Implementasi Algoritma Greedy ‘Dua Sisi’ pada Permainan 2048

Ramos Janoah (13514089)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13514089@std.stei.itb.ac.id

***Abstrak*— Makalah ini akan mejelaskan tentang penggunaan algoritma *greedy* pada permainan 2048**

***Kata kunci*— Pohon Keputusan, Pohon, Unit Kegiatan Mahasiswa, Unit**

I. PENDAHULUAN

Algoritma *greedy* adalah salah satu algoritma yang masih sering dipakai sebagai persoalan optimasi. Meskipun sudah lama ditemukan, tetapi algoritma *greedy* masih efektif untuk digunakan, setidaknya untuk aprokasimasi.

Permainan 2048 adalah permainan yang diciptakan pada tahun 2014 oleh seseorang bernama Gabrielle Cirulli asal Italia pada saat ia berumur 19 tahun. Permainan ini sangatlah sederhana. Permainan terdiri dari bidak-bidak dan sebuah papan berukuran 4x4. Pada bidak tersebut, akan terisi angka 2^n (2, 4, 8, 16, dst..). Bila sebuah angka 2^n bertemu, maka bidak tersebut akan bersatu dan akan menghasilkan angka 2^{n+1}, dan tujuan permainan ini adalah untuk menghasilkan ubin dengan angka 2048 (2^{11}). Cara mempertemukan angka tersebut adalah dengan cara menggeser semua bidak ke satu sisi (rata kiri, rata kanan, rata atas atau rata bawah).

Gambar 1. Contoh Kondisi Papan Permainan 2048

Pada makalah ini akan dibahas mengenai penerapan sebuah algoritma *greedy* untuk menjadi prinsip dalam permainan ini. Yang diharapkan dari makalah ini adalah proses pemecahan masalah yang akan semakin dimengerti setelah membaca makalah ini, baik dalam permainan ataupun kehidupan sehari-hari.

II. DASAR TEORI

A. *Algoritma Greedy*

Algoritma *greedy* adalah algoritma yang terkenal dalam masalah optimasi. Dalam bahasa inggris, *greedy* memiliki arti ‘rakus’ atau ‘tamak’. Prinsip yang digunakan sama dengan artinya, yaitu prinsip ‘ambil yang bisa diambil lebih dahulu’. Pada setiap selesai melangkah, akan dievaluasi setiap langkah yang akan diambil, dan akan diambil langkah yang terbaik dari pilihan langkah tersebut. Langkah terbaik dari setiap pilihan tersebut dinamakan langkah ‘optimum lokal’. Dan langkah-langkah tersebut dijadikan satu kesatuan solusi yang dijadikan ‘solusi global’. Jadi, ada dua hal yang harus dilakukan pada algoritma *greedy*:

1. Mengambil pilihan yang terbaik yang bisa didapat pada saat itu, tanpa memperhatikan konsekuensi kedepan (optimum lokal).

2. Terus lakukan langkah pertama sehingga solusi ditemukan, dengan berharap solusi global tersebut adalah solusi optimum.

Secara Umum, algoritma *greedy* memiliki elemen-elemen sebagai berikut:

1. **Himpunan Kandidat**.
Himpunan kandidat adalah pilihan-pilihan dari setiap langkah yang bisa diambil pada saat tersebut.

2. **Himpunan Solusi**.
Himpunan solusi adalah himpunan yang berisi pilihan-pilihan yang sudah diambil untuk mencapai solusi.

3. **Fungsi Seleksi**.
Fungsi seleksi adalah fungsi untuk memilih pilihan yang paling optimal pada saat langkah akan diambil (fungsi untuk memilih optimum lokal)

4. **Fungsi kelayakan.**
Fungsi kelayakan adalah fungsi yang memeriksa apakah dari perjalanan mencari solusi tersebut, pilihan-pilihan tersebut masih memenuhi aturan yang berlaku atau tidak.

5. **Fungsi Obyektif**.
Fungsi obyektif adalah fungsi yang digunakan untuk mengetahui apakah himpunan solusi yang dihasilkan sudah memenuhi solusi yang paling optimal atau belum.

