

Tugas Besar 2 IF2211 Strategi Algoritma
Semester II tahun 2024/2025

Pemanfaatan Algoritma *BFS* dan *DFS* dalam Pencarian Recipe pada Permainan
Little Alchemy 2

Revisi 1: Jumat, 9 Mei 2025, pukul 14:35 WIB (Perubahan solusi terpendek yang perlu dicari dan sistem tier recipe)

Batas pengumpulan : Hari Selasa, 13 Mei 2025 pukul 22.11 WIB

Arsip pengumpulan :

- Source program yang dapat dijalankan disertai README
- Laporan (*soft copy*)

Deskripsi Tugas:



Gambar 1. Little Alchemy 2
(sumber: <https://www.thegamer.com>)

Little Alchemy 2 merupakan permainan berbasis web / aplikasi yang dikembangkan oleh Recloak yang dirilis pada tahun 2017, permainan ini bertujuan untuk membuat 720 elemen dari

4 elemen dasar yang tersedia yaitu *air*, *earth*, *fire*, dan *water*. Permainan ini merupakan sekuel dari permainan sebelumnya yakni Little Alchemy 1 yang dirilis tahun 2010.

Mekanisme dari permainan ini adalah pemain dapat menggabungkan kedua elemen dengan melakukan *drag and drop*, jika kombinasi kedua elemen valid, akan memunculkan elemen baru, jika kombinasi tidak valid maka tidak akan terjadi apa-apa. Permainan ini tersedia di *web browser*, Android atau iOS

Pada Tugas Besar pertama Strategi Algoritma ini, mahasiswa diminta untuk menyelesaikan permainan Little Alchemy 2 ini dengan menggunakan **strategi Depth First Search dan Breadth First Search**.

Komponen-komponen dari permainan ini antara lain:

1. Elemen dasar

Dalam permainan Little Alchemy 2, terdapat 4 elemen dasar yang tersedia yaitu *water*, *fire*, *earth*, dan *air*, 4 elemen dasar tersebut nanti akan di-*combine* menjadi elemen turunan yang berjumlah 720 elemen.



Gambar 2. Elemen dasar pada Little Alchemy 2

2. Elemen turunan

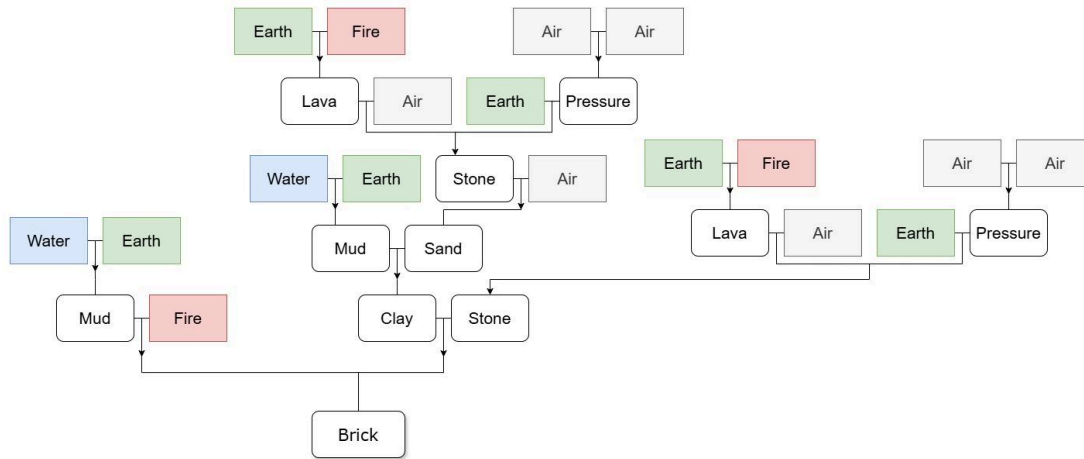
Terdapat 720 elemen turunan yang dibagi menjadi beberapa *tier* tergantung tingkat kesulitan dan banyak langkah yang harus dilakukan. Setiap elemen turunan memiliki *recipe* yang terdiri atas elemen lainnya atau elemen itu sendiri.

