Scientific Journal of Informatics
Vol. 5, No. 1, Nov 2018
p-ISSN 2407-7658 http://journal.unnes.ac.id/nju/index.php/sji e-ISSN 2460-0040

Comparison of Dynamic Programming Algorithm and Greedy Algorithm on Integer Knapsack Problem in Freight Transportation

Global Ilham Sampurno[1], Endang Sugiharti[2], Alamsyah[3]

[1,2,3]Computer Sciences Department, FMIPA, Universitas Negeri Semarang
Email: [1] globalilhams@students.unnes.ac.id, [2]endangsugiharti@mail.unnes.ac.id,
[3]Alamsyah@mail.unnes.ac.id

Abstract

At this time the delivery of goods to be familiar because the use of delivery of goods services greatly facilitate customers. PT Post Indonesia is one of the delivery of goods. On the delivery of goods, we often encounter the selection of goods which entered first into the transportation and held from the delivery. At the time of the selection, there are Knapsack problems that require optimal selection of solutions. Knapsack is a place used as a means of storing or inserting an object. The purpose of this research is to know how to get optimal solution result in solving Integer Knapsack problem on freight transportation by using Dynamic Programming Algorithm and Greedy Algorithm at PT Post Indonesia Semarang. This also knowing the results of the implementation of Greedy Algorithm with Dynamic Programming Algorithm on Integer Knapsack problems on the selection of goods transport in PT Post Indonesia Semarang by applying on the mobile application. The results of this research are made from the results obtained by the Dynamic Programming Algorithm with total weight 5022 kg in 7 days. While the calculation result obtained by Greedy Algorithm, that is total weight of delivery equal to 4496 kg in 7 days. It can be concluded that the calculation results obtained by Dynamic Programming Algorithm in 7 days has a total weight of 526 kg is greater when compared with Greedy Algorithm.

Keywords: Integer Knapsack problem, Greedy Algorithm, Dynamic Programming Algorithm, freight transportation.

1. INTRODUCTION

Along with the development of technology, the completion of a problem that was originally done manually, now can be done systematically through the application. Problem solving process can be done by Algorithm on an application. Algorithms are an effective method of well-defined commands to compute a function. Starting from a start condition and initial input, the instructions describe a computation that when executed or processed through a limited number of sequence conditions can be well defined and produce output [1]. At this time the delivery of goods become familiar things because the business is now a lot of transact on the internet. People will get easier to shop even seller and buyer do not meet in person. That is why freight forwarding services are increasingly needed. Business opportunities and prospects for freight forwarding services are still very good and growing [2]. PT. Pos Indonesia is one of the companies engaged in the delivery of goods and letters in