

Aplikasi Algoritma *Branch and Bound* di Permainan *Wikipedia Speedrun*

Muhammad Alif Putra Yasa - 13520135

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520135@std.stei.itb.ac.id

Abstract—Wikipedia merupakan ensiklopedia multibahasa daring yang memiliki lebih dari 6 juta artikel berbahasa Inggris dan 55 juta artikel dalam 309 bahasa. *Speedrun* adalah upaya untuk menyelesaikan suatu permainan atau bagian dari permainan dalam waktu yang sesingkat-singkatnya. *Wikipedia Speedrun* merupakan permainan dimana pemain-pemain memilih dua artikel di Wikipedia kemudian mencoba untuk mencapai salah satu artikel hanya menggunakan *hyperlink* di artikel lainnya. Sifat dari permainan ini dapat dimodelkan dengan struktur data pohon, sehingga mungkin untuk mendapatkan rute dari satu artikel ke artikel lain menggunakan algoritma traversal pohon.

Keywords—Wikipedia; *Speedrun*; *Branch and Bound*;

I. PENDAHULUAN

Wikipedia merupakan ensiklopedia daring yang didirikan pada tahun 2001 oleh Jimmy Wales dan Larry Sanger. Wikipedia awalnya bermula sebagai proyek pelengkap dari Nupedia, ensiklopedia daring yang artikelnya ditulis oleh pakar-pakar dan ditinjau secara ketat dengan prosedur yang formal. Kedua Nupedia dan Wikipedia beroperasi bersama-sama sampai tahun 2003 dimana server Nupedia berhenti dan kemudian artikel-artikel Nupedia digabung dengan Wikipedia.

Wikipedia sekarang memiliki 6 juta artikel berbahasa Inggris dan 55 juta artikel dalam 309 bahasa yang berbeda. Permainan *Wikipedia Speedrun* muncul sebagai akibat dari banyaknya dan saling terlibatnya antar satu sama lain artikel di Wikipedia.

Speedrun merupakan upaya untuk menyelesaikan suatu atau sebagian dari permainan, biasanya *video game*, dalam waktu sesingkat-singkatnya. Di Permainan *Wikipedia Speedrun*, pemain menentukan dua artikel di Wikipedia, salah satu menjadi artikel awal dan yang lain menjadi artikel akhir. Tujuan dari permainan ini adalah untuk mencapai artikel akhir hanya dengan menggunakan *hyperlink* yang terdapat pada artikel awal dan artikel yang dibuka menggunakan *hyperlink* sebelumnya.

Artikel-artikel di Wikipedia yang dihubungkan dengan *hyperlink* ini dapat dimodelkan menggunakan sebuah graf berarah, dimana simpul merupakan artikel dan panah menggambarkan hubungan keduanya berdasarkan *hyperlink* di masing-masing artikel.

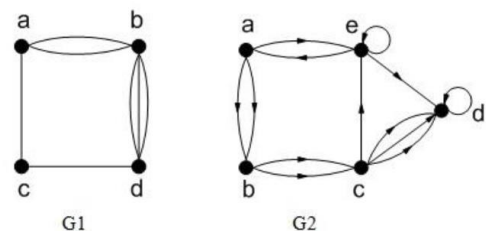
II. LANDASAN TEORI

A. Graf

Graf merupakan struktur matematis yang digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antar objek-objek tersebut.

Graf $G = (V, E)$ merupakan graf dengan V merupakan himpunan tidak kosong dari simpul-simpul dan E merupakan himpunan sisi yang menghubungkan sepasang simpul.

Berdasarkan orientasi arah pada sisi graf, graf dapat dibagi menjadi dua, Graf berarah dan Graf tidak berarah.