Langkah-langkah algoritma *Greedy* memiliki banyak contoh dan variasi. Misalnya, pada kasus mencari jumlah pecahan uang dengan jumalh unit uang yang minimum, seseorang diminta mencari pecahan uang dari uang Rp.78.000,-, dengan uang pecahan yang dimiliki adalah Rp.100.000,-, Rp.50.000,-, Rp.20.000,-, Rp.10.000,-, Rp.5.000,-, Rp.2.000,-, Rp.1.000,-. Maka pemecahan dengan algoritma *greedy* adalah sebagai berikut:

1. Buatlah himpunan kosong sebagai himpunan calon solusi, misalnya S = {}
2. Lakukan *sorting* terhadap pilihan-pilihan yang bisa diambil. Dalam hal ini, urutan dari terbesar hingga terkecil adalah: 100.000, 50.000, 20.000, 10.000, 5.000, 2.000, 1.000.
3. Lakukan pemilihan dari pilihan-pilihan pecahan yang bisa diambil dan masukan ke dalam himpunan solusi. Langkah ini diulang hingga jumlah dari himpunan solusi sudah sama dengan uang yang diinginkan. Berikut adalah perinciannya.
    - Pecahan uang terbesar yang diambil adalah 50.000. Sisa uang 28.000, S = {50.000}
    - Berikutnya, pecahan uang terbesar yang dapat diambil adalah 20.000. Sisa uang 8.000 S = {50.000, 20.000}
    - Berikutnya, pecahan uang terbesar yang dapat diambil adalah 5.000. Sisa uang 3.000 S = {50.000, 20.000, 5.000}
    - Berikutnya, pecahan uang terbesar yang dapat diambil adalah 2.000. Sisa uang 1.000 S = {50.000, 20.000, 5.000, 2.000}
    - Berikutnya, pecahan uang terbesar yang dapat diambil adalah 1.000. Tidak ada uang yang bersisa. S = {50.000, 20.000, 5.000, 2.000, 1.000}
    - Jumlah himpunan S sudah sama dengan 78.000. Himpunan S sudah menjadi himpunan Solusi.

Selain masalah mencari jumlah pecahan uang dengan jumalh unit uang yang minimum, algoritma *greedy* juga sering dipakai dalam masalah-masalah lain, seperti:

1. Memiminimalisasi waktu sistem
2. Integer *knapsack*
3. Penyelesaian *Egyptian Fraction*
4. Mencari pohon merentang minimum (algoritma Prim dan algoritma Kruskal).
5. Mencari lintasan terpendek
6. Masalah-masalah seleksi lain yang berkaitan dengan optimasi.

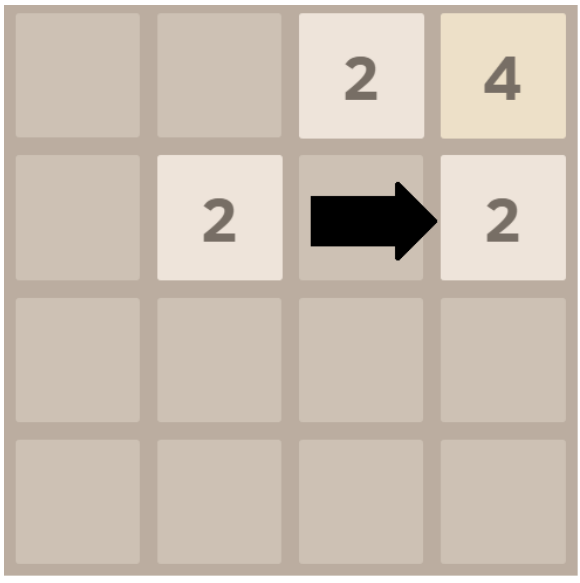
Algoritma *Greedy* tidak selalu menghasilkan nilai yang optimal, tapi cukup untuk mencari hampiran dari solusi optimal, dengan kompleksitasnya yang cukup singkat dibandingkan dengan *'exhaustive algorithm'*

B. Peraturan dan Cara Bermain 2048

Permainan 2048 biasa dimainkan di platform *mobile* dengan sistem operasi *Android* atau *iOS*. Berikut adalah peraturan-peraturan pada permainan 2048.

