Optimalisasi Implementasi Algoritma Greedy dalam Fungsi Penukaran Mata Uang Rupiah

Hery Sunandar[1], Pristiwanto[2]
[1,2]STMIK Budi Darma Medan, JL. Sisingamangaraja Np. 338 Simpang Limun Medan
Email : herysunandar@gmail.com[1], 4nt0.82@gmail.com[2]

Abstrak

Perkembangan teknologi saat ini semakin mempermudah setiap penggunanya dalam mengakses maupun mempercepat pekerjaan dengan sistem Aplikasi yang diciptakan. Aplikasi inilah memberikan kemudahan kepada pengguna dalam mengerjakan pekerjaan dengan cepat dan mudah. Terutama sistem untuk penukaran uang menjadi pecahan melalui proses optimasi. Penelitian ini dibuat suatu sistem Penerapan Algoritma Greedy untuk penukaran uang Rupiah. Aplikasi yang akan dibuat di skripsi ini menawarkan kemudahan dalam penukaran uang Rupiah menjadi pecahan. Khususnya dapat mempermudah pekerjaan dalam dunia perbankan untuk menukarkan uang nasabahnya dengan mengambil hasil yang paling optimal. Oleh karena itu maka diciptakanlah Aplikasi Penukaran uang Rupiah ini agar proses penukaran uang yang tadinya membutuhkan waktu lama dan sulit untuk menukarkan uang rupiah.

Kata kunci : *Penukaran Uang, Algoritma Greedy, Optimasi, Solusi Greedy*

Abstract

Current technological developments make it easier for each user to access and accelerate work with the application system that was created. This application makes it easy for users to do work quickly and easily. Especially the system for exchanging money into fractions through the optimization process. This research created a Greedy Algorithm Application System for Rupiah exchange. The application that will be made in this thesis offers the convenience of converting Rupiah into fractions. In particular, it can simplify work in the banking world to exchange customers' money by taking the most optimal results. Therefore, this Rupiah Currency Exchange Application was created so that the money exchange process that used to take a long time and was difficult to exchange rupiah currency.

Keywords: *Money Exchange, Greedy Algorithm, Optimization, Greedy Solution*

1. PENDAHULUAN

Dalam kehidupan sehari-hari, banyak terdapat persoalan yang menuntut pencarian solusi optimum. Persoalan tersebut dinamakan persoalan optimasi *(optimization problems)*. Persoalan optimasi adalah persoalan yang tidak hanya mencari sekedar solusi, tetapi mencari solusi terbaik. Solusi terbaik adalah solusi yang memiliki nilai minimum atau maksimum dari sekumpulan alternatif solusi yang mungkin.

Algoritma greedy adalah salah satu algoritma yang dapat digunakan untuk mendapatkan solusi terbaik dan merupakan algoritma yang paling populer saat ini. Aplikasi algoritma greedy dapat digunakan dalam masalah seperti: mencari jalur terpendek, strategi permainan monopoli, masalah penukaran uang dan lain-lain. Algoritma greedy adalah algoritma yang memecahkan masalah langkah per langkah, pada setiap langkah yaitu mengambil pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi ke depan juga berharap bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global[1].

Penelitian ini membahas masalah penukaran uang, yaitu mencari jumlah minimum uang pecahan yang dihasilkan dari uang yang akan ditukarkan[2]. Untuk perhitungan manual maka akan sangat sulit mencari jumlah minimum uang pecahan tersebut, oleh karena itu akan dibuat suatu aplikasi yang dapat mencari solusi dari masalah tersebut, yaitu dengan menggunakan algoritma Greedy.

Aplikasi Penerapan Algortima Greedy untuk penukaran uang Rupiah ini digunakan oleh bagian yang mengelola data pernukaran uang, salah satunya yaitu bank atau perbankan, seperti data nasabah ingin menukarkan uang ke bentuk pecahan[3].

