Penerapan Algoritma *Backtracking* untuk Penempatan Furnitur dalam Serenitea Pot

Rhapsodya Piedro Asmorobangun - 13519084[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*piedro@itb.ac.id*

***Abstrak*—Genshin Impact adalah sebuah permainan dengan sistem dunia terbuka. Salah satu aspek dari permainan ini adalah Serenitea Pot, yaitu membangun rumah berisi berbagai furnitur. Agar pemain bisa mendapatkan keuntungan paling banyak dari fitur ini, perlu ditempatkan furnitur sebanyak mungkin dengan ruang yang terbatas dan keuntungan yang berbeda antar furnitur. Kombinasi furniture yang memberikan keuntungan paling banyak dapat dicari dengan menggunakan algoritma greedy.**

***Keywords*—Serenitea Pot, Algoritma *Backtracking*, Persoalan Knapsack**

I. Pendahuluan

Genshin Impact adalah sebuah permainan aksi bermain peran (*action role-playing game*) yang berasal dari Cina. Permainan ini memiliki sistem dunia terbuka, sehingga pemain dapat menjelajahi dan menemukan berbagai lokasi dan barang di dunia permainan secara bebas. Dengan sistem ini, interaksi permainan tidak linear maupun terstruktur.

Salah satu fitur yang baru ditambahkan dalam permainan ini adalah fitur *housing* bernama Serenitea Pot, yaitu fitur membangun rumah. Dalam Serenitea Pot, pemain diminta untuk membangun dan menempatkan furnitur sesuai dengan kreativitas masing-masing. Terdapat banyak pilihan furnitur yang tersedia, tetapi terdapat perbedaan besar antar furnitur. Semakin besar sebuah furnitur, semakin banyak ruang yang dimakan oleh furnitur tersebut. Ruang yang tersedia juga terbatas, maka pemain harus hati-hati agar tidak menempatkan terlalu banyak furnitur.

Tiap furnitur juga dapat memberikan energi. Energi yang terakumulasi dapat ditukar dengan hadiah-hadiah tertentu. Tiap furnitur memberikan jumlah energi yang berbeda, yang tidak berkaitan dengan besar dari sebuah furnitur. Agar dapat mendapatkan keuntungan sebanyak mungkin, pemain perlu menaruh sebanyak mungkin furnitur yang memberi banyak energi, namun masih mematuhi batas ruang yang tersedia.

II. Dasar Teori

A. Persoalan Knapsack 1/0

Persoalan Knapsack adalah sebuah persoalan dalam optimisasi kombinatorial. Jika diberikan sejumlah objek dan setiap objek memiliki properti bobot dan keuntungan, tentukan jumlah masing-masing objek yang dapat dimasukkan ke dalam koleksi sehingga sehingga diperoleh total keuntungan yang maksimal dan bobot total lebih kecil dari atau sama dengan sebuah batas yang ditentukan. Salah satu variasi persoalannya adalah persoalan Knapsack 1/0, dengan tiap objek dapat dimasukkan ke dalam knapsack (1) atau tidak dimasukkan sama sekali (0).

B. Algoritma *Backtracking*

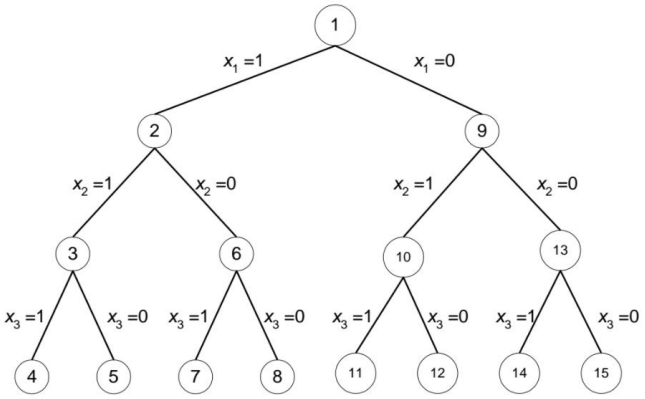
Algoritma *backtracking*, atau algoritma runut-balik, adalah algoritma yang digunakan untuk mencari semua solusi dalam sebuah masalah komputasional. Algoritma backtracking membuang kandidat solusi jika telah ditentukan bahwa kandidat solusi tersebut tidak mungkin bisa menjadi sebuah solusi yang lengkap.

