjutnya, klasemen liga, keuangan, catatan cedera, dan dukungan dari klub dan fans. *Toolbar* berwarna biru yang ada di pojok kiri layar dapat digunakan untuk menampilkan informasi-informasi tersebut secara spesifik. Khusus untuk tulisan ini, penulis hanya akan menampilkan antar muka halaman *scouting* yang berisi antar muka dari pemain-pemain yang diincar dan ingin dibeli oleh pemain FM.

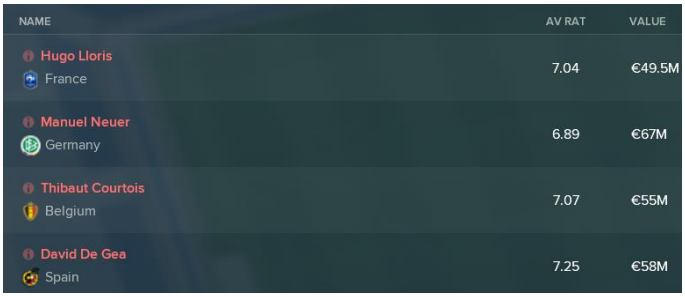


*Gambar 3.2 Antar Muka halaman Scouting FM*

Antar muka halaman *Scouting* diatas menampilkan informasi mengenai penjaga gawang yang ingin dibeli oleh pemain FM. Akan tetapi, kita hanya akan menggunakan beberapa informasi dari banyak informasi tersebut.

### *B. Elemen Algoritma Greedy*

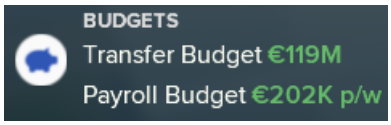
Salah satu kasus penggunaan algoritma *greedy* pada permainan ini yakni kita ingin membeli beberapa penjaga gawang untuk menggantikan para penjaga gawang utama tim yang tidak lagi bertanding dengan performa baik. Salah satu fitur dalam FM yakni kita mampu menyimpan daftar pemain yang ingin kita beli dinamakan dengan *shortlist*. Kita sudah membuat daftar penjaga gawang yang ingin dibeli sebagai berikut.




*Gambar 3.1 Shorlist penjaga gawang*

*Shortlist* diatas menunjukkan nama, nilai rata-rata performa, serta harga dari masing-masing penjaga gawang yang ingin dibeli oleh kita. Jika dilihat dengan seksama, persoalan pemilihan penjaga gawang diatas mirip dengan persoalan Knapsack yang sudah dibahas. Oleh karena itu, kita mampu menggunakan algoritma *greedy* untuk memilih para penjaga gawang yang akan dibeli.

Persoalan Knapsack membutuhkan informasi kapasitas yang bisa ditampung oleh tempat penampung (*container*). Dalam FM, *container* tersebut dapat kita analogikan dengan anggaran transfer (*transfer budget*) yang kita punyai di dalam permainan. Untuk kasus ini, kita mempunyai *transfer budget* dengan besar sebagai berikut.




*Gambar 3.2 Transfer budget yang dimiliki pemain*

*Payroll budget* adalah anggaran untuk membayar gaji para pemain. Akan tetapi, kita mengabaikan *payroll budget* untuk kasus ini.

Setelah mengetahui cara penyelesaian persoalan yakni dengan menggunakan solusi persoalan Knapsack. Kita mampu menentukan elemen-elemen dasar dari algoritma *greedy*. Elemen-elemen dasar algoritma *greedy* dalam FM dapat direpresentasikan sebagai berikut.

- Himpunan kandidat (C) yang berisi masing-masing penjaga gawang di dalam *shortlist*.
- Himpunan solusi (S) yang berisi masing-masing penjaga gawang yang akan dibeli.
- Fungsi seleksi yang masing-masing menggunakan pendekatan pada persoalan Knapsack. Pada kasus ini, fungsi seleksi berupa *greedy by Rating*, *greedy by value*, dan *greedy by density*.
- Fungsi kelayakan yang memeriksa apakah pemain yang telah terpilih dari seleksi memiliki nilai *value* yang melebihi *transfer budget*.
- Fungsi objektif yang mengoptimasi pembelian penjaga gawang, yakni mendapatkan penjaga gawang dengan performa baik sebanyak-banyaknya.