2. LANDASAN TEORI

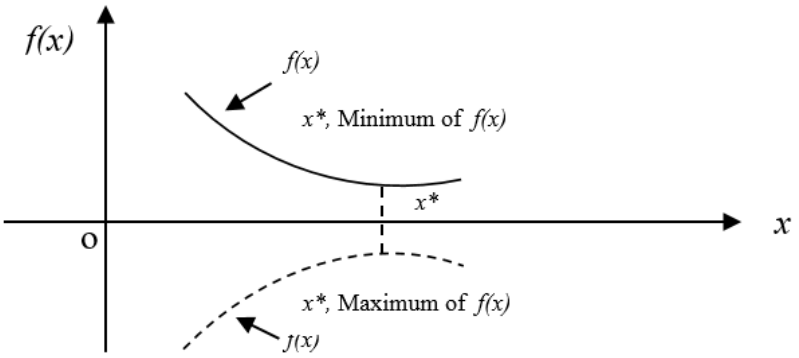
2.1 Definisi Optimasi

Kata optimasi sifatnya termasuk global, karena banyak digunakan sebagai kata kunci paling popular. Optimasi secara umum adalah untuk memaksimalkan atau mengoptimalkan sesuatu hal yang bertujuan untuk mengelola sesuatu yang dikerjakan. Sehingga, optimasi bisa dikatakan kata benda yang berasal dari kata kerja, dan optimasi bisa dianggap baik sebagai ilmu pengetahuan dan seni menurut tujuan yang ingin dimaksimalkan[4].

Banyak Fakor yang berkaitan dengan optimasi, seperti optimasi computer, optimasi Web, optimasi booting, optimasi CPU (Processor), optimasi bateray (pada Laptop), Optimasi Suara, Optimasi tampilan, dan lain sebagainya, sehingga optimasi memang diperlukan untuk hal apa pun untuk membuat sesuatu sebagus mungkin atau paling maksimal. Persoalan optimasi adalah persoalan yang sangat penting untuk diterapkan untuk segala sistem maupun organisasi. Dengan optimasi pada sebuah sistem kita akan bisa berhemat dalam segala hal antara lain energi, keuangan, sumber daya alam, kerja dan lain-lain, tanpa mengurangi fungsi sistem tersebut[5].

Secara matematis optimasi adalah cara mendapatkan harga ekstrim baik maksimum atau minimum dari suatu fungsi tertentu dengan faktor-faktor pembatasnya. Jika persoalan yang akan diselesaikan dicari nilai maksimumnya, maka keputusannya berupa maksimasi. Optimasi dalam penyelesaian masalah merupakan suatu cara pengambilan keputusan sehingga didapatkan hasil penyelesaian yang optimal sesuai dengan kendala " *state of nature*" yang harus dipenuhi. Metode yang banyak digunakan antara lain *Calculus*, *Dinamic Programming*, *Linear Programming*, *Geomatry* dan *Inventory Theory*.

Optimasi juga dapat didefinisikan sebagai proses untuk mendapatkan keadaan yang memberikan nilai maksimum atau minimum dari suatu fungsi. Hal ini dapat dilihat dari gambar II.1, bahwa jika titik x* berkaitan dengan nilai minimum fungsi f(x), titik yang sama juga berkaitan dengan nilai maksimum dari negatif fungsi tersebut –f(x). Tanpa menghilangkan keumumannya, optimasi dapat diartikan meminimalkan, karena maksimum suatu fungsi dapat diperoleh melalui minimum dari negatif fungsi yang sama.



Gambar 1. Minimum dari f(x) sama dengan Maksimum dari –f(x)[4]

Jadi, Optimasi adalah Suatu proses memaksimalkan dan meminimumkan suatu fungsi yang ada dengan maksud untuk memperoleh hasil yang terbaik atau hasil maksimal dari fungsi tersebut dengan keputusan dalam beberapa tahap. Tujuan akhir dari semua keputusan seperti itu adalah meminimalkan upaya yang diperlukan atau untuk memaksimalkan manfaat yang diinginkan.

2.2. Metode Optimasi

Metode mencari optimum dikenal sebagai teknik *mathematical programming* dan biasa dipelajari sebagai bagian riset operasi. Riset operasi adalah cabang matematika yang berkaitan dengan penerapan metode ilmiah dan teknik pengambilan keputusan dan penetapan penyelesaian terbaik atau optimal. Pada awal dari subyek riset operasi dapat ditelusuri pada periode awal Perang Dunia II, selama perang, militer inggris menghadapi masalah mengalokasikan sumber daya yang sangat langka dan terbatas seperti : pesawat tempur, radar, dan kapal selam, untuk beberapa kegiatan penyebaran ke berbagai target dan tujuan[6][1].

Karena tidak ada metode sistematis yang tersedia untuk memecahkan masalah alokasi sumber daya militer diatas, tim matematikawan mengembangkan metode untuk memecahkan masalah secara ilmiah. Metode yang dikembangkan oleh tim berperan penting dalam memenangkan pertempuran udara oleh inggris. Metode tersebut seperti program linier, yang dikembangkan sebagai hasil riset pada militer.