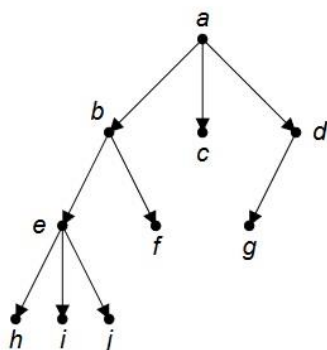
Gambar 2.1 G1: Graf Tidak Berarah; G2: Graf Berarah
Sumber: <https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf>

B. Pohon

Pohon merupakan graf tidak berarah terhubung yang tidak mengandung sirkuit. Misalkan $G = (V, E)$ adalah graf tidak berarah sederhana dengan jumlah simpul n . Maka, pohon memenuhi syarat-syarat sebagai berikut:

1. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
2. G terhubung dan memiliki $m = n - 1$ sisi.
3. G tidak mengandung sirkuit dan memiliki $m = n - 1$ sisi.
4. G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
5. G terhubung dan semua sisinya adalah jembatan.

Pohon yang satu buah simpulnya diperlakukan sebagai akar dan sisi-sisinya diberi arah sehingga menjadi graf berarah dinamakan pohon berakar (*rooted tree*).



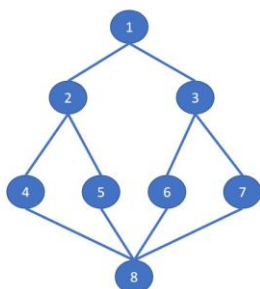
Gambar 2.2 Pohon dengan *a* sebagai akar

Sumber: <https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag2.pdf>

C. Algoritma Breadth First Search

Algoritma Breadth First Search merupakan salah satu algoritma traversal graf, yaitu algoritma untuk mengunjungi simpul secara sistematis. Berikut merupakan Langkah-langkah pada algoritma Breadth First Search dengan asumsi bahwa traversal dimulai dari simpul *v*:

1. Kunjungi simpul *v*.
2. Kunjungi semua simpul yang bertetangga dengan *v*.
3. Kunjungi simpul yang belum dikunjungi tetapi bertetangga dengan simpul-simpul sebelumnya.



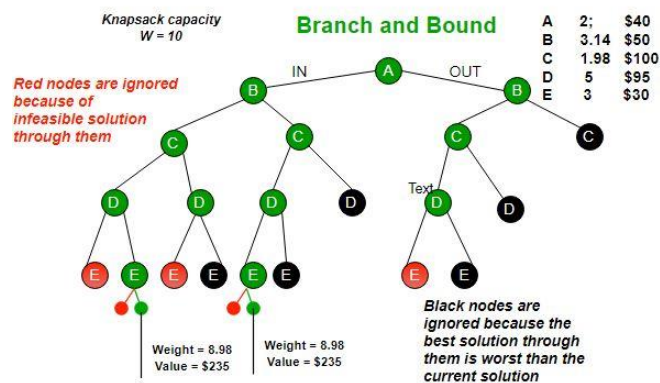
Iterasi	V	Q	dikunjungi							
			1	2	3	4	5	6	7	8
Inisialisasi	1	{1}	T	F	F	F	F	F	F	F
Iterasi 1	1	{2,3}	T	T	F	F	F	F	F	F
Iterasi 2	2	{3,4,5}	T	T	T	T	F	F	F	F
Iterasi 3	3	{4,5,6,7}	T	T	T	T	T	F	F	F
Iterasi 4	4	{5,6,7,8}	T	T	T	T	T	T	T	F
Iterasi 5	5	{6,7,8}	T	T	T	T	T	T	T	T
Iterasi 6	6	{7,8}	T	T	T	T	T	T	T	T
Iterasi 7	7	{8}	T	T	T	T	T	T	T	T
Iterasi 8	8	{}	T	T	T	T	T	T	T	T

Urutan simpul2 yang dikunjungi: 1, 2, 3, 4, 5, 6, 7, 8

Gambar 2.3 Ilustrasi algoritma BFS

Sumber: <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag1.pdf>

D. Algoritma Branch and Bound



Gambar 2.4 Ilustrasi algoritma Branch and Bound

Sumber: <https://www.geeksforgeeks.org/branch-and-bound-algorithm/>

Algoritma Branch and Bound merupakan algoritma yang digunakan pada persoalan optimalisasi, dimana diinginkan nilai minimal atau maksimal suatu fungsi objektif, yang tidak melanggar *constraint* persoalan. Algoritma ini merupakan gabungan dari algoritma *Breadth First Search* dan *Least Cost Search*

Pada implementasinya, algoritma Branch and Bound dapat dibagi menjadi dua bagian, yaitu:

1. Branching

Proses ini merupakan proses pembangkitan simpul-simpul baru dari pohon yang sudah dibuat.