1. Pemain bisa melakukan Sapu kanan, Sapu kiri, Sapu atas atau Sapu bawah bila langkah tersebut mungkin. Sebuah langkah akan disebut sebagai langkah yang mungkin jika:

a. Pada saat melakukan sapu ke arah tertentu, ada ubin yang bernomor sama. Contohnya adalah sebagai berikut :



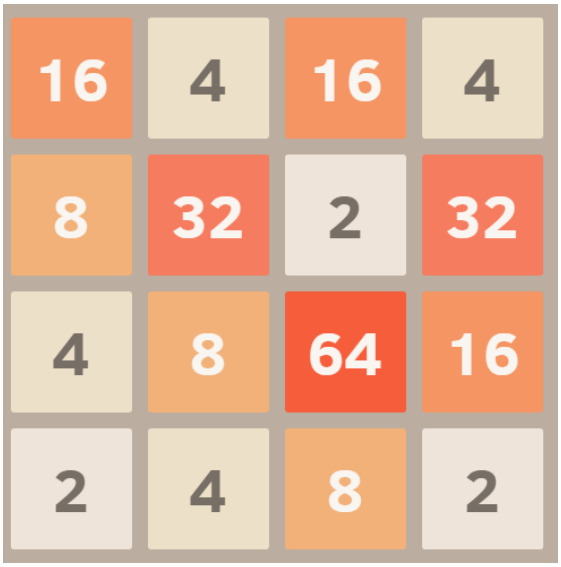
Gambar 2. Langkah yang Mungkin Dilakukan I. Pada kondisi ini, sapu kanan mungkin dilakukan.

b. Pada saatu melakukan sapu ke arah tertentu, ada ubin yang memiliki kolom kosong di arah

tersebut. Misalkan, pada saat melakukan sapu ke arah kanan, ada ubin yang memiliki celah kosong di sebelah kanan.

Gambar 3. Langkah yang Mungkin Dilakukan II. Pada kondisi ini, sapu kiri mungkin dilakukan.

2. Bila pemain sudah tidak mampu melakukan langkah apapun, maka permainan selesai.



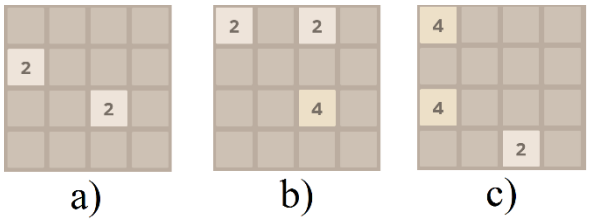
Gambar 4. Kondisi Permainan Selesai.

Alur permainannya adalah sebagai berikut:

1. Pada saat awal permainan, aka nada dua ubin bernomor ‘2’, yang akan muncul secara acak.

2. Pemain akan melakukan langkah, dan setelah pemain melakukan langkah, akan keluar sebuah ubin pada papan secara acak dengan nomor rendah. Nomor rendah yang dimaksud pada permainan adalah tergantung versi 2048 yang dimainkan. Pada kali ini, nomor yang mungkin keluar hanyalah ‘2’ dan ‘4’.

3. Permainan akan terus berlanjut sampai game over. Sebenarnya, tujuan permainan ini adalah bermain sampai menghasilkan ubin yang memiliki nomor 2048. Namun, pada kenyataannya, permainan tidak berakhir pada saat ubin tersebut muncul, dan sebenarnya permainan masih bisa dilanjutkan sampai berakhir. Oleh karena itu, pada makalah ini, permainan disebut berakhir jika pemain sudah tidak dapat melakukan langkah.