Perkembangan metode optimasi semakin mengalami kemajuan sampai ke masa modern, hal ini dapat dilihat dengan semakin banyak metode optimasi yang ditemukan dan dapat menghasilkan solusi yang semakin optimal. Metode optimasi yang popular dan banyak dipakai antara lain seperti *Dynamic Programming, Integer Programming, Game Theory*, dan metode optimasi modern.

Metode optimasi modern juga disebut metode optimasi non-tradisional, muncul sebagai metode yang ampuh dan popular untuk menyelesaikan masalah teknik optimasi yang kompleks. Metode yang termasuk seperti algoritma genetik, optimasi partikel swarm, optimasi koloni semut, optimasi berbasis jaringan syaraf tiruan, optimasi fuzzy, dan simulated annealing.

2.3. Pengertian Algoritma

Ditinjau dari asal usul katanya, Algoritma sendiri mempunyai sejarah yang aneh. Orang hanya menemukan kata *Algorism* yang berarti proses menghitung dengan angka arab. Anda akan dikatakan *Algorist* jika anda menghitung menggunakan Angka Arab [7].

Para ahli bahasa berusaha menemukan asal kata ini namun hasilnya kurang memuaskan. Akhirnya para ahli sejarah matematika menemukan asal kata tersebut yang berasal dari nama penulis buku arab yang terkenal yaitu Abu Ja'far Muhammad Ibnu Musa Al-Khuwarizmi. Al-Khuwarizmi dibaca orang barat menjadi Algorism[8].

Al-Khuwarizmi menulis buku yang berjudul Kitab Al Jabar Wal-Muqabala yang artinya "Buku pemugaran dan pengurangan" (*The book of restoration and reduction*). Dari judul buku itu kita juga memperoleh akar kata "Aljabar" (*Algebra*). Perubahan kata dari *Algorism* menjadi *Algorithm* muncul karena kata *Algorism* sering dikelirukan dengan *Arithmetic*.

Karena perhitungan dengan angka Arab sudah menjadi hal yang biasa, Maka lambat laun kata *Algorithm* berangsur-angsur dipakai sebagai metode perhitungan (komputasi) secara umum, sehingga kehilangan makna kata aslinya. Dalam Bahasa Indonesia, kata *Algorithm* diserap menjadi Algoritma.

2.4. Definisi Algoritma Greedy

Algoritma greedy merupakan salah satu dari sekian banyak algoritma yang sering di pakai dalam implementasi sebuah system atau program yang menyangkut mengenai pencarian"optimasi". Dalam kehidupan sehari hari, banyak terdapat persoalan yang menuntut pencarian solusi optimum. Persoalan tersebut dinamakan persoalan optimasi (optimization Problems). Persoalan Optimasi adalah persoalan yang tidak hanya mencari sekedar solusi, tetapi mencari solusi terbaik[3].

Algoritma Greedy ini hampir sama dengan metode *exhaustive search* dan *brute force,* dimana *Exhaustive search* ialah teknik pencarian solusi secara brute force pada masalah yang melibatkan pencarian elemen dengan sifat khusus, biasanya di antara objek-objek kombinatorik seperti permutasi, kombinasi, atau himpunan bagian dari sebuah himpunan. Berdasarkan definisi ini, maka exhaustive search adalah *brute force* juga. Oleh karena itu *exhaustive search* adalah salah satu implementasi dalam brute force dalam kasus pencarian[7].

Secara Harfiah Greedy artinya rakus atau tamak, sifat yang berkonotasi negatif. Orang yang memiliki sifat ini akan mengambil sebanyak mungkin atau mengambil ang paling bagus atau yang paling mahal. Sesuai dengan arti tersebut, "Prinsip Greedy adalah *take what you can get now*".

Oleh karena itu, Algoritma Greedy merupakan algoritma yang lazim untuk memecahkan persoalan optimasi meskipun hasilnya tidak selalu merupakan solusi yang optimum. Prinsip utama dari algoritma ini adalah mengambil sebanyak mungkin apa yang dapat diperoleh sekarang.