Terdapat beberapa properti umum dari algoritma *backtracking*. Solusi persoalan biasanya dinyatakan sebagai vektor dengan *n-tuple*: $X = (x_1, x_2, \ldots, x_n)$, $x_i \in S_i$. Umumnya $S_1 = S_2 = ... = S_n$. Fungsi pembangkit dinyatakan sebagai predikat T(). T(x[1], x[2], …, x[k-1]) membangkitkan nilai untuk x_k, yang merupakan komponen vektor solusi. Fungsi pembatas dinyatakan sebagai predikat $B(x_1, x_2, \ldots, x_k)$. B bernilai true jika $(x_1, x_2, \ldots, x_k)$ mengarah ke solusi atau tidak melanggar kendala. Jika *true*, maka pembangkitan nilai untuk x_{k+1} dilanjutkan, tetapi jika false, maka $(x_1, x_2, \ldots, x_k)$ dibuang.

Semua kemungkinan solusi dari persoalan disebut sebagai ruang solusi. Ruang solusi diorganisasikan ke dalam struktur pohon berakar. Tiap simpul pohon menyatakan status persoalan, sedangkan sisi dilabeli dengan nilai -nilai x_i. Lintasan dari akar ke daun menyatakan solusi yang mungkin. Seluruh lintasan dari akar ke daun membentuk ruang solusi. Pengorganisasian pohon ruang solusi diacu sebagai pohon ruang status.

Misalnya untuk persoalan Knapsack 1/0 dengan n = 3, solusi persoalan dinyatakan sebagai $X = (x_1, x_2, x_3)$, dengan $x_i \in \{0, 1\}$. Ruang solusinya adalah { (0, 0, 0), (0, 1, 0), (0, 0, 1), (1, 0, 0),

(1, 1, 1), (1, 1, 0), (1, 0, 1), (0, 1, 1), (1, 1, 1)}. Ruang solusinya adalah sebagai berikut:



Gambar 1. Pohon ruang solusi dari persoalan Knapsack 1/0 untuk n = 3
Sumber: Bahan Kuliah IF2211 Strategi Algoritma

Solusi dapat dicari dengan membangkitkan simpul-simpul status sehingga menghasilkan lintasan dari akar ke daun. Aturan pembangkitan simpul yang dipakai mengikuti aturan algoritma DFS. Simpul-simpul yang sudah dibangkitkan dinamakan simpul hidup, sementara simpul hidup yang sedang diperluas dinamakan simpul-E.

Tiap kali simpul-E diperluas, lintasan yang dibangun olehnya bertambah panjang. Jika lintasan yang sedang dibentuk tidak mengarah ke solusi, maka simpul-E tersebut tidak akan diperluas lagi, sehingga menjadi simpul mati. Fungsi yang digunakan untuk mematikan simpul-E adalah fungsi pembatas.

Jika pembentukan lintasan berakhir dengan simpul mati, maka proses pencarian akan *backtrack* ke simpul pada aras di atasnya. Langkah tersebut diulang dengan membangkitkan simpul anak yang lainnya, yang menjadi simpul-E yang baru. Pencarian dihentikan bila telah sampai pada *goal node*.

III. METODE PENYELESAIAN MASALAH

Gambar 2 menunjukkan tampilan *Serenitea Pot* dalam Genshin Impact. Di bagian bawah terdapat pilihan-pilihan furnitur yang bisa ditempatkan. Di bagian kiri bawah tiap gambar furnitur, terdapat angka yang menunjukan berapa energi yang bisa didapatkan dengan menempatkan furnitur tersebut. Namun, bobot dari tiap furnitur tidak dapat dilihat langsung dalam *Serenitea Pot*. Pengguna Reddit dengan nama akun u/ruzuki telah mengkompilasi energi (AE) dan bobot (*Load*) yang dimiliki oleh tiap furnitur dalam sebuah *spreadsheet*, seperti yang dapat dilihat di gambar 3.

Gambar 2. Tampilan *Serenitea Pot*

Adeptal Energy vs Load Analysis

Indoors | Outdoors | Master List | Change History

Category	Subcategory	Name	Icon	AE	Load	AE/
Courtyard	Courtyard Wall	Carved Courtyard Fence Ending		90	35	
Courtyard	Courtyard Wall	Carved Courtyard Fence		90	35	
Landscape	Item	Feather-Light Praise		90	35	
Landscape	Table	Alchemist's Crafting Bench		90	35	
Landscape	Cabinet	The Blue Ocean's Treasure		90	35	
Courtyard	Courtyard Wall	Birch Main Courtyard Gate		90	45	
Landscape	Ornament	Stone Lion Statue: The Warding		90	50	
Landscape	Ornament	Stone Lion Statue: The Knowing		90	50	
Landscape	Item	Fruit Seller's Caution		60	35	
Landscape	Item	Fruit Seller's Toil		60	35	

Published by Google Sheets – Report Abuse – Updated automatically every 5 minutes