2. Bounding

Proses ini merupakan proses pemangkasan simpul-simpul yang dianggap tidak mengarah pada solusi. Kriteria simpul ini adalah:

- a. Nilai simpul tidak lebih baik dari nilai terbaik.
- b. Simpul tidak merepresentasikan solusi yang feasible karena ada batasan yang dilanggar.
- c. Solusi hanya ada satu.

Berikut merupakan Langkah-langkah umum algoritma Branch and Bound:

1. Masukkan simpul akar ke dalam antrian Q. Jika simpul akar adalah simpul solusi (goal node), maka solusi telah ditemukan. Jika hanya satu solusi yang diinginkan, maka stop.
2. Jika Q kosong, Stop.
3. Jika Q tidak kosong, pilih dari antrian Q simpul *i* yang mempunyai nilai cost paling kecil. Jika terdapat beberapa simpul *i* yang memenuhi, pilih satu secara sembarang.
4. Jika simpul *i* adalah simpul solusi, berarti solusi sudah ditemukan. Jika satu solusi yang diinginkan, maka stop. Pada persoalan optimasi dengan pendekatan least cost search, periksa cost semua simpul hidup.

Jika cost nya lebih besar dari cost simpul solusi, maka matikan simpul tersebut.

5. Jika simpul *i* bukan simpul solusi, maka bangkitkan semua anak-anaknya. Jika *i* tidak mempunyai anak, kembali ke langkah 2.
6. Untuk setiap anak *j* dari simpul *i*, hitung cost-nya, dan masukkan semua anak-anak tersebut ke dalam *Q*.
7. Kembali ke langkah 2.

III. IMPLEMENTASI

A. Node Wikipedia

Sebelum mengimplementasikan algoritma, penting untuk membuat terlebih dahulu representasi graf dari artikel Wikipedia. Setiap artikel Wikipedia merupakan sebuah node yang menyimpan informasi tentang nama url, judul artikel, dan informasi node-node anaknya. Setiap node juga menyimpan reference ke node parent masing-masing. Berikut merupakan sebagian dari implementasi node Wikipedia yang berada di class WikiNode menggunakan Bahasa Python:

```
class WikiNode(object):
    def __init__(
        self, url : str,
        title : str = "",
        visited_links = None,
        parent = None
    ):
        self.url : str = url
        self.title : str = title

        if visited_links == None:
            self.memo = {}
        else:
            self.memo = visited_links

        self.parent = parent
        self.children : set(WikiNode) = set()
```

Selain itu, WikiNode juga memiliki fungsi-fungsi tambahan seperti `__str__`, `__hash__`, `__eq__`, dan lainnya untuk mempermudah objek lain dalam menggunakan kelas ini.

Kelas juga memiliki metode `parse_page` yang merupakan metode yang menggunakan url yang telah disimpan kemudian mengisi atribut `title` dan `children` sesuai dengan artikelnya. Berikut merupakan implementasi singkat dari fungsi `parse_page` dalam Bahasa Python:

```
def parse_page(self):
    """
    Mem-parse halaman,
    self.title diisi dengan judul artikel
    self.children berisi Node-node anak
    self.memo berisi node yang sudah
    dikunjungi untuk mempercepat kode
    (Memoization)
    """
    PAGE_RAW = requests.get(self.url)
    self.url = PAGE_RAW.url
    # Request Sukses
    if PAGE_RAW.status_code == 200:
        soup = BeautifulSoup(
            PAGE_RAW.content,
```

```
            "html.parser"
        )
        # Mengambil judul artikel
        self.title = soup.find(
            "h1", id="firstHeading"
        ).get_text()

        body = soup.find("div", id="bodyContent")

        # Mencari link text
        for t in body.find_all(
            lambda t: t.name == 'a' and
            t.get_text(strip=True) != ""
        ):
            link_url = t.get("href")
            if link_url and re.match(
                u"\\/wiki\\/\\w+$", link_url
            ):
                full_url =
                    "https://en.wikipedia.org"
                    + link_url
                if full_url not in self.memo:
                    self.memo[full_url] =
                        WikiNode(
                            full_url,
                            visited_links=self.memo,
                            parent=self,
                            title=t.get_text()
                        )
                page = self.memo[full_url]
                self.children.add(page)
            else:
                print("Error")
```