Untuk memecahkan persoalan dengan algoritma Greedy, memerlukan elemen-elemen berikut:

a. Himpunan kandidat
   Berisi elemen-elemen pembentuk solusi.
b. Himpunan solusi
   Berisi kandidat-kandidat yang terpilih sebagai solusi persoalan.
c. Fungsi seleksi (*selection function*)
   Memilih kandidat yang paling memungkinkan mencapai solusi optimal. Kandidat yang sudah dipilih pada suatu langkah tidak pernah dipertimbangkan lagi pada langkah selanjutnya.
d. Fungsi kelayakan (*feasible*)
   Memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak, yakni kandidat tersebut bersama-sama dengan himpunan solusi yang sudah terbentuk tidak melanggar kendala (*constraints*) yang ada. Kandidat yang layak dimasukkan ke dalam himpunan solusi, sedangkan kandidat yang tidak layak dibuang dan tidak pernah dipertimbangkan lagi.
e. Fungsi obyektif, yaitu fungsi yang memaksimumkan atau meminimumkan nilai solusi (misalnya panjang lintasan, keuntungan, dan lain-lain).

Optimum global dengan menggunakan algoritma Greedy belum tentu merupakan solusi optimum (terbaik), tetapi sub-optimum atau pseudo-optimum. Alasannya adalah sebagai berikut:

1. Algoritma Greedy tidak beroperasi secara menyeluruh terhadap semua alternatif solusi yang ada (sebagaimana pada metode *Exhaustive Search*).
2. Terdapat beberapa fungsi seleksi yang berbeda, sehingga harus dipilih fungsi yang tepat agar algoritma yang digunakan menghasilkan solusi optimal.

3. METODE PENELITIAN

Metode yang digunakan dalam penelitian ini meliputi metode pengumpulan data dan metode pengembangan perangkat lunak[9] adalah:

1. Metode Pengumpulan Data
   a. Metode Studi Literatur
      Dengan mengumpulkan data dan mempelajari literatur yang berkaitan dengan optimasi, dan Algoritma greedy.
   b. Metode Wawancara
      Dengan melakukan wawancara dengan suatu bank atau instansi yang dijadikan objek penelitian yaitu untuk mendapatkan data-data atau info-info yang diperlukan untuk penelitian dan pembangunan perangkat lunak.
2. Metode Pembangunan Perangkat Lunak
   a. Metode Pendekatan Perangkat Lunak
      Metode yang digunakan yaitu dengan menggunakan metode pendekatan terstruktur, yakni analisis yang terfokus pada aliran data. Pendekatan terstruktur mengenalkan beberapa alat untuk mengembangkan sistem terstruktur. Alat-alat tersebut yaitu antara lain : entity relationship diagram (ERD) , data flow diagram (DFD), Flowchart diagram.
   b. Metode Analisis dan Perancangan
      Analisis dan perancangan perangkat lunak dilakukan untuk menentukan permasalahan mengenai bahasa pemrograman yang akan digunakan, input dan output program dan permasalahan teknik algoritma yang akan diimplementasikan

4. HASIL DAN PEMBAHASAN

Pada bab ini membahas tentang analisa dan perancangan sistem tentang pembuatan Penerapan Algortima Greedy Untuk Penukaran Uang Rupiah. Analisis dan perancangan ini meliputi analisis sistem, perancangan proses, tahap pembuatan sistem, dan perancangan antarmuka.

Ada beberapa hal yang harus di analisis terlebih dahulu sebelum membuat perancangan sistem, antara lain yaitu ruang lingkup atau batasan sistem, apa yang ingin dihasilkan oleh sistem yang di bangun (tujuan dari sistem atau output), dan siapa saja yang terlibat dai dalam sistem. Ruang lingkup sistem akan dibahas adalah mengenai Penerapan Algoritma Greedy Untuk Penukaran Uang Rupiah.

Contoh: dimana jika ada seorang ingin menukarkan 43 rupiah dengan pecahan yang tersedia 1,3,5,6,7, berapakan jumlah koin yang paling optimum ? maka akan menghabiskan waktu yang sangat lama untuk mencari jumlah pecahan yang paling minimum. Oleh karena itu aplikasi ini akan memberi kemudahan kepada pengguna dalam masalah penukaran uang, sehingga dapat mempercepat proses penukaran serta dapat mempercepat waktu penukaran.