Gambar 5. a) Kondisi awal permainan, b) Pemain melakukan sapu atas. Semua ubin tergeser ke atas, kemudian muncul ubin 4 secara acak (baris 3 kolom 3). c) Pemain melakukan sapu kiri. Semua ubin tergeser ke kiri, muncul ubin 2 secara acak (baris 4 kolom 3)

III. PEMBAHASAN

A. Penerapan Prinsip Greedy Pada Permainan 2048

Prinsip greedy bisa diterapkan pada permainan 2048. Prinsip ‘take what you can get now’ tersebut diterapkan pada keputusan pengambilan langkah yang diambil. Nilai langkah yang diambil dapat dinilai dari nomor terbesar dari bidak yang dapat disatukan. Gambar berikut adalah ilustrasinya.

Gambar 6. Nilai untuk Sapu Atas adalah 8, Nilai untuk sapu kiri adalah 4.

Dapat dilihat bahwa nilai sapu atas akan selalu sama dengan sapu bawah, dan nilai sapu kiri akan selalu sama dengan nilai sapu kanan. Dengan begitu, prinsip *greedy* bisa diterapkan pada permainan 2048.

B. Algoritma Greedy Dua Sisi Pada Permainan 2048

Telah diketahui sebelumnya bahwa setiap langkah bisa memiliki nilai, prinsip *greedy* terbukti bisa diterapkan pada permainan 2048. Selain itu, untuk langkah yang mungkin versi I (lih. Gambar 2, tidak berlaku untuk versi II), dapat diketahui juga bahwa nilai sapu atas akan sama dengan sapu bawah dan nilai sapu bawah selalu sama dengan sapu atas, dan selalu ada pilihan untuk mengambil langkah sapu ke arah tertentu atau kearah sebaliknya. Namun, permainan serasa tidak wajar jika mengambil langkah dengan nilai yang lebih tinggi secara acak. Misalnya, suatu saat langkah terbesar adalah ke atas atau ke bawah, dan langkah yang diambil adalah langkah ke atas. Lalu, beberapa saat kemudian, langkah yang diambil adalah langkah ke bawah. Hal tersebut membuat ketidak-efektifan dan membuat permainan akan menjadi lebih cepat berakhir. Oleh karena itu, harus dipilih dua sisi yang akan dijadikan patokan bila terjadi pilihan langkah (oleh karena itu dinamakan dua sisi), dan kedua sisi tersebut tidak boleh bersebrangan. Misal, yang dipilih adalah sisi kanan dan atas.

Untuk kemudahan penyebutan, maka dua sisi yang akan dipilih akan disebut sebagai sisi 1 dan sisi 2. Sementara, sisi 3 dan sisi 4 adalah sisi yang pasif atau sebagai sisi yang akan dipilih jika sisi 1 dan sisi 2 tidak mungkin dijadikan arah sapuan.

Dengan memegang prinsip *greedy* dan prinsip dua sisi, kita memiliki elemen-elemen untuk algoritma *greedy* sebagai berikut :

1. **Himpunan Kandidat**.
Himpunan kandidat untuk permainan ini adalah langkah-langkah yang mungkin dilakukan untuk setiap langkah. Himpunan kandidat yang paling memiliki banyak anggota adalah C = {Sapu atas, sapu bawah, sapu kiri, sapu kanan} (Semua langkah mungkin). Kenyataannya, himpunan kandidat tersebut hanya mungkin didapat pada langkah pertama.

2. **Himpunan Solusi**.
Himpunan solusi untuk permainan ini adalah langkah-langkah yang dimainkan pada saat permainan. Namun, Himpunan solusi untuk permainan ini sedikit tidak relevan untuk dijadikan patokan, karena ubin yang keluar pada saat selesai mengambil langkah adalah acak. Pada kali ini, himpunan solusi tidak akan terlalu dimuat. Yang akan dimuat hanyalah kondisi akhir dari permainan (kondisi papan dan letak ubin)