3. *Combine Mechanism*

Untuk mendapatkan elemen turunan pemain dapat melakukan *combine* antara 2 elemen untuk menghasilkan elemen baru. Elemen turunan yang telah didapatkan dapat digunakan kembali oleh pemain untuk membentuk elemen lainnya.

Spesifikasi Wajib

- Buatlah aplikasi pencarian *recipe* elemen dalam permainan Little Alchemy 2 dengan menggunakan strategi **BFS dan DFS**.
- Tugas dikerjakan berkelompok dengan anggota **minimal 2 orang** dan **maksimal 3 orang**, boleh lintas kelas dan lintas kampus.
- Aplikasi berbasis **web**, untuk *frontend* dibangun menggunakan bahasa **Javascript** dengan *framework* **Next.js atau React.js**, dan untuk *backend* menggunakan bahasa **Golang**.
- Untuk *repository frontend* dan *backend* diperbolehkan **digabung** maupun **dipisah**.
- Untuk data elemen beserta resep dapat diperoleh dari **scraping** [website Fandom Little Alchemy 2](https://www.fandom.com/wiki/Little_Alchemy_2).
- Terdapat opsi pada *aplikasi* untuk **memilih algoritma** BFS atau DFS (juga *bidirectional* jika membuat bonus)
- Terdapat *toggle button* untuk memilih untuk menemukan **sebuah recipe** (Silahkan yang mana saja) ~~terpendek (output dengan rute terpendek)~~ atau **mencari banyak recipe (multiple recipe)** menuju suatu elemen tertentu. Apabila pengguna ingin mencari banyak *recipe* maka terdapat cara bagi pengguna untuk **memasukkan parameter banyak recipe** maksimal yang ingin dicari. Aplikasi boleh mengeluarkan *recipe* apapun asalkan berbeda dan memenuhi banyak yang diinginkan pengguna (apabila mungkin).
- Mode pencarian *multiple recipe* wajib dioptimasi menggunakan **multithreading**.
- Elemen yang digunakan pada suatu *recipe* harus berupa elemen dengan **tier lebih rendah** dari elemen yang ingin dibentuk.
- Aplikasi akan **memvisualisasikan recipe** yang ditemukan sebagai sebuah *tree* yang menunjukkan kombinasi elemen yang diperlukan dari elemen dasar. Agar lebih jelas perhatikan contoh berikut



Gambar 3. Contoh visualisasi *recipe* elemen

Gambar diatas menunjukkan contoh visualisasi *recipe* dari elemen *Brick*. Setiap elemen bersebelahan menunjukkan elemen yang perlu dikombinasikan. Amati bahwa *leaf* dari *tree* selalu berupa elemen dasar. Apabila dihitung, gambar diatas menunjukkan 5 buah *recipe* untuk *Brick* (karena *Brick* dapat dibentuk dengan kombinasi *Mud+Fire* atau *Clay+Stone*, begitu pula *Stone* yang dapat dibentuk oleh kombinasi *Lava+Air* atau *Earth+Pressure*). Visualisasi pada aplikasi tidak perlu persis seperti contoh diatas, tetapi pastikan bahwa *recipe* ditampilkan dengan jelas.


- Aplikasi juga menampilkan **waktu pencarian** serta **banyak node** yang dikunjungi.