4.1. Analisis Algoritma Greedy

Algoritma Greedy merupakan algoritma yang lazim untuk memecahkan persoalan optimasi meskipun hasilnya tidak selalu merupakan solusi yang optimum. Sesuai arti harafiah, Greedy berarti tamak. Prinsip utama dari algoritma ini adalah mengambil sebanyak mungkin apa yang dapat diperoleh sekarang. Untuk memecahkan persoalan dengan algoritma Greedy, kita memerlukan elemen-elemen sebagai berikut.

a. Himpunan kandidat
   Berisi elemen-elemen pembentuk solusi.
b. Himpunan solusi
   Berisi kandidat-kandidat yang terpilih sebagai solusi persoalan.

c. Fungsi seleksi (*selection function*)
   Memilih kandidat yang paling memungkinkan mencapai solusi optimal. Kandidat yang sudah dipilih pada suatu langkah tidak pernah dipertimbangkan lagi pada langkah selanjutnya.
d. Fungsi kelayakan (*feasible*)
   Memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak, yakni kandidat tersebut bersama-sama dengan himpunan solusi yang sudah terbentuk tidak melanggar kendala (*constraints*) yang ada. Kandidat yang layak dimasukkan ke dalam himpunan solusi, sedangkan kandidat yang tidak layak dibuang dan tidak pernah dipertimbangkan lagi.
e. Fungsi obyektif, yaitu fungsi yang memaksimumkan atau meminimumkan nilai solusi (misalnya panjang lintasan, keuntungan, dan lain-lain).

Skema umum algoritma Greedy adalah sebagai berikut.
a. Inisialisasi S dengan kosong.
b. Pilih sebuah kandidat C dengan fungsi seleksi.
c. Kurangi C dengan kandidat yang sudah dipilih dari langkah (b) di atas.
d. Periksa apakah kandidat yang dipilih tersebut bersama-sama dengan himpunan solusi membentuk solusi yang layak atau feasible (dengan fungsi kelayakan).
e. Periksa apakah himpunan solusi sudah memberikan solusi yang lengkap serta optimal (dengan fungsi objektif).

4.2. Analisis Penyelesaian Kasus Dengan Algoritma Greedy

Berdasarkan judul yang penulis buat yaitu mengenai penukaran koin, maka di bawah ini akan dibuat contoh beserta penyelesaiannya menggunakkan algoritma greedy.
Contoh Persoalan :
Dimana seseorang ingin menukarkan uangnya, tinjau masalah menukarkan uang 32 dengan koin pecahan yang tersedia adalah 1, 5, 10, dan 25:
Pertanyaan :
Berapakah jumlah minimum pecahan dari uang yang ditukar yaitu 32
Solusi :
Langkah 1 : Pilih 1 buah koin 25 (Total = 25)
Langkah 2 : Pilih 1 buah koin 5 (Total = 25 + 5 = 30)
Langkah 3 : Pilih 2 buah koin 1 (Total = 25+5+1+1= 32)

Maka dengan menggunakkan algoritma greedy akan ditemukan solusi minuman dari koin pecahan yaitu Jumlah koin minimum = 4 (solusi optimal!) dimana untuk mendapatkan jumlah yang cocok dari uang 32 adalah = 25, 5, 1, 1, Sedangkan jumlah maksimum dari uang 32 adalah = koin pecahan 1 sebanyak 32.

Untuk ilustrasi contoh dari algoritma greedy dimana mengenai penukaran uang yaitu mencari jumlah minimum adalah sebagai berikut :

a. Koin: 5, 4, 3, dan 1
   Uang yang ditukar : 7
   Solusi greedy : 7 = 5 + 1 + 1 (3 koin) → Tidak optimal
   Solusi optimal : 7 = 4 + 3 (2 koin)
b. Koin: 10, 7 dan 1
   Uang yang ditukar : 15
   Solusi greedy : 15 = 10 + 1 + 1 + 1 + 1 + 1 (6 koin)
   Solusi optimal : 15 = 7 + 7 + 1 (hanya 3 koin)
c. Koin: 15, 10, dan 1
   Uang yang ditukar : 20

Solusi greedy : 20 =15 + 1 + 1 + 1 + 1 + 1 (6 koin)
Solusi optimal : 20 = 10 + 10 (2 koin)

Pseudo-code algoritma greedy adalah sebagai berikut:

```
function greedy (input C: himpunan_kandidat)® himpunan_kandidat
{ Mengembalikan solusi dari persoalan optimasi dengan algoritma greedy
  Masukan: himpunan kandidat C
  Keluaran: himpunan solusi yang bertipe himpunan_kandidat
}
Deklarasi
   x : kandidat
   S : himpunan_kandidat

Algoritma:
     S ¬ {}   { inisialisasi S dengan kosong }
     while (not SOLUSI(S)) and (C ¹ {} ) do
     x ¬ SELEKSI(C)              { pilih sebuah kandidat dari C}
     C ¬ C - {x}          { elemen himpunan kandidat berkurang satu }
         if LAYAK(S È {x}) then
            S ¬ S È {x}
        endif
    endwhile
{SOLUSI(S) or C = {} }

if SOLUSI(S) then
  return S
else
  write('tidak ada solusi')
endif
```