B. Algoritma Branch and Bound

Implementasi Algoritma Branch and Bound akan memanfaatkan kelas WikiNode sebelumnya. Selain struktur general dari algoritma, beberapa aspek penting lainnya adalah fungsi heuristik atau fungsi untuk mencari nilai cost suatu simpul.

Pada Implementasi ini, fungsi heuristik yang digunakan adalah fungsi `similarity` di library `spacy`. Fungsi ini akan me-return tingkat kemiripan dua judul secara semantis. Alasan dipilihnya fungsi heuristik ini adalah untuk meniru upaya manusia ketika bermain permainan ini. Pada umumnya, pemain memilih halaman yang akan mereka lalui berdasarkan tingkat kedekatan artikel tersebut dengan artikel tujuan. Contohnya, jika ingin mencapai artikel *tent* atau tenda, akan lebih masuk akal untuk memilih artikel *cave* atau gua daripada *knife* atau pisau. Library ini menggunakan Natural Language Processing untuk memperoleh jarak semantic dari kedua judul artikel.

Berikut merupakan *snippet* dari fungsi heuristik yang digunakan pada implementasi ini:

```
def h_func_1(txt1: WikiNode, txt2: WikiNode) ->
float:
    global nlp
    txt_1 = nlp(txt1.title)
    txt_2 = nlp(txt2.title)
    if(txt1.title == "" or txt2.title == ""):
        print(txt1.url, txt2.url)
    return round(
        abs(1 - txt_1.similarity(txt_2)), 5
    )
```

Fungsi ini menerima dua WikiNode dan menghitung tingkat kemiripan antara judul artikel sekarang dan judul artikel tujuan.

Berikut merupakan implementasi singkat dari kelas implementasi algoritma, yaitu kelas BnBSolver:

```
class BnBSolver(object):
    def __init__(self, start: str, end: str, h_func) -> None:

        self.start = WikiNode(start)
        self.start.parse_page()

        self.end = WikiNode(end)
        self.end.parse_page()

        self.heuristik = h_func
        self.prio_queue = PriorityQueue()
        self.opened = {}
        self.enqueue(self.start)
        self.visit_counter = 0

    def enqueue(self, node) -> None:
        if node.url not in self.opened:
            self.opened[node.url] = False

        if not self.opened[node.url]:
            rating = self.heuristik(node, self.end)
            self.prio_queue.put((rating, node))

    def dequeue(self) -> WikiNode:
        res = self.prio_queue.get()
        self.opened[res[1].url] = True
        return res

    def solve(self):
        if self.prio_queue.empty():
            return None
        else:
            curr = self.dequeue()
            curr_page = curr[1]

            self.visit_counter += 1
            print(f"[{self.visit_counter}] Visiting {curr_page.title} at {curr_page.url}")

            if(curr_page == self.end):
                return curr_page
            else:
                curr_page.parse_page()
                for curr_child in curr_page.children:
                    self.enqueue(curr_child)
                return self.solve()
```

IV. EKSPERIMEN

Percobaan dilakukan menggunakan tiga sampel pasangan artikel yang memiliki topik yang berhubungan dekat dan jauh. Berikut merupakan pasangan artikel yang akan di uji cobakan:

1. [Banana](#) ke [Knife](#)
2. [Spoon](#) ke [Cartoons](#)
3. [Coconut](#) ke [Tent](#)