Sama seperti proses penyelesaian pada persoalan Knapsack. Kita juga akan menerapkan beberapa pendekatan untuk mencari solusi terbaik, serta mencari solusi optimum dengan menggunakan *Exhaustive Search*. Akan tetapi, kita perlu melakukan tabulasi terlebih dahulu terhadap informasi penjaga gawang yang kita punya. Rata-rata performa dan *value* penjaga gawang direpresentasikan dengan simbol *R* dan *V*.

| **Penjaga Gawang** | | | | |
|---|---|---|---|---|
| *i* | *Nama* | $R_i$ | $V_i$ | $R_i/V_i$ |
| 1 | Lloris | 7.04 | 49.5 | 0.142 |
| 2 | Neuer | 6.89 | 67 | 0.102 |
| 3 | Courtois | 7.07 | 55 | 0.129 |
| 4 | De Gea | 7.25 | 58 | 0.125 |

*Tabel 3.1 Tabulasi informasi penjaga gawang*

### *C. Penerapan Algoritma Greedy dengan greedy by rating*

Pendekatan *greedy* pertama yakni *greedy by rating*. Pendekatan ini mengutamakan performa penjaga gawang sebagai komponen pemilihan. Penjaga gawang dengan performa yang baik akan memiliki kemungkinan lebih besar untuk dipilih. Berikut adalah tabel informasi penjaga gawang jika diurutkan berdasarkan *R*.

| **Penjaga Gawang** | | | | |
|---|---|---|---|---|
| *i* | *Nama* | $R_i$ | $V_i$ | $R_i/V_i$ |
| 4 | De Gea | 7.25 | 58 | 0.125 |
| 3 | Courtois | 7.07 | 55 | 0.129 |
| 1 | Lloris | 7.04 | 49.5 | 0.142 |
| 2 | Neuer | 6.89 | 67 | 0.102 |

*Tabel 3.2 Tabel informasi penjaga gawang terurut berdasarkan R*

Proses pemilihan penjaga gawang dimulai dengan kondisi awal berupa himpunan solusi (S) kosong dan himpunan kandidat (C) berisi nama penjaga gawang. Kedua himpunan tersebut

dapat ditulis dengan S = {}, C = {Lloris, Neuer, Courtois, De Gea}.

De Gea adalah penjaga gawang dengan performa terbaik. Oleh karena itu, kita memasukkan De Gea ke dalam himpunan solusi sehingga S = {De Gea} dan C = {Courtois, Lloris, Neuer}. *Transfer budget* kita berkurang dari 119 menjadi 61.

Courtois adalah penjaga gawang dengan performa terbaik kedua setelah De Gea dan mempunyai *value* sebesar 55. *Transfer budget* kita masih memungkinkan untuk membeli Courtois sehingga kita memasukkan Courtois ke dalam himpunan solusi. Oleh karena itu, S = {De Gea, Courtois}, C = {Lloris, Neuer} dan *transfer budget* kita menjadi 6.

Kemudian, kita kembali menyeleksi penjaga gawang yang ada di himpunan kandidat. Akan tetapi, *Transfer budget* kita sudah tidak memungkinkan untuk membeli penjaga gawang lain sehingga nilai S = {De Gea, Courtois} menjadi himpunan solusi akhir kita dan dapat direpresentasikan sebagai berikut.

| **Penjaga Gawang** | | | | | |
|---|---|---|---|---|---|
| *i* | *Nama* | $R_i$ | $V_i$ | $R_i/V_i$ | *Greedy by Rating* |
| 1 | Lloris | 7.04 | 49.5 | 0.142 | 0 |
| 2 | Neuer | 6.89 | 67 | 0.102 | 0 |
| 3 | Courtois | 7.07 | 55 | 0.129 | 1 |
| 4 | De Gea | 7.25 | 58 | 0.125 | 1 |
| Rata-rata *rating* | | | | | 7.16 |
| Total biaya | | | | | 113 |

*Tabel 3.3 Pemilihan penjaga gawang dengan greedy by rating*

### *D. Penerapan Algoritma Greedy dengan greedy by value*

Pendekatan *greedy* kedua yakni *greedy by value*. Pendekatan ini mengutamakan harga pemain sebagai komponen pemilihan. Penjaga gawang dengan harga yang lebih mahal atau *value* yang lebih tinggi akan kemungkinan lebih besar untuk dipilih. Berikut adalah tabel informasi penjaga gawang jika diurutkan berdasarkan *V*.

| **Penjaga Gawang** | | | | |
|---|---|---|---|---|
| *i* | *Nama* | $R_i$ | $V_i$ | $R_i/V_i$ |
| 2 | Neuer | 6.89 | 67 | 0.102 |
| 4 | De Gea | 7.25 | 58 | 0.125 |
| 3 | Courtois | 7.07 | 55 | 0.129 |
| 1 | Lloris | 7.04 | 49.5 | 0.142 |

*Tabel 3.4 Tabel informasi penjaga gawang terurut berdasarkan V*

Proses pemilihan penjaga gawang dimulai dengan kondisi awal berupa himpunan solusi (S) kosong dan himpunan kandidat (C) berisi nama penjaga gawang. Kedua himpunan tersebut dapat ditulis dengan S = {}, C = {Lloris, Neuer, Courtois, De Gea}.