4.3. Halaman Solusi Greedy

Halaman ini akan tampil jika user mengclick tombol greedy proses yang ada pada halaman sebelumnya, pada halaman ini akan ditampilkan solusi optimal pecahan dari nilai yang mau ditukar, dimana dari contoh sampel diatas nilai pecahan yang ditentukan adalah pecahan 1, pecahan 2, pecahan 3, dan pecahan 13 dan nilai yang mau ditukar adalah 87, maka hasil dari penentuan pecahan dan nilai adalah :

Solusi 1 : Jumlah Diterima 9, Dengan Rincian Pecahan 13 Sejumlah 6, Pecahan 3 Sejumlah 3
Solusi 2 : Jumlah Diterima 29, Dengan Rincian Pecahan 3 Sejumlah 29,
Solusi 3 : Jumlah Diterima 44, Dengan Rincian Pecahan 2 Sejumlah 43, Pecahan 1 Sejumlah 1
Solusi 4 : Jumlah Diterima 87, Dengan Rincian Pecahan 1 Sejumlah 87

Halaman solusi Greedy tersebut dapat dilihat pada gambar 2. Berikut:

Gambar 2. Halaman Solusi Greedy

5. KESIMPULAN

Kesimpulan dan saran dari hasil Penerapan Algortima Greedy Untuk Penukaran Uang Rupiah akan dibahas pada bab ini. Berdasarkan penelitian yang dilakukan oleh peneliti mengenai perancangan penerapan algortima Greedy untuk penukaran uang rupiah, maka dapat ditarik beberapa kesimpulan sebagai berikut :

1. Perancangan Aplikasi Penerapan Algortima Greedy untuk penukaran uang Rupiah dapat mempermudah melakukan proses pencarian solusi optimal untuk menentukan pecahan yang paling kecil.
2. Penerapan Algortima Greedy Untuk Penukaran Uang Rupiah ini dapat dipakai di bank-bank dimana fungsinya untuk masalah penukaran uang agar petugas tidak membutuhkan waktu yang lama dalam proses menentukan jumlah pecahan nominal yang diterima nasabah.
3. Dapat dipakai oleh masyarakat luas karena dibuat berbasiskan webbase, dan pecahan tidak berbentuk statis tetapi dinamis, dimana dapat menentukan nilai pecahan yang tersedia.

DAFTAR PUSTAKA

[1] Paryati, "OPTIMASI STRATEGI ALGORITMA GREEDY UNTUK MENYELESAIKAN PERMASALAHAN KNAPSACK 0-1," *Semin. Nas. Inform.*, vol. 2009, no. semnasIF, pp. 101–110, 2009.

[2] I. Mukhlis, "Analisis Volatilitas Nilai Tukar Mata Uang Rupiah Terhadap Dolar," *J. Indones. Appl. Econ.*, vol. 005, no. 02, pp. 172–182, 2011.

[3] G. I. Sampurno, E. Sugiharti, and A. Alamsyah, "Comparison of Dynamic Programming Algorithm and Greedy Algorithm on Integer Knapsack Problem in Freight Transportation," *Sci. J. Informatics*, vol. 5, no. 1, p. 49, May 2018.

[4] N. Gunantara, *Teknik Optimasi*. 2018.

[5] J. Supratman, "Perencanaan Optimasi Produksi Produk Freezer Dan Showcase Di Pt Fps," *J. PASTI*, vol. 10, no. 3, pp. 320–341, 2016.

[6] A. Ambarwari and N. Yanto, *Penerapan Algoritma Greedy Pada Permasalahan Knapsack Untuk Optimasi Pengangkutan Peti Kemas*. 2016.

[7] Mesran, "Implementasi Algoritma Brute Force Dalam Pencarian Data," no. February, 2014.
[8] I. Rinaldi Munir, "Algoritma Knapsack Disusun oleh," 2004.
[9] Sugiyono, *Metode Penelitian Kuantitatif, Kualitatif dan R&D.* Bandung: PT Alfabet, 2016.