Spesifikasi Bonus


- **(maks 5)** Membuat video tentang aplikasi BFS dan DFS pada permainan Little Alchemy 2 di Youtube. Video dibuat harus memiliki audio dan menampilkan wajah dari setiap anggota kelompok. Untuk contoh video tubes stima tahun-tahun sebelumnya dapat dilihat di Youtube dengan kata kunci “Tubes Stima”, “strategi algoritma”, “Tugas besar stima”, dll. **Semakin menarik video, maka semakin banyak poin yang diberikan.**
- **(maks 3)** Aplikasi dijalankan menggunakan **Docker** baik untuk *frontend* maupun *backend*.
- **(maks 3)** Aplikasi **di-deploy** ke aplikasi *deployment* (aplikasi *deployment* bebas) agar bisa diakses secara daring
- **(maks 3)** Menambahkan algoritma bukan hanya DFS dan BFS, tetapi juga [strategi bidirectional](#)
- **(maks 6)** Aplikasi memiliki fitur **Live Update** visualisasi *recipe* selama proses pencarian. *Tree* visualisasi akan dibangun bertahap secara **real time** sesuai

dengan progress pencarian. Tambahkan **delay** pada Live Update karena pencarian dapat berjalan dengan sangat cepat.

- Jika terdapat kesulitan selama mengerjakan tugas besar sehingga memerlukan bimbingan, maka dapat melakukan asistensi tugas besar kepada asisten (opsional). Dengan catatan asistensi hanya bersifat membimbing, bukan memberikan “jawaban”.
- Terdapat demo dengan asisten untuk mendemonstrasikan aplikasi yang telah dibuat. Pengumuman mengenai demo akan diberitahukan lebih lanjut oleh asisten.
- Setiap kelompok harap mengisi nama kelompok dan anggotanya pada link berikut, paling lambat **Kamis, 24 April pukul 22.11 WIB**.

 [Pendataan Kelompok Tubes 2 Stima 2024/2025](#)

- Diwajibkan untuk memilih asisten meskipun tidak melakukan asistensi, karena asisten yang dipilih akan menjadi asisten saat asistensi (opsional) dan demo tugas besar. Pemilihan asisten dapat dilakukan pada link berikut, paling lambat **Kamis, 24 April 2025 pukul 22.11 WIB**.

 [Pendataan Kelompok Tubes 2 Stima 2024/2025](#)

- Program disimpan dalam repository yang bernama **Tubes2_NamaKelompok** (bila digabung) dan **Tubes2_FE/BE_NamaKelompok** (bila dipisah) dengan nama kelompok sesuai dengan yang di *sheets* diatas. Berikut merupakan struktur dari isi repository tersebut:
 - a. Folder src berisi **program yang dapat dijalankan**
 - b. Folder doc berisi laporan tugas besar dengan format **NamaKelompok.pdf**
 - c. README untuk tata cara penggunaan yang minimal berisi:
 - i. Penjelasan singkat algoritma DFS dan BFS yang diimplementasikan
 - ii. Requirement program dan instalasi tertentu bila ada
 - iii. Command atau langkah-langkah dalam meng-compile atau build program
 - iv. Author (identitas pembuat)
- Sangat disarankan untuk menggunakan [semantic commit](#). Buatlah release dengan format **v1.x** dengan x adalah nomor revisi dimulai dari revisi 0. Contoh v1.0 untuk release pertama, v1.1 untuk revisi selanjutnya.
- Pastikan untuk membuat repository bersifat **Public** paling lambat **H+1 deadline (1 hari setelah deadline)**. Sebelum deadline repository harus bersifat **Private**.
- Laporan dikumpulkan hari **Selasa, 13 Mei 2025** pada alamat Google Form berikut paling lambat **pukul 22.11 WIB**:

<https://bit.ly/tubes2stima25>

PERINGATAN: Keterlambatan akan mengurangi nilai sebanyak 1 poin untuk setiap menit keterlambatan.

- Adapun pertanyaan terkait tugas besar ini bisa disampaikan melalui QnA berikut: <https://bit.ly/QnA-Stima-25>.