Neuer adalah penjaga gawang dengan *value* tertinggi sehingga Neuer dimasukkan ke dalam himpunan solusi. S = {Neuer}, C = {Lloris, Courtois, De Gea}, dan *transfer budget* menjadi 52.

De Gea menjadi penjaga gawang dengan *value* tertinggi setelah Neuer diikuti dengan Courtois. Akan tetapi, *value* kedua penjaga gawang tersebut melebihi *transfer budget* kita. Oleh karena itu, kita menghapus kedua penjaga gawang tersebut dari himpunan kandidat. C = {Neuer}, S = {Lloris}, dan *transfer budget* = 52.

Lloris menjadi pilihan penjaga gawang terakhir dan mempunyai *value* yang lebih rendah dibandingkan *transfer budget* yang kita miliki. Oleh karena itu, kita memasukkannya ke dalam himpunan solusi. S = {Neuer, Lloris}, C = {}, *transfer budget* menjadi 2.5. Himpunan Kandidat C menjadi kosong sehingga algoritma *greedy* berakhir dan S menjadi solusi akhir.

| **Penjaga Gawang** | | | | | |
|---|---|---|---|---|---|
| *i* | *Nama* | $R_i$ | $V_i$ | $R_i/V_i$ | *Greedy by Value* |
| 1 | Lloris | 7.04 | 49.5 | 0.142 | 1 |
| 2 | Neuer | 6.89 | 67 | 0.102 | 1 |
| 3 | Courtois | 7.07 | 55 | 0.129 | 0 |
| 4 | De Gea | 7.25 | 58 | 0.125 | 0 |
| Rata-rata *rating* | | | | | 6.97 |
| Total biaya | | | | | 116.5 |

*Tabel 3.5 Pemilihan penjaga gawang dengan greedy by value*

### *E. Penerapan Algoritma Greedy dengan greedy by density*

Pendekatan *greedy* yang terakhir adalah pendekatan *greedy by density*. *Density* adalah nilai *rating* dibandingkan dengan nilai *value* masing-masing penjaga gawang. Nilai *density* tersebut dijadikan komponen pemilihan. Berikut adalah urutan penjaga gawang jika diurutkan berdasarkan *density*.

| **Penjaga Gawang** | | | | |
|---|---|---|---|---|
| *i* | *Nama* | $R_i$ | $V_i$ | $R_i/V_i$ |
| 1 | Lloris | 7.04 | 49.5 | 0.142 |
| 3 | Courtois | 7.07 | 55 | 0.129 |
| 4 | De Gea | 7.25 | 58 | 0.125 |
| 2 | Neuer | 6.89 | 67 | 0.102 |

*Tabel 3.6 Tabel informasi penjaga gawang terurut berdasarkan density ($R_i/V_i$)*

Proses pemilihan penjaga gawang dimulai dengan kondisi awal berupa himpunan solusi (S) kosong dan himpunan kandidat (C) berisi nama penjaga gawang. Kedua himpunan tersebut dapat ditulis dengan S = {}, C = {Lloris, Neuer, Courtois, De Gea}.

Lloris adalah penjaga gawang dengan *density* tertinggi sehingga ia terlebih dahulu dimasukkan ke dalam himpunan solusi dan *transfer budget* berkurang sejumlah nilai *value* Lloris. S = {Lloris}, C = {Neuer, Courtois, De Gea}, dan *transfer budget* sebesar 69.5.

Courtois menjadi penjaga gawang dengan *density* tertinggi berikutnya. Selain itu, besar *transfer budget* masih cukup untuk membeli Courtois sehingga Courtois dimasukkan ke dalam himpunan solusi dengan *transfer budget* berkurang sejumlah nilai *value* Courtois. S = {Lloris, Courtois}, C = {De Gea, Neuer}, dan *transfer budget* sebesar 14.5. Nilai *transfer budget* tidak lagi memungkinkan untuk membeli penjaga gawang lain sehingga algoritma *greedy* dihentikan dengan S menjadi solusi.