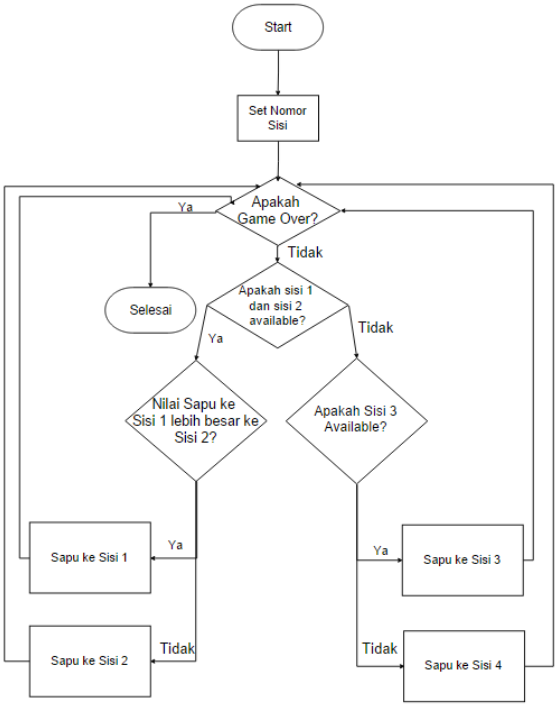
3. **Fungsi Seleksi**.
Fungsi seleksi pada permainan ini adalah sebagai berikut:

a. Menentukan apakah diantara sisi 1 dan sisi 2 ada yang *available* atau tidak. Jika salah satu atau keduanya *available*, maka pilihlah sisi yang memiliki nilai lebih besar. Bila keduanya memiliki nilai yang lebih besar, maka
b. Jika diantara sisi 1 dan sisi 2 tidak ada yang available, maka dipilihlah sisi berikutnya yang menjadi opsi bila sisi 1 dan sisi 2 tidak *available*.
c. Jika sisi tersebut juga belum dapat memenuhi, maka sapulah ke sisi terakhir.

4. **Fungsi kelayakan.**
Fungsi kelayakan adalah fungsi untuk mengecek apakah sebuah langkah masih dapat diambil atau tidak.

5. **Fungsi Obyektif**.
Fungsi obyektif tidak terlalu dipakai dalam permainan ini, karena tidak ada batasan sebuah solusi disebut optimal atau tidak.

Dengan penjelasan di atas, algoritma dapat dibuat dengan flow diagram sebagai berikut:



Gambar 7. *Flowchart* Algoritma *Greedy* '2 sisi'

Implementasi dalam *pseudo-code* adalah sebagai berikut.

```
Procedure Solve2048(input/output : PapanPermainan P)
{- I.S : P berisikan sebuah kondisi start, dimana
         P masih berisi sebuah kondisi awal.
 - F.S : P sudah berisi kondisi akhir dengan
         skor terbesar yang dihasilkan dengan
         algoritma greedy }

KAMUS LOKAL

ALGORITMA
    setSisi(P, 1, Kanan)
    setSisi(P, 2, Atas)
    setSisi(P, 3, Kiri)
    setSisi(P, 4, Bawah)
    while (not(GameOver(P))) do
        if (IsAvailable(P, 1) or IsAvailable(P, 2)
        { Prinsip Greedy Dipakai di 'if' ini }
            if ((NilaiSwipe(P, 1) => (NilaiSwipe(P, 2))
                SwipeBoard(P, 1)
            else if
                SwipeBoard(P, 2)
            endif
        else if (IsAvailable(P, 3))
            SwipeBoard(P, 3)
        else
            SwipeBoard(P, 4)
        endif
        AddNewTilesRandomly(P)
    endwhile

    -> P
```

Dan berikut adalah penjelasan-penjelasan fungsi yang dipakai pada algoritma tersebut. Fungsi-fungsi tersebut tidak diimplementasikan karena hal implementasi fungsi-fungsi tersebut adalah hal yang berbeda dengan algoritma *greedy*.