Isi laporan

- **Cover:** Cover laporan ada foto anggota kelompok (foto bertiga). Foto ini menggantikan logo “gajah” ganesha.
- **Bab 1:** Deskripsi tugas (dapat menyalin spesifikasi tugas ini).
- **Bab 2:** Landasan Teori.
 - Dasar teori (Penjelajahan Graf serta algoritma *Breadth First Search* dan *Depth First Search* secara umum serta bonus).
 - Penjelasan singkat mengenai aplikasi web yang dibangun.
- **Bab 3:** Analisis Pemecahan Masalah.
 - Langkah-langkah pemecahan masalah.
 - Proses pemetaan masalah menjadi elemen-elemen algoritma DFS dan BFS.
 - Fitur fungsional dan arsitektur aplikasi web yang dibangun.
 - Contoh ilustrasi kasus.
- **Bab 4:** Implementasi dan pengujian.
 - Spesifikasi teknis program (struktur data, fungsi, dan prosedur yang dibangun).
 - Penjelasan tata cara penggunaan program (*interface* program, fitur-fitur yang disediakan program, dan sebagainya).
 - Hasil pengujian **minimal 3** buah elemen dalam bentuk *screenshot* antarmuka. Variasikan setiap kasus dengan berbagai fitur yang diimplementasikan (Algoritma serta banyak *recipe* yang dicari).
 - Analisis hasil pengujian.
- **Bab 5:** Kesimpulan, Saran, dan Refleksi tentang Tugas Besar 2.
- **Lampiran:** Tautan *repository* GitHub (dan video jika membuat)
- **Daftar Pustaka**

Keterangan laporan:

1. Laporan ditulis dalam bahasa Indonesia yang baik dan benar.
2. Laporan mengikuti format pada *section* “Isi laporan” dengan baik dan benar.
3. Identitas per halaman harus jelas (misalnya : halaman, kode kuliah).

Penilaian

1. **Bagian 1: Laporan (40%)**

- a. Langkah-langkah pemecahan masalah, proses pemetaan masalah, fitur fungsional dan arsitektur aplikasi web yang dibangun, dan contoh ilustrasi kasus. (20 %)
- b. Spesifikasi teknis program (struktur data, fungsi, dan prosedur yang dibangun), penjelasan tata cara penggunaan program (interface program, fitur-fitur yang disediakan program, dan sebagainya), hasil pengujian (screenshot antarmuka dan variasi pengujian kasus berdasarkan fitur aplikasi), dan analisis hasil pengujian. (15 %)
- c. Kelengkapan komponen-komponen pada laporan dan README. (5 %)

2. Bagian 2: Implementasi Program dan Demo (60%)

- a. Kecocokan output dari test case yang dimiliki Asisten. (25 %)
- b. Kebenaran algoritma *Depth First Search* dan *Breadth First Search*, sesuai dengan apa yang telah diajarkan di kelas. (15%)
- c. Pemahaman tugas besar dan algoritma yang telah dibuat oleh masing-masing anggota. (10%)
- d. Keberhasilan input dan output, sesuai dengan komponen-komponen yang ada pada spesifikasi. (5%)
- e. Modularitas/keterbacaan penulisan program. (5%)

3. Bagian 3: Komponen Bonus (Maksimal 20 Poin)

- a. Program dapat menampilkan *Live Update* visualisasi secara *real time*. (**bonus** maksimal 6 poin)
- b. Aplikasi dapat melakukan pencarian dengan strategi Bidirectional. (**bonus** maksimal 3 poin)
- c. Aplikasi di-deploy dan dapat diakses melalui internet. (**bonus** maksimal 3 poin)
- d. Program dijalankan menggunakan Docker baik untuk *frontend* maupun *backend*. (**bonus** maksimal 3 poin)
- e. Bonus dalam membuat video kelompok. (**bonus** maksimal 5 poin)

Perhatian

- **Dilarang keras** *copy paste* program dari internet, AI, repository lain, ataupun program milik teman. Program harus dibuat sendiri, kecurangan akan diberikan sanksi berat yaitu nilai tugas menjadi **nol**.
- Pastikan program **dapat dikompilasi setidaknya** pada *windows* dan *linux*.
- Apabila program **tidak dapat dijalankan** maka tidak akan dinilai oleh asisten.
- **Keterlambatan pengumpulan akan mengurangi 1 poin untuk setiap menit keterlambatan.**
- Tambahkan tabel berikut yang diisi *checklist* (✓) pada bagian lampiran laporan Anda.