Gambar 3. Spreadsheet energi dan bobot dari tiap furnitur dalam *Serenitea Pot*
Sumber: https://docs.google.com/spreadsheets/d/e/2PACX-1vQd68uxHchlvj5_0uQbEldHpCr0faYSD8nndKmk6Ol7niszg2vM1t4R86y_2e5ym03TmVH8C9jw9YIt/pubhtml#

Furnitur-furnitur tersebut dapat ditempatkan di tempat-tempat yang berbeda. *Serenitea Pot* terbagi menjadi dua bagian utama, yaitu *indoor* dan *outdoor*. Kedua bagian tersebut dibagi lagi menjadi beberapa area yang berbeda. Tiap area juga memiliki batas bobot furnitur yang berbeda. Misalnya, untuk Area 1 di bagian *outdoor* seperti yang dapat dilihat di gambar 2 memiliki batas bobot sekitar 8800.

Persoalan ini sama dengan persoalan Knapsack, sehingga solusinya dapat dicari dengan algoritma *backtracking*. Solusi persoalan dinyatakan sebagai vektor X = (x_1, x_2, …, xn), dengan $x_i \in \{0, 1\}$. x adalah furnitur-furnitur yang dimiliki oleh pemain, sementara n adalah jumlah furnitur yang dimiliki oleh pemain. 1 berarti furnitur ditempatkan di area, sementara 0 sebaliknya.

Fungsi pembatas di persoalan ini adalah total bobot dari semua furnitur yang akan ditempatkan tidak melebihi batas bobot dari area yang akan ditempatkan furnitur. Jika beban sudah melebihi, maka simpul tidak akan diperluas lagi dan akan menjadi simpul mati.

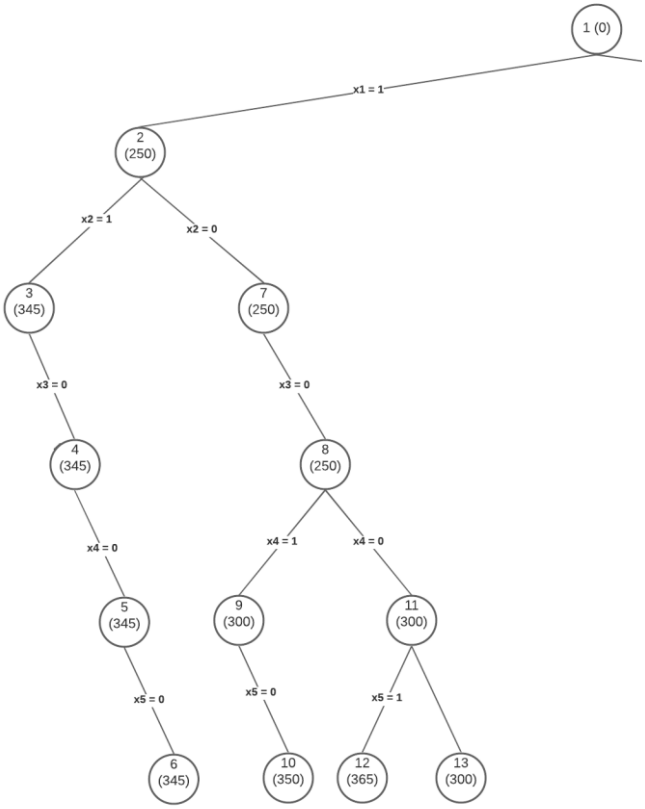
IV. STUDI KASUS

Batas bobot yang akan digunakan adalah 350. Berikut adalah daftar furnitur yang dimiliki, beserta dengan energi dan bobot dari tiap furnitur.

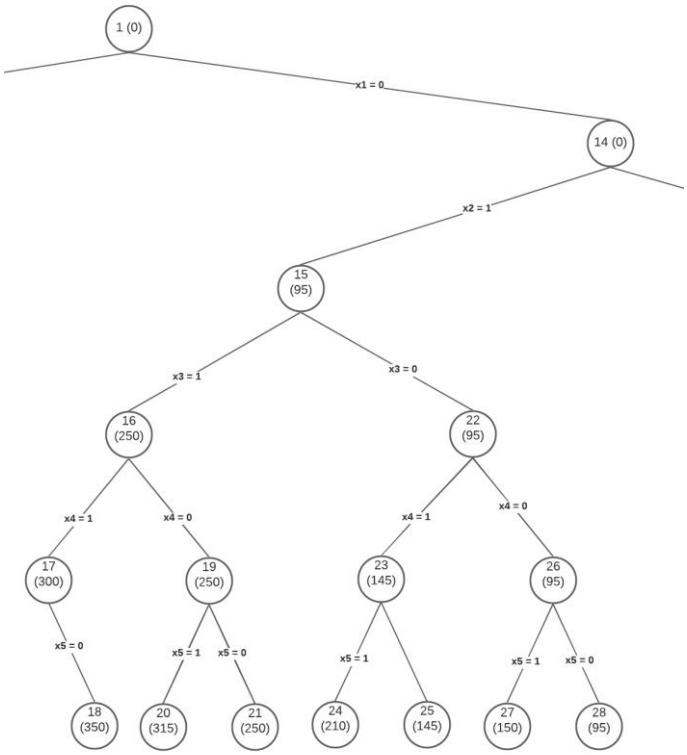
Nama Furnitur	Energi	Bobot
Tall Birch (x1)	20	250
Old Well (x2)	30	95
Hilichurl Outpost Hut (x3)	60	155
Kindletree (x4)	20	50
Fir Shelves (x5)	30	65