1. SetSisi

```
Procedure SetSisi(input/output : PapanPermainan
        input : integer N,
        input : Sisi S)
{ I.S : Sembarang
F.S : Sisi S di-set pada sisi no N pada papan P
```

2. GetSisi

```
Function GetSisi(input/output : PapanPermainan
        input : integer N
{ Mengembalikan sisi nomor N pada papan P }
```

3. IsAvailable

```
Function IsAvailable(input : PapanPermainan P,
                     input : Sisi S) -> boolean
{ Mengembalikan nilai true jika pemain bisa melakukan sl
    ke arah sisi S }
```

4. SwipeBoard

```
Procedure SwipeBoard(input/output : PapanPermainan
                     input : int N)
{ I.S : Sembarang
F.S : Melakukan sapuan ke arah nomor sisi N }
```

5. NilaiSwipe

```
Function NilaiSwipe(input/output : PapanPermainan P,
                    input : int N)
{ Mengembalikan nilai dari sapuan ke arah sisi bernomor
```

6. AddNewTilesRandomly

```
AddNewTilesRandomly(input/output : PapanPermainan P)
{ I.S : Sembarang
F.S : Mengembalikan Papan P dengan sebuah ubin acak ditaruh
        pada P tanpa menimpa ubin yang lain }
```

IV. HASIL PENGUJIAN DAN ANALISIS

A. Hasil Pengujian

Algoritma tersebut diaplikasikan secara manual untuk mencoba keefektifan algoritma tersebut. Eksperimen dilakukan sebanyak 10 kali.

Nomor Percobaan	Banyak langkah yang bisa dilakukan	Nomor ubin terbesar pada saat akhir permainan	Skor (Skor adalah total nomor ubin yang sudah dijumlahkan)
1	235	256	4360
2	214	128	3520
3	291	512	5112
4	272	256	4778
5	255	256	4520
6	205	128	4120
7	321	512	5224
8	**340**	**512**	**5384**
9	265	256	4700
10	242	256	4446

Dapat dilihat bahwa dari sepuluh percobaan, nomor ubin tertinggi yang dapat diraih adalah 512 (2^9), dengan skor 5384 melalui 340 langkah.

B. Analisis

Setiap percobaan memiliki hasil yang cukup berbeda diantara hasil terburuk dengan hasil terbaik. Hasil terburuk adalah : 205 langkah, nomor maksimal ubin 128 dan skor 4120. Sedangkan hasil terbaik adalah 340 langkah, nomor maksimal ubin adalah 512, dan skor 5384. Hal ini disebabkan karena sistem penaruh ubin secara acak yang membuat setiap permainan dapat berbeda cukup jauh. Hal ini membuktikan bahwa Algoritma *greedy* bisa dipakai dalam permainan ini, namun tidak semurni itu langsung diimplementasikan permainan.

V. KESIMPULAN

Algoritma *greedy* sangat dapat dipakai sebagai prinsip untuk memainkan permainan 2048. Hanya saja, untuk pengembangan, penggunaannya tidak bisa semurni pada kasus-kasus lain. Harus ada pengembangan-pengembangan yang dilakukan bila ingin membuat suatu algoritma untuk memainkan permainan ini lebih efektif. Dengan pembuatan algoritma untuk permainan ini, diharapkan algoritma *greedy* dapat digunakan untuk berbagai macam hal lain baik dalam permainan ataupun kehidupan sehari-hari.

VI. UCAPAN TERIMAKASIH

Penulis ingin mengungkapkan rasa terimakasih kepada Tuhan Yang Maha Esa, atas kasih dan penyertaan-Nya, makalah ini bisa terselesaikan. Mungkin makalah ini jauh dari sempurna, tapi tanpa penyertaan-Nya, makalah ini tidak dapat terselesaikan. Penulis juga ingin memberikan terimakasih kepada yang terhormat Bapak Dr. Ir. Rinaldi Munir, M.T. dan Ibu Dr.Nur Ulfa Maulidevi, ST, M.T. atas pendidikan dan pengajaran yang diberikan, terutama pada mata kuliah strategi algoritma, sehingga makalah ini bisa terselesaikan dengan baik.

DAFTAR PUSTAKA

[1] Munir, Rinaldi. 2009. Strategi Algoritma Bandung: Penerbit Informatika, Palasari.

[2] http://games.tooliphone.net/id/game/2048, diakses pada tanggal 4 Mei 2016, pukul 12.00

[3] http://www.it-jurnal.com/2015/09/pengertian-algoritma-greedy.html, diakses pada tanggal 4 Mei 2016, pukul 11.00

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, 9 Desember 2015

Ramos Janoah Hasudungan
13514089