A. Uji Coba Pertama

Uji Coba pertama adalah untuk mencapai artikel [Knife](#) dari [Banana](#). Kedua artikel terhubung dengan proses memasak. Berikut merupakan output program:

```
malify@malify: /mnt/c/Users/malif/Projects/wikipedia-speedrun-solver$ python3 BnBSolver.py
SOLVING ==
[1] Visiting Banana at https://en.wikipedia.org/wiki/Banana
[2] Visiting Lady Finger at https://en.wikipedia.org/wiki/Lady_Finger_banana
[3] Visiting Cooking banana at https://en.wikipedia.org/wiki/Cooking_banana
[4] Visiting Cooking banana at https://en.wikipedia.org/wiki/Cooking_banana
[5] Visiting cooking at https://en.wikipedia.org/wiki/Cooking
[6] Visiting Knife at https://en.wikipedia.org/wiki/Knife
== PATH ==
Banana [https://en.wikipedia.org/wiki/Banana]
Cooking [https://en.wikipedia.org/wiki/Cooking]
Knife [https://en.wikipedia.org/wiki/Knife]
```

Gambar 4.1.1 Hasil Percobaan Pertama

Sumber: Arsip Pribadi

Berikut merupakan peragaan ulang jalur hasil:

Developing countries

Bananas and plantains constitute a major staple **food crop** for millions of people in **developing countries**. In many tropical countries, green (unripe) bananas used for **cooking** represent the main cultivars. Most producers are small-scale **farmers** either for home consumption or local markets. Because bananas and plantains produce fruit year-round, they provide a valuable food source during the **hunger season** (when the food from one annual/semi-annual harvest has been consumed, and the next is still to come). Bananas and plantains are important for global **food security**.^[98]

Gambar 4.1.2 Link Cooking pada artikel Banana

Sumber: Arsip Pribadi

Prehistoric technology		Index
Prehistory (Timeline · Outline · Stone Age · Subdivisions · New Stone Age) · Technology (History · Glossary)		
Tools		Index
Farming	Neolithic Revolution (Founder crops · New World crops) · Airl · Juggah · Coll · Digging stick · Domestication · Good · Irrigation · Secondary products · Sickle · Terracing	
Food processing	Fire · Basket · Cooking (Earth oven) · Granaries · Grinding slab · Ground stone · Hearth (Hqk · Hqk · Qesem cave) · Maize · Mallet · Mortar and pestle · Pottery · Quern stone · Storage pit · Arrow · Boomerang (throwing stick) · Bow and arrow (History) · Nets · Spear (Spear-thrower · Sabin · Harpoon · Vomer · Quernstone spurs)	
Hunting	Projectile points · Arrowhead · Bare Island · Cascade · Clovis · Crossed · Cumberland · Eden · Folsom · Llanos · Maria Site · Plano · Transverse arrowhead	
Systems	Game drive system (Buffalo jump)	
Early farming (Chalcolithic · Neolithic · Mesolithic)	Agriculture · Clovis culture · Cupstone · Fire hardening · Gravettian culture · Halft · Hand axe (Glossary) · Langdale axe industry · Levallois technique · Lithic core · Lithic reduction (analysis · debitage · flake) · Lithic technology · Magdalenian culture · Metallurgy · Microblade technology · Mining · Prepared-core technique · Solihull industry · Stirling platform · Tool stone · Uniface · Vulture technique	
Other tools	Adze · Axl (stone) · Axl · Bannockburn · Beale (instrument) · Bone tool · Bone drill · Buret · Carver (Clay · Plastic carver) · Chopper (tool) · Chisels · Double-flint tool · Fine plough · Fine saw · Hammerstone · Knife · Moccasin · Quern stone · Ratchet · Rope · Scraper (Index) · Stone tool · Slaty stick · Weapons · Wheel (Illustration)	
Architecture		Index
Arts and culture		Index