| **Penjaga Gawang** | | | | | |
|---|---|---|---|---|---|
| *i* | *Nama* | $R_i$ | $V_i$ | $R_i/V_i$ | *Greedy by Density* |
| 1 | Lloris | 7.04 | 49.5 | 0.142 | 1 |
| 2 | Neuer | 6.89 | 67 | 0.102 | 0 |
| 3 | Courtois | 7.07 | 55 | 0.129 | 1 |
| 4 | De Gea | 7.25 | 58 | 0.125 | 0 |

| | |
|---|---|
| Rata-rata *rating* | 7.06 |
| Total biaya | 104.5 |

*Tabel 3.7 Pemilihan penjaga gawang dengan greedy by value*

## IV. ANALISIS

Ketiga pendekatan algoritma *greedy* menghasilkan pilihan penjaga gawang yang berbeda-beda. Hal ini menyebabkan nilai rata-rata *rating* dan total biaya untuk masing-masing pilihan juga turut berbeda sehingga diperlukan analisis lebih lanjut untuk menentukan pilihan penjaga gawang yang terbaik. Berikut adalah tabel perbandingan hasil dari masing-masing pendekatan algoritma *greedy* terhadap pemilihan penjaga gawang.

| *Penjaga Gawang* | | | | | *Greedy by* | | | *Solusi* |
|---|---|---|---|---|---|---|---|---|
| *i* | *Nama* | $R_i$ | $V_i$ | $R_i/V_i$ | *Rating* | *Value* | *Density* | *Optimal* |
| 1 | Lloris | 7.04 | 49.5 | 0.142 | 0 | 1 | 1 | 0 |
| 2 | Neuer | 6.89 | 67 | 0.102 | 0 | 1 | 0 | 0 |
| 3 | Courtois | 7.07 | 55 | 0.129 | 1 | 0 | 1 | 1 |
| 4 | De Gea | 7.25 | 58 | 0.125 | 1 | 0 | 0 | 1 |
| Rata-rata *rating* | | | | | 7.16 | 6.97 | 7.06 | 7.16 |
| Total biaya | | | | | 113 | 116.5 | 104.5 | 113 |

*Tabel 4.1 Tabel perbandingan pendekatan algoritma Greedy*

Tabel diatas menunjukkan bahwa *greedy by density* tidaklah menghasilkan penjaga gawang dengan *rating* tertinggi, seperti kebanyakan persoalan Knapsack melainkan *greedy by rating* menghasilkan solusi yang sesuai dengan solusi optimal. Analisis untuk masing-masing pendekatan *greedy* akan diuraikan di paragraph-paragraf selanjutnya.

*Greedy by Rating* menghasilkan solusi terbaik, bahkan solusi yang dihasilkan sama dengan solusi optimal sehingga metode ini sangat dianjurkan untuk para pemain FM yang lebih mementingkan kualitas dalam pembelian pemain.

*Greedy by Value* menghasilkan solusi terburuk dibandingkan kedua pendekatan lain. Selain menghasilkan penjaga gawang dengan rata-rata *rating* terendah, pendekatan ini juga menggunakan total biaya tertinggi sehingga membuat pendekatan ini sangatlah tidak dianjurkan untuk para pemain FM.

*Greedy by Density* tidak menghasilkan solusi yang terbaik. Rata-rata *rating* yang dihasilkan juga berjarak cukup jauh dari rata-rata *rating* yang dihasilkan pendekatan *greedy by rating*. Akan tetapi, total biaya yang digunakan oleh pendekatan ini merupakan yang terkecil. Total biaya yang dihasilkan *greedy by density* jauh lebih kecil dibandingkan total biaya terkecil kedua dengan selisih 8.5. Selisih biaya tersebut dapat digunakan di FM untuk menaikkan *payroll budget* ataupun keperluan-keperluan klub lainnya. Oleh karena itu, pengguaan *greedy by density* dianjurkan bagi para pemain FM yang ketat dalam hal finansial klub tanpa menghilangkan kualitas pembelian pemain.

## V. KESIMPULAN

Algoritma *greedy* mampu digunakan di berbagai hal, termasuk pembelian pemain di permainan Football Manager. Penggunaan *greedy by rating* dianjurkan bagi pemain yang mementingkan kualitas pemain. Penggunaan *greedy by density* dianjurkan bagi pemain yang ketat dalam finansial klub sedangkan *greedy by value* tidak dianjurkan bagi pemain.

## REFERENSI


[1] Munir, Rinaldi. *Diktat Kuliah IF2211 Strategi Algoritma.*, Bandung: Departemen Teknik Informatika, 2004.

[2] http://www.footballmanager.com/, diakses 10 Mei 2018.


## 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, 14 Mei 2018

Shandy
13516097