Tabel I. Data furnitur yang dimiliki

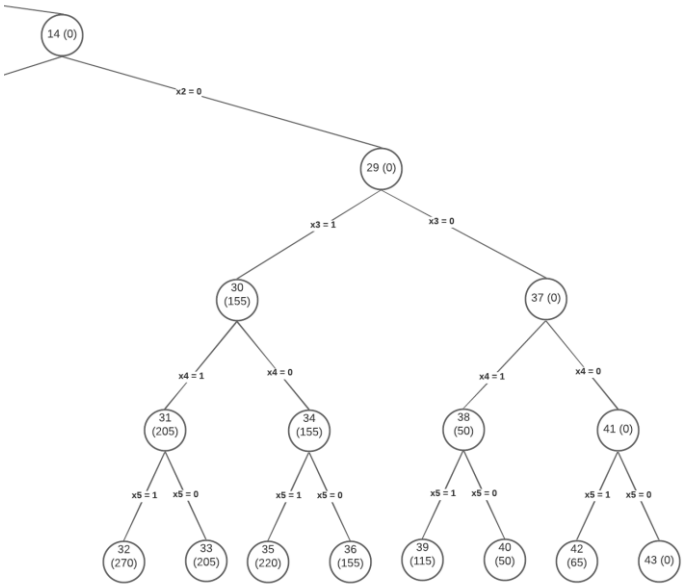
Dan berikut adalah pohon dinamis yang dibentuk selama pencarian. Bobot ditandai dengan angka yang terdapat di dalam kurung sebuah simpul.



Gambar 4. Pohon dinamis untuk x1 = 1



Gambar 5. Pohon dinamis untuk x1 = 0 dan x2 = 1



Gambar 6. Pohon dinamis untuk x1 = 0 dan x2 = 1

Jalur dengan energi terbesar adalah jalur x2 = 1, x3 = 1, dan x5 = 1, dengan total energi yang didapat sebesar 120 dan load sebesar 315.

V. ANALISA

Penempatan furnitur sehingga mendapatkan energi terbanyak terbukti dapat dicari dengan mencari solusi dari persoalan Knapsack. Walaupun memang solusi yang didapat adalah solusi yang paling optimal, terdapat beberapa kendala dalam

pengaplikasian persoalan ini ke dalam *Serenitea Pot*. Pertama, studi kasus yang dilakukan hanya mengambil sebagian kecil dari furnitur yang bisa dimiliki oleh pemain, dan batas bobot furnitur yang diujikan juga jauh lebih kecil dibandingkan batas bobot pada sebuah area di *Serenitea Pot*. Untuk mencoba mencari solusi yang optimal untuk kondisi nyata dalam *Serenitea Pot*, pemain perlu mencatat setiap furnitur yang ia miliki.

Kedua, batas bobot di dalam *Serenitea Pot* tidak menjadi batasan yang sangat ketat dan tidak bisa dihindari. Seiring berjalannya waktu, pemain dapat mengakses area-area baru. Sehingga, furnitur-furnitur yang terdapat di area lama dapat dipindahkan ke area baru, sehingga meringankan bobot yang ada di area lama.

REFERENSI

[1] R. Munir, *Diktat Kuliah IF2251 Strategi Algoritmik.* Bandung: Informatika Bandung, 2007.

[2] E. Gurari, "DATA STRUCTURES: Chapter 19: Backtracking Algorithms", CIS 680, 1999.

[3] https://docs.google.com/spreadsheets/d/e/2PACX-1vQd68uxHchlvj5_0uQbEldHpCr0faYSD8nndKmk6Ol7niszg2vM1t4R86y_2e5ym03TmVH8C9jw9YIt/pubhtml#. Diakses pada 11 Mei 2021.

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, 11 Mei 2021

Rhapsodya Piedro Asmorobangun / 13519084