Gambar 4.1.3 Link Knife pada artikel Cooking

Sumber: Arsip Pribadi

B. Uji Coba Kedua

Uji coba kedua adalah untuk mencapai artikel [Cartoons](#) dari artikel [Spoon](#). Kedua artikel ini memiliki hubungan yang cukup jauh sehingga program akan mengalami kesusahan dibandingkan dengan uji coba pertama sebelumnya. Berikut merupakan output program:

```
malify@malify: /mnt/c/Users/malif/Projects/wikipedia-speedrun-solver$ python3 BnBSolver.py
SOLVING ==
[1] Visiting Spoon at https://en.wikipedia.org/wiki/Spoon
[2] Visiting British Museum at https://en.wikipedia.org/wiki/British_Museum
[3] Visiting The Burlington House Cartoon at https://en.wikipedia.org/wiki/The_Virgin_and_Child_with_St_Anne_and_St_John_the_Baptist
[4] Visiting cartoon at https://en.wikipedia.org/wiki/Modello
[5] Visiting Raphael Cartoons at https://en.wikipedia.org/wiki/Raphael_Cartoons
[6] Visiting drawings at https://en.wikipedia.org/wiki/Drawings
[7] Visiting animation at https://en.wikipedia.org/wiki/Traditional_animation
[8] Visiting Animated cartoon at https://en.wikipedia.org/wiki/Animated_cartoon
[9] Visiting cartoons at https://en.wikipedia.org/wiki/Cartoons
== PATH ==
Spoon [https://en.wikipedia.org/wiki/Spoon]
British Museum [https://en.wikipedia.org/wiki/British_Museum]
Drawing [https://en.wikipedia.org/wiki/Drawings]
Traditional animation [https://en.wikipedia.org/wiki/Traditional_animation]
Animation [https://en.wikipedia.org/wiki/Animated_cartoon]
Cartoons [https://en.wikipedia.org/wiki/Cartoons]
```

Gambar 4.2.1 Hasil output program untuk uji coba ketiga

Sumber: Arsip Pribadi

Berikut merupakan peragaan ulang jalur hasil:

History

Preserved examples of various forms of spoons used by the **ancient Egyptians** include those **symbols**.^[1] During the Neolithic Ozieri civilization in **Sardinia**, ceramic **ladles** and spoons were **bronze** spoons in China were designed with a sharp point, and may have also been used as the form of a spike or pointed stem.^[1] There are many examples in the **British Museum** from bowl with the handle.^[1]

Gambar 4.2.2 Link British Museum di Spoon

Sumber: Arsip Pribadi

- Large **Cloisonné** jar with dragon made for the **Ming Dynasty** Imperial Court, pair
- Pair of ceramic **Kakiemon elephants** from Japan, (17th century AD)
- **Moon jar** from the **Joseon Dynasty** collected by the potter **Bernard Leach**, Korea
- Japanese prints including **The Great Wave off Kanagawa**, (1829–32 AD)
- Illustrations for the Great Picture Book of Everything, rare album of **drawings** by

Gambar 4.2.3 Link *Drawing* di *British Museum*
Sumber: Arsip Pribadi

A drawing instrument releases a small amount of material onto a surface, leaving a visible mark. The such as **cardboard**, **wood**, plastic, **leather**, **carvas**, and **board**, have been used. Temporary drawings popular and fundamental means of public expression throughout human history. It is one of the simp availability of drawing instruments makes drawing one of the most common artistic activities.

In addition to its more artistic forms, drawing is frequently used in commercial **illustration**, **animation**,

Gambar 4.2.4 Link *Traditional Animation* di *Drawing*
Sumber: Arsip Pribadi

See also [edit]

- **History of animation**
- **Animated cartoon**
- **Computer generated imagery**
- **Stop motion**

Gambar 4.2.4 Link *Animated Cartoon* di *Traditional Animation*
Sumber: Arsip Pribadi

Émile Cohl's *Fantasmagorie* (1908) is the oldest known e films were created by **Ladislas Starevich** with his puppet and **Gertie the Dinosaur** (1914).

During the 1910s, the production of animated **"cartoons"**

Gambar 4.2.4 Link *Cartoons* di *Animated Cartoon*
Sumber: Arsip Pribadi

C. Uji Coba Ketiga

Uji coba ketiga adalah untuk mencapai artikel **Tent** dari artikel **Ceramic**. Kedua artikel ini memiliki hubungan yang jauh sehingga program akan mengalami kesusahan dibandingkan dengan uji coba kedua sebelumnya. Berikut merupakan output program:

```
malifpy@malifpy: /mnt/c/Users/malif/Projects/wikipedia-speedrun-solver$ python3 BnBSolver.py
== SOLVING ==
[1] Visiting Ceramic at https://en.wikipedia.org/wiki/Ceramic
[2] Visiting floor at https://en.wikipedia.org/wiki/Flooring
[3] Visiting Earthen floor at https://en.wikipedia.org/wiki/Earthen_floor
[4] Visiting large sheets at https://en.wikipedia.org/wiki/Sheet_vinyl_flooring
[5] Visiting wet rooms at https://en.wikipedia.org/wiki/Wet_room
[6] Visiting folding doors at https://en.wikipedia.org/wiki/Folding_screen
[7] Visiting folding doors at https://en.wikipedia.org/wiki/Folding_door
[8] Visiting Curtains at https://en.wikipedia.org/wiki/Curtain
[9] Visiting Sleeping at https://en.wikipedia.org/wiki/Sleep
[10] Visiting Sleeping bag at https://en.wikipedia.org/wiki/Sleeping_bag
[11] Visiting tent at https://en.wikipedia.org/wiki/Tent
== PATH ==
Ceramic [https://en.wikipedia.org/wiki/Ceramic]
Flooring [https://en.wikipedia.org/wiki/Flooring]
Sheet vinyl flooring [https://en.wikipedia.org/wiki/Sheet_vinyl_flooring]
Shower [https://en.wikipedia.org/wiki/Wet_room]
Curtain [https://en.wikipedia.org/wiki/Curtain]
Sleep [https://en.wikipedia.org/wiki/Sleep]
Sleeping bag [https://en.wikipedia.org/wiki/Sleeping_bag]
tent [https://en.wikipedia.org/wiki/Tent]
```

Gambar 4.3.1 Hasil output untuk percobaan ketiga
Sumber: Arsip Pribadi

Berikut merupakan peragaan ulang jalur hasil:

By usage [edit]

For convenience, ceramic products are usually divided into four main types; these are shown below with some examples:

1. Structural, including **bricks**, **pipes**, **floor** and **roof tiles**

Gambar 4.3.2 Link *Flooring* di *Ceramic*
Sumber: Arsip Pribadi

Vinyl flooring is available in **large sheets** or **pre-cut tiles**; the former is resilient. Some come with a pre-applied adhesive for peel-and-stick installation, others require adhesive to be spread on to the substrate.^[3]

Gambar 4.3.3 Link *Sheet Vinyl Flooring* di *Flooring*
Sumber: Arsip Pribadi

Functions [edit]

As it does not let water through, sheet flooring is suitable for kitchens and bathrooms. Some types can be used in **wet rooms**.

Gambar 4.3.4 Link *Shower* di *Sheet Vinyl Flooring*
Sumber: Arsip Pribadi

Shower and bathtub curtains

^{*}*Shower curtain* redirects here. For the physical phenomenon, see *shower-curtain effect*.

Curtains can be used in shower or bathtub enclosures with two main purposes: to provide privacy and to prevent water from flooding or spraying outside the shower or

Gambar 4.3.5 Link *Curtain* di *Shower*
Sumber: Arsip Pribadi

Curtains are often hung on the inside of a building's windows to block the passage of light. For instance, at **night** to aid **sleeping**, or to stop light from escaping outside the building (stopping people outside from being able to see inside, often for privacy reasons). In this application, they are also known as "draperies". Curtains hung over a **doorway** are known as **portières**.^[2] Curtains come in a variety of shapes, materials, sizes, colors, and patterns. They often have their own sections within **department stores**, while some shops are completely dedicated to selling curtains.

Gambar 4.3.6 Link *Sleeping* di *Curtain*
Sumber: Arsip Pribadi

Daily life [edit]

Bed (Blank bed - Daybed - Four-poster bed - Futon - Hammock - Mattress - **Sleeping bag**) - Bed bug - Bedding - Bedroom - Bedtime - Bedtime story - Biphasic and polyphasic sleep - Chronotype - Comfort object - Dream diary - Microsleep - Mouth breathing - Nap - Nightwear - Power nap - Second wind - Siesta - Sleep and creativity - Sleep and learning - Sleep deprivation - Sleep debt - Sleeping while on duty - Sleepover - Snoring

Gambar 4.3.7 Link *Sleeping Bag* di *Sleeping*
Sumber: Arsip Pribadi

A **sleeping bag** is an insulated covering for a person, essentially a lightweight quilt that can be closed with a **zipper** or similar means to form a tube, which functions as lightweight, portable **bedding** in situations where a person is sleeping outdoors (e.g. when **camping**, **hiking**, **hii walking** or **climbing**). Its primary purpose is to provide warmth and **thermal insulation** through its synthetic or down insulation. It also typically has a water-resistant or water-repellent cover that protects, to some extent, against **wind chill** and light **precipitation**, but a **tent** is usually used in addition to a sleeping bag, as it performs those functions better. The bottom surface also provides some cushioning, but a **sleeping pad** or **camp cot** is usually used in addition for that purpose. The bottom surface of a sleeping bag may be moderately water repellent, but a plastic **tarp** or groundsheet is often used to protect against moist ground.

Gambar 4.3.8 Link *Tent* di *Sleeping Bag*
Sumber: Arsip Pribadi

V. KESIMPULAN

Pencarian solusi di permainan Wikipedia Speedrun dapat dilakukan dengan memodelkan Wikipedia sebagai pohon dan mengaplikasikan algoritma pencarian atau traversal graf pada model Wikipedia tersebut. Pada makalah ini, digunakan algoritma Branch and Bound untuk mencari jalur yang tepat untuk mencapai artikel tujuan.

Algoritma Branch and Bound bekerja dengan cara memilih untuk membuka artikel dengan judul yang mirip dengan judul artikel tujuan. Kemiripan ini ditentukan oleh fungsi heuristik similarity di library spacy.

Alternatif lain dari algoritma Branch and Bound untuk mendapatkan rute tersingkat adalah denan menggunakan algoritma Breadth First Search. Kendala dari menggunakan algoritma ini adalah karena setiap artikel memiliki banyak hyperlink, sehingga jumlah artikel yang harus di cek meningkat secara eksponensial setiap kenaikan kedalaman pohon.

Beberapa optimalisasi dapat dilakukan untuk meningkatkan performa pencarian menggunakan algoritma Branch and Bound, salah satunya adalah dengan menggunakan fungsi heuristik yang lebih bagus. Pemilihan node yang akan ditempuh ditentukan oleh node yang memiliki nilai heuristik

terkecil, sehingga fungsi heuristik yang lebih optimal dapat meningkatkan performa algoritma.

VIDEO LINK YOUTUBE

UCAPAN TERIMA KASIH

Puji syukur penulis panjatkan ke hadirat Allah SWT atas berkat rahmat dan karunia-Nya artikel berjudul “Aplikasi Algoritma Branch and Bound di Permainan Wikipedia Speedrun” dapat diselesaikan dengan tepat waktu.

Ucapan terima kasih penulis sampaikan kepada Bapak Dr. Ir. Rinaldi Munir, M.T. selaku dosen pengajar Mata Kuliah IF2211 Strategi Algoritma Kelas 03 yang telah membimbing mahasiswanya untuk memahami materi perkuliahan. Situs yang Beliau kelola juga menjadi salah satu sumber utama penulisan artikel ini karena terdapat banyak informasi dan dokumentasi yang terarsip.

Terakhir, penulis mengucapkan terima kasih kepada keluarga dan kolega yang telah membantu dalam mendukung penyelesaian makalah ini.

REFERENCES

- [1] Kock, Ned; Jung, Yusun; Syn, Thant (2016). "[Wikipedia and e-Collaboration Research: Opportunities and Challenges](#)". International Journal of e-Collaboration. IGI Global. 12 (2): 1–8. doi:10.4018/IJeC.2016040101. ISSN 1548-3681.
- [2] <https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf> (diakses pada 22 Mei 2022)
- [3] <https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag2.pdf> (diakses pada 22 Mei 2022)
- [4] <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag1.pdf> (diakses pada 23 Mei 2022)
- [5] <https://www.geeksforgeeks.org/branch-and-bound-algorithm/> (diakses pada 23 Mei 2022)

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, 23 Mei 2022



Muhammad Alif Putra Yasa
13520135