No	Poin	Ya	Tidak
1	Aplikasi dapat dijalankan.		
2	Aplikasi dapat memperoleh data <i>recipe</i> melalui scraping.		
3	Algoritma <i>Depth First Search</i> dan <i>Breadth First Search</i> dapat menemukan <i>recipe</i> elemen dengan benar.		
4	Aplikasi dapat menampilkan visualisasi <i>recipe</i> elemen yang dicari sesuai dengan spesifikasi.		
5	Aplikasi mengimplementasikan multithreading.		
6	Membuat laporan sesuai dengan spesifikasi.		
7	Membuat bonus video dan diunggah pada Youtube.		
8	Membuat bonus algoritma pencarian <i>Bidirectional</i> .		
9	Membuat bonus <i>Live Update</i> .		
10	Aplikasi di-containerize dengan Docker.		
11	Aplikasi di-deploy dan dapat diakses melalui internet.		

Referensi

“Little Alchemy 2”

<https://littlealchemy2.com/>

Web permainan Little Alchemy 2.

“All elements in Little Alchemy 2”

[https://little-alchemy.fandom.com/wiki/Elements_\(Little_Alchemy_2\)](https://little-alchemy.fandom.com/wiki/Elements_(Little_Alchemy_2))

Halaman wiki Little Alchemy 2 yang menampilkan seluruh kombinasi elemen (juga digunakan untuk scraping).

“Client-Server Architecture”

https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/First_steps/Client-Server_overview

Penjelasan arsitektur *Client-Server* yang umum digunakan pada pengembangan aplikasi *web*.

“What is an API?”

<https://aws.amazon.com/what-is/api/>

Penjelasan apa itu API bagi yang memerlukan.

“Next Documentation”

<https://nextjs.org/docs>

Dokumentasi *framework* Next.js untuk *Front end*.

“React Documentation”

<https://go.dev/doc/>

Dokumentasi *framework* React.js untuk *Front end*.

“Golang Documentation”

<https://go.dev/doc/>

Dokumentasi bahasa Go untuk *Back end*.

“Gin”

<https://gin-gonic.com/>

Framework *backend* dalam bahasa Go.

“goquery”

<https://github.com/PuerkitoBio/goquery>

Kakas web scraping menggunakan bahasa Go.

“Effective Go Concurrency”

https://go.dev/doc/effective_go#concurrency

Dokumentasi Golang untuk mempelajari multithreading secara efektif.

--- Selamat Mengerjakan! ---

"Bes bes bes bes bes bes bes tubeess, anomali mengerikan yang menghantui mahasiswa ITB. Konon katanya kalau ada orang yang tidak mengerjakan tubes 3 hari

berturut-turut, maka mahluk ini datang di rumah kalian. Hiiiiii seremnyaa. Anomali ini sering kali muncul ketika kalian mau rebahan. Share ke teman kalian yang malas mengerjakan tubes."

— Farhan —

"Bisa lah 3 hari selesai"

— Ciko —

"Kenapa waktu *combine* kemiskinan dan penderitaan hasilnya Indonesia, Bang?"

— Haikal —

"Let him Cook!"

— Ariel —

"Kurang kurangin nge prompt—"

— Rafi —

"Chicken Jockey 🐔🐔👦👦"

— Aland —

"Avatar"

— Eka —

"BASEBALL HUH"

— Ikhwan —