Optimalisasi Jadwal Rencana Perjalanan dengan Algoritma Greedy

Eunice Sarah Siregar - 13521013
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): eunicesarah011@gmail.com

***Abstract*—Merencanakan perjalanan merupakan hal yang wajib sebelum melakukan perjalanan. Pada umumnya rencana perjalanan dibuat untuk memaksimalkan tujuan dan efisiensi dari perjalanan tersebut. Untuk mencapai tujuan tersebut, dapat dioptimasi dengan algoritma greedy. Memiliki prinsip yang tamak, algoritma greedy menawarkan penjadwalan rencana perjalanan secara optimal. Beberapa alternatif yang dapat diimplementasikan, yaitu berdasarkan banyaknya tempat yang dikunjungi, biaya yang diperlukan, dan durasi selama perjalanan.**

Keywords—algoritma, greedy, perjalanan, penjadwalan, wisata

I. PENDAHULUAN

Penjadwalan merupakan proses merencanakan urutan aktivitas yang harus diproses berdasarkan waktu. Membuat jadwal adalah hal yang sangat penting terlebih sebelum melakukan perjalanan. Perjalanan yang direncanakan dengan baik akan membawa dampak yang positif untuk penggunanya, terlebih ketika perjalanan tersebut dikhususkan untuk berlibur.

Dalam era globalisasi dan kemajuan teknologi yang pesat, perjalanan telah menjadi bagian tak terpisahkan dari kehidupan manusia. Banyak orang melakukan perjalanan untuk berbagai tujuan, seperti bisnis, liburan, atau keperluan pribadi lainnya. Dalam konteks perjalanan, perencanaan jadwal yang efisien dan optimal memegang peranan penting dalam memaksimalkan pengalaman perjalanan, mengoptimalkan penggunaan sumber daya, dan menghemat waktu serta biaya.

Optimalisasi jadwal rencana perjalanan menjadi tantangan yang kompleks dan menarik dalam bidang optimasi. Tujuan utama dari optimalisasi jadwal rencana perjalanan adalah mengatur urutan kunjungan tempat dengan cara yang paling efisien, mengingat berbagai batasan dan preferensi yang ada. Oleh karena itu, dibutuhkan banyak pertimbangan untuk melakukan optimalisasi jadwal rencana perjalanan. Beberapa faktor yang perlu dipertimbangkan dalam perencanaan jadwal perjalanan antara lain jarak antar lokasi, waktu tempuh antar lokasi, preferensi pengguna, prioritas tujuan, biaya yang akan dikeluarkan, serta ketersediaan waktu dan sumber daya yang dimiliki.

Salah satu pendekatan yang telah banyak digunakan dalam masalah optimasi jadwal rencana perjalanan adalah menggunakan algoritma *greedy*. Algoritma Greedy adalah algoritma yang memilih langkah terbaik pada setiap tahap dengan harapan dapat mencapai solusi optimal secara keseluruhan. Dalam konteks perjalanan, algoritma Greedy banyak sekali digunakan untuk memilih urutan kunjungan tempat. Pada makalah ini, algoritma greedy diimplementasikan dengan diasumsikan bahwa memilih lokasi terdekat atau waktu tempuh terpendek pada setiap tahapan akan menghasilkan jadwal perjalanan yang optimal.

Penelitian terkait optimasi jadwal rencana perjalanan dengan menggunakan algoritma Greedy telah dilakukan. Smith et al. (2018) mengembangkan algoritma Greedy untuk mengoptimalkan rute perjalanan wisatawan di kota-kota besar. Mereka menggunakan data historis wisatawan dan informasi jarak antar lokasi untuk memilih urutan kunjungan yang optimal berdasarkan preferensi wisatawan. Penelitian lain oleh Chen et al. (2020) menggunakan algoritma Greedy untuk mengoptimalkan jadwal perjalanan bisnis dengan mempertimbangkan faktor-faktor seperti jarak, waktu, dan prioritas tugas.

Dalam konteks makalah ini, bertujuan untuk mengoptimalkan jadwal rencana perjalanan dengan menggunakan algoritma Greedy. Beberapa factor yang akan preferensi pengguna, dan prioritas tujuan perjalanan. Dengan menerapkan algoritma Greedy, berharap dapat menghasilkan jadwal perjalanan yang optimal dalam waktu yang efisien, meminimalkan waktu perjalanan dan memaksimalkan efisiensi penggunaan sumber daya.

Melalui penelitian ini, diharapkan akan ditemukan cara yang efisien untuk mengoptimalkan jadwal rencana perjalanan menggunakan algoritma Greedy. Hal ini dapat memberikan manfaat bagi individu dan perusahaan dalam merencanakan perjalanan mereka dengan lebih efisien, mengurangi biaya perjalanan dan waktu tempuh yang tidak efektif. Selain itu, hasil penelitian ini juga dapat menjadi landasan untuk pengembangan lebih lanjut dalam bidang optimisasi jadwal perjalanan, dengan mempertimbangkan faktor-faktor tambahan seperti preferensi pengguna yang lebih kompleks, pertimbangan cuaca, atau aspek lingkungan lainnya.

Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi penting dalam pengembangan metode

dan algoritma untuk optimisasi jadwal rencana perjalanan. Seiring dengan kemajuan teknologi, diharapkan bahwa hasil penelitian ini dapat diimplementasikan dalam aplikasi perencanaan perjalanan yang dapat digunakan oleh individu maupun perusahaan untuk mengoptimalkan pengalaman perjalanan mereka.

II. LANDASAN TEORI

A. Algoritma Greedy

Algoritma Greedy merupakan salah satu metode untuk memecahkan masalah dengan mengambil solusi terbaik untuk setiap iterasi dengan harapan bahwa solusi optimum lokal tersebut dapat menghasilkan solusi optimum global juga. Pada saat mengimplementasikan greedy, diterapkan prinsip *"take what you can get, now!"*, yang berarti setiap langkah yang dapat dilakukan, lakukan saat itu juga. Dengan adanya prinsip tersebut, menghasilkan greedy yang selalu mengambil solusi optimum local karena sifatnya yang tamak membuat mengambil sebanyak-banyaknya kesempatan yang ada. Oleh karena itu, greedy juga memiliki prinsip lain sebagai batasan dalam pengambilan langkah. Ketika melakukan suatu langkah, tidak dapat kembali ke langkah selanjutnya, maka dari itu penerapan algoritma greedy tidak bisa dilakukan secara sembarangan, diperlukan keputusan yang terbaik untuk menentukan langkah yang akan diambil.

Algoritma Greedy umumnya digunakan untuk memecahkan permasalahan yang membutuhkan optimasi solusi dengan memaksimalkan maupun meminimalkan suatu solusi. Ketika ingin memaksimalkan, dapat diambil nilai maksimum untuk setiap iterasinya. Sedangkan Ketika ingin meminimalkan, dengan algoritma greedy akan memilih yang paling minimum untuk setiap iterasinya.

Algoritma Greedy dalam penerapannya diperlukan pemetaan masalah-masalah yang dipecah menjadi elemen-elemen. Beberapa elemen tersebut, yakni himpunan kandidat, himpunan solusi, fungsi solusi, fungsi seleksi, fungsi kelayakan, dan fungsi objektif. Untuk penjelasa lebih lanjut terkait elemen-elemen algoritma greedy, dijabarkan sebagai berikut.

- Himpunan Kandidat

  Himpunan kandidat atau yang dilambangkan dengan huruf C. merupakan langkah-langkah yang memiliki peluang untuk dipilih dari setiap iterasi
- Himpunan Solusi

  Himpunan solusi dilambangkan dengan huruf S yang berisikan langkah-langkah yang dipilih dari himpunan kandidat.
- Fungsi Solusi

  Fungsi solusi merupakan fungsi yang membantu untuk melakukan penentuan terhadap langkah yang sudah dipilih dan langkah tersebut akan dikategorikan sebagai himpunan solusi yang sudah memberikan hasil sesuai dengan ekspektasi pengguna.
- Fungsi Seleksi

  Fungsi seleksi merupakan fungsi yang berfungsi untuk melakukan seleksi terhadap langkah yang akan dipilih selanjutnya berdasarkan algoritma greedy, yaitu memaksimalkan atau meminimalkan.
- Fungsi Kelayakan

  Fungsi kelayakan merupakan metode untuk memeriksa langkah yang dipilih oleh fungsi seleksi tidak melanggar batasan yang terdapat pada kenyataan dan dapat dimasukkan ke dalam himpunan solusi.
- Fungsi Objektif

  Komponen terakhir dalam algoritma greedy yaitu fungsi objektif. Fungsi objektif merupakan fungsi untuk memaksimalkan atau meminimalkan langkah yang akan diambil terhadap persoalan yang dimiliki.

Berdasarkan komponen tersebut, maka dapat disimpulkan bahwa algoritma Greedy melibatkan pencarian himpunan bagian dari himpunan kandidat, dengan himpunan bagian harus memenuhi syarat seperti himpunan bagian menyatakan suatu solusi dan dioptimasi dengan fungsi objektif. Skema umum algoritma Greedy seperti berikut.

```
function greedy(C : himpunan_kandidat) → himpunan_solusi
{ Mengembalikan solusi dari persoalan optimasi dengan algoritma greedy }
Deklarasi
  x : kandidat
  S : himpunan_solusi

Algoritma:
  S ← {}      { inisialisasi S dengan kosong }
  while (not SOLUSI(S)) and (C ≠ {} ) do
      x ← SELEKSI(C)     { pilih sebuah kandidat dari C}
      C ← C – {x}        { buang x dari C karena sudah dipilih }
      if LAYAK(S ∪ {x}) then      { x memenuhi kelayakan untuk dimasukkan ke dalam himpunan solusi }
          S ← S ∪ {x}     { masukkan x ke dalam himpunan solusi }
      endif
  endwhile
  {SOLUSI(S) or C = {} }

  if SOLUSI(S) then    { solusi sudah lengkap }
    return S
  else
    write('tidak ada solusi')
  endif
```

Gambar 1. Pseudocode Umum Algoritma Greedy
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf

Walaupun algoritma greedy berupaya untuk menemukan solusi optimum global dari setiap iterasinya, tetapi solusi dari algoritma greedy tidak dapat dipastikan solusi yang optimum global. Solusi yang dihasilkan merupakan solusi optimum semu atau pseudo-optimum dikarenakan hal-hal seperti berikut.

- Setiap persoalan yang didapat menggunakan fungsi seleksi yang berbeda-beda sehingga diperlukan untuk mencar fungsi seleksi yang paling tepat untuk menyelesaikan persoalan tersebut menjadi optimal, dan
- *greedy* tidak seakurat brute force yang melakukan iterasi terhadap semua kemungkinan yang terjadi.

Alasan tersebut membuat greedy tidak selalu menghasilkan solusi optimal, tetapi ada beberapa permasalahan yang dapat diselesaikan secara optimal oleh greedy. Berikut adalah permasalahan yang dapat diselesaikan dengan greedy beserta hasil akhir keoptimalan solusinya.

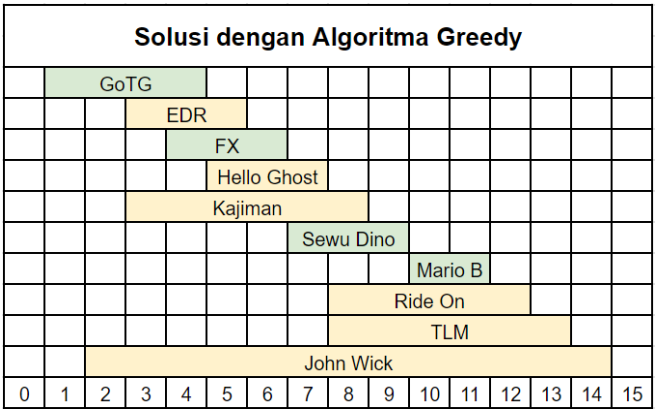
Kegiatan	Hasil Akhir
Coin Exchange Problem	tidak selalu optimal
Knapsack 1/0	tidak selalu optimal
Shortest Path	tidak selalu optimal
Egyptian Fraction	tidak selalu optimal
Travelling Salesman Problem (TSP)	tidak selalu optimal
Job Scheduling with Deadlines	selalu optimal
Penyelesaian kode Huffman	selalu optimal
Minimum Spanning Tree	selalu optimal
Pemilihan kegiatan	selalu optimal

Berdasarkan table tersebut, algoritma greedy dapat menghasilkan solusi yang selalu optimal pada beberapa permasalahan. Meskipun algoritma Greedy tidak menghasilkan solusi terbaik, tetapi solusi algoritma Greedy yang dihasilkan dapat dianggap sebagai solusi hampiran (*approximation*).

B. Activity Selection Problem

Dalam memilih aktivitas seringkali kita mengalami kebingungan untuk mendahulukan aktivitas yang ingin dilakukan terlebih dahulu. Waktu yang bersamaan dan terbatas mengharuskan kita untuk dapat melakukan aktivitas sebanyak mungkin. Setiap aktivitas yang ingin dilakukan memiliki waktu mulai dan waktu selesai. Aktivitas dapat dikatakan compatible jika interval durasi aktivitas yang satu dengan yang sebelumnya tidak beririsan satu sama lain. Di bawah ini merupakan contoh dari *activity selection problem* dalam penentuan menonton film di bioskop:

Aktivitas	Waktu Mulai	Waktu Selesai
Guardian of The Galaxy	1	4
Evil Dead Rise	3	5
Fast X	4	6
Hello Ghost	5	7
Kajiman	3	8
Sewu Dino	7	9
Mario Bros	10	11
Ride On	8	12
The Little Mermaid	8	13
John Wick: Chapter 4	2	14



Gambar 2. Solusi Activity Selection Problem dengan Algoritma Greedy
Sumber: dokumen pribadi penulis

Keterangan:

Warna kuning: durasi aktivitas tersebut berlangsung

Warna hijau: aktivitas yang dapat dilakukan dengan algoritma greedy

Berdasarkan contoh diatas, untuk memaksimalkan film yang di bioskop dapat memilih film Guardiang of The Galaxy, Fast X, Sewu Dino, dan Mario Bros. Persoalan tersebut dapat diselesaikan dengan algoritma greedy dan exhaustive search. Pada penyelesaian dengan algoritma greedy, dapat melakukan strategi seperti berikut.

1. Mengurutkan semua aktivitas berdasarkan waktu selesainya dan terurut dari waktu yang kecil ke waktu yang besar,
2. Setiap langkah yang akan diambil, pilih waktu yang mulainya lebih besar atau sama dengan waktu selesai aktivitas yang sudah dipilih sebelumnya agar tidak terdapat waktu yang terbuang untuk menunggu aktivitas selanjutnya, dan
3. Memilih aktivitas yang memiliki durasi paling kecil dan waktu mulai nya tidak lebih besar dari waktu selesai aktivitas lain yang sebelumnya sudah dipilih.

Ketika membandingkan algoritma greedy dengan exhaustive search, hasil dari algoritma greedy jauh lebih cepat. Exhaustive search memiliki kompleksitas $O(n\ 2^n)$, sedangkan menggunakan algoritma greedy kompleksitas yang diperoleh jauh lebih kecil, yaitu sebesar $O(n)$.

Di bawah ini merupakan pseudocode dari penyelesaian *activity selection problem* dengan algoritma greedy.

```
function Greedy-Activity-Selector(s1, s2, ..., sn : integer, f1, f2, ..., fn : integer) → set of integer
{ Asumsi: aktivitas sudah diurut terlebih dahulu berdasarkan waktu selesai:   f1 <= f2 <= ... <= fn }
Deklarasi
   i, j, n : integer
   A : set of integer
Algoritma:
   n ← length(s)
   A ← {1}     { aktivitas nomor 1 selalu terpilih }
   j ← 1
   for i ← 2 to n do
      if si ≥ fj then
         A ← A ∪ {i}
         j ← i
      endif
   endif
```

Gambar 3. Pseudocode Activity Selection Problem Algoritma Greedy
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf

III. PEMBAHASAN

Dalam hal melakukan penjadwalan rencana perjalanan memerlukan durasi perjalanan akan dilakukan. Untuk makalah ini akan difokuskan untuk melakukan perjalanan dengan tujuan berlibur. Selain itu, fokus dalam makalah ini adalah dengan berkunjung ke tempat-tempat wisata sebanyak mungkin dalam hari itu. Dengan tidak memerhatikan durasi berpindah dari satu tempat ke tempat lain, penjadwalan akan ditekankan untuk durasi berkunjung selama di tempat wisata tersebut.

Pada bab sebelumnya, telah dibahas mengenai penyelesaian activity selection problem. Pembahasan tersebut bertujuan sebagai gambaran dalam memasuki bab pembahasan ini dikarenakan membuat jadwal rencana perjalanan dapat dikatakan mirip dengan activity selection problem. Perbedaan yang terjadi dalam hal dibutuhkannya informasi mengenai jam buka dan jam tutup tempat wisata tersebut. Selain itu, dibutuhkan juga perkiraan durasi berkunjung oleh wisatawan dengan memerhatikan *peak hour*. Dengan memerhatikan peak hour diharapkan pengguna dapat menikmati liburannya dengan maksimal.

A. Penerapan dalam Algoritma Greedy

Dikarenakan memiliki perbedaan informasi yang dibutuhkan, maka dalam algoritma greedy dilakukan penambahan kolom. Untuk melakukan penjadwalan rencana perjalanan, terdapat beberapa langkah. Dalam kasus ini, akan membahas rencana perjalanan ke Malaysia dalam waktu satu hari. Berikut adalah list tempat beserta informasi mengenai waktu operasional tempat wisata tersebut dan perkiraan rata-rata durasi dalam berkunjung ke tempat wisata tersebut.

Nama Tempat	Jam Buka	Jam Tutup	Durasi Kunjungan (jam)
Petronas Tower	10	18	2
Genting Highland Theme Park	2	21	5
Batu Caves	7	21	3
Gunung Mulu National Park	8	17	4
Langkawi Sky Bridge	10	19	5
Legoland Malaysia	10	18	8
Kuala Lumpur Bird Park	9	20	6
Bukti Bintang	12	22	3
Kuala Lumpur Tower	10	22	3
Sunway Lagoon	11	20	7
Kuala Lumpur City Center	10	20	2
Kuala Lumpur Butterfly Park	8	17	3
Kuala Lumpur Deer Park	8	17	3
Kuala Lumpur Orchid Garden	8	17	3
Kuala Lumpur Lake Garden	8	17	3

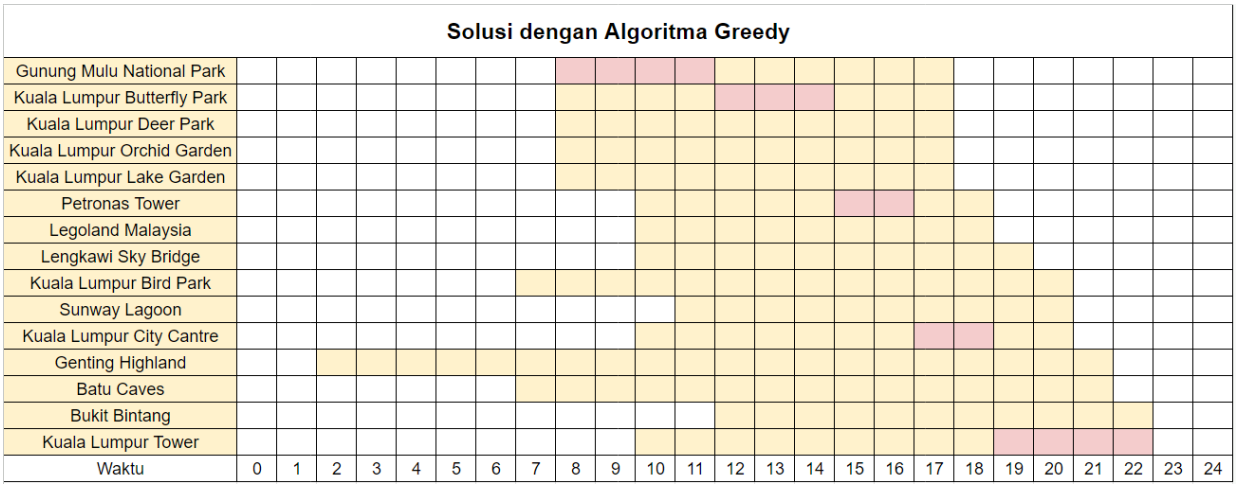
Berdasarkan table yang berisikan informasi dari tempat wisata diatas, dapat ditemukan solusi untuk memaksimalkan kunjungan di Kuala Lumpur dalam satu hari. Berikut adalah solusi dari permasalahan activity selection yang diimplementasikan dalam bahasa pemrograman Python.

```
                  Nama Tempat  Jam Buka  Jam Tutup  Durasi
0     Gunung mulu national park         8         17       4
1   Kuala Lumpur Butterfly Park         8         17       3
2        Kuala Lumpur Deer Park         8         17       3
3    Kuala Lumpur Orchid Garden         8         17       3
4     Kuala Lumpur Lake Gardens         8         17       3
5                      Petronas        10         18       2
6             Legoland Malaysia        10         18       8
7           Lengkawi Sky Bridge        10         19       5
8         Kuala Lumpur Bird Park         9         20       6
9                  Sunway Lagoon        11         20       7
10     Kuala Lumpur City Centre        10         20       2
11              Genting Highland         2         21       5
12                    Batu caves         7         21       3
13                Bukit Bintang        12         22       4
14           Kuala Lumpur Tower        10         22       3

Rencana Perjalanan hasil Greedy Algorithm:
Pada jam 8 sampai 12 berkunjung ke Gunung mulu national park
Pada jam 12 sampai 15 berkunjung ke Kuala Lumpur Butterfly Park
Pada jam 15 sampai 17 berkunjung ke Petronas
Pada jam 17 sampai 19 berkunjung ke Kuala Lumpur City Centre
Pada jam 19 sampai 22 berkunjung ke Kuala Lumpur Tower
```

Gambar 4. Hasil dari Eksperimen Penjadwalan Rencana Perjalanan
Sumber: dokumen pribadi penulis

Dapat dilihat berdasarkan hasil diatas bahwa pengguna dapat berkunjung ke lima tempat dalam satu hari dengan tempat kunjungan Gunung Mulu National Park, Kuala Lumpur Butterfly Park, Petronas Tower, Kuala Lumpur City Centre, Kuala Lumpur Tower dengan total durasi sebanyak 14 jam tanpa melakukan perhitungan dalam perjalanan yang ditempuh dari satu tempat ke tempat lainnya.



Gambar 5. Hasil dari Eksperimen Penjadwalan Rencana Perjalanan
Sumber: dokumen pribadi penulis

Keterangan:

Warna kuning: durasi tempat wisata beroperasi
Warna merah: waktu kunjungan ke tempat wisata

B. Pseudocode Algoritma Greedy

Untuk menemukan solusi dari penjadwalan rencana perjalanan, dapat dilakukan dengan pseudocode terbaru seperti dibawah ini.

```
function tripPlanningGreedy( list of tuples: activities) → list of tuples

declaration
    currentTime, startTime, endTime: integer
    plan: list of tuples
    activity: list

algorithm
activities.sort(x[2])
for activity in activities do
    startTime ← max(activity[1], currentTime)
    endTime ← startTime + activity[3]

    if endTime <= activity 2 then
        plan.append(startTime, endTime, activity[0])
        currentTime ← endTime
    endif

    if currentTime >= 24 then
        break
    endif
```

Seperti yang tertera dalam pseudocode diatas, dapat dilihat bahwa kompleksitas algoritma untuk penyelesaian masalah ini semakin memburuk dibandingkan kompleksitas permasalahan activity selection. Berikut adalah komponen yang mempengaruhi kompleksitas algoritma semakin memburuk.

1. Melakukan sorting terlebih dahulu. Tempat wisata diurutkan berdasarkan waktu tutupnya dengan kompleksitas O(n log n), dimana n adalah jumlah kegiatan
2. Terdapat pengulangan untuk melakukan iterasi disetiap bagian list. Algoritma menjadi berulang sebanyak n kali, dimana n merupakan jumlah kegiatan. Pengulangan ini memiliki kompleksitas O(1).

Perubahan algoritma yang terjadi dengan penambahan informasi durasi, terdapat perbedaan pemetaan algoritma greedy. Dibawah ini adalah pemetaan persoalan penjadwalan rencana perjalanan dengan algoritma greedy.

1. Himpunan Kandidat (C)
   Kegiatan atau tempat wisata yang akan dikunjungi pada hari tersebut
2. Himpunan Solusi (S)
   Kegiatan atau tempat wisata yang terpilih untuk dikunjungi pada hari tersebut
3. Fungsi Solusi
   Melakukan pengecekan apakah kegiatan atau tempat wisata yang akan dikunjungi sudah maksmal
4. Fungsi seleksi
   Melakukan pengurutan berdasarkan waktu tutup tempat wisata dan memilih tempat wisata yang membutuhkan durasi kunjungan paling kecil
5. Fungsi Kelayakan
   Melakukan pengecekan apakah saat melakukan kunjungan tempat wisata tersebut masih beroperasi

dan melakukan pengecekan ketika penambahan kegiatan sudah melebihi 24 jam atau tidak

6. Fungsi Objektif
   Memaksimalkan jumlah kegiatan atau tempat wisata yang akan dikunjungi pada hari tersebut.

IV. PENUTUP

A. *Kesimpulan*

Algoritma Greedy merupakan algoritma yang sering digunakan untuk memaksimalkan atau meminimalkan suatu masalah. Meskipun solusi yang diambil merupakana solusi global, tetapi algoritma greedy belum tentu menghasilkan solusi yang optimum. Salah satu contoh permasalahan sehari-hari yang dapat diselesaikan dengan algortima greedy adalah membuat jadwal rencana perjalanan. Permasalahan tersebut dapat dikatakan mirip dengan activity selection problem sehingga membuat jadwal rencana perjalanan dengan algoritma greedy dapat menghasilkan hasil yang optimum.

Dengan menambah beberapa informasi seperti waktu operasional tempat wisata dan perkiraan durasi berkunjung, algoritma greedy dapat menyelesaikan permasalahan tersebut. Kompleksitas yang dihasilkan dari algoritma greedy pun sudah cukup baik, yakni O(n log n). Berbeda dengan exhaustive search yang memiliki kompleksitas $O(n\ 2^n)$.

B. *Saran*

Untuk berpindah dari satu tempat ke tempat wisata yang lain membutuhkan waktu perpindahan. Dalam makalah ini, belum mengaplikasikan perhitungan perpindahan tersebut karena memiliki jarak dan waktu yang relative untuk melakukan perpindahan. Oleh karena itu, dapat dikembangkan algoritma ini dengan menambah informasi mengenai jarak dan waktu perpindahan dengan mengkombinasikan algoritma *shorthest path* dan algoritma *greedy*.

V. UCAPAN TERIMA KASIH

Puji dan syukur dipanjatkan kepada Tuhan Yang Maha Esa karena atas berkat dan rahmat-Nya dapat menyelesaikan tugas makalah IF2211 Strategi Algoritma. Terima kasih kepada dosen pengampu mata kuliah ini, Bapak Ir. Rila Mandala, M.Eng., Ph.D yang telah memberikan penulis ilmu serta bimbingan selama belajar mengajar. Penulis juga mengucapkan terima kasih kepada orang tua dan teman-teman penulis yang selama ini sudah memberikan dukungan serta bantuan dalam proses penyusunan makalah ini.

VIDEO LINK AT YOUTUBE

https://youtu.be/YV1uAIHK9rE

REFERENCES

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf, Diakses pada 20 Mei 2023

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag2.pdf, Diakses pada 20 Mei 2023

[3] https://repository.unikom.ac.id/37395/1/5-Algoritma%20Greedy.pdf, Diakses pada 20 Mei 2023

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, 22 Mei 2023

Eunice Sarah Siregar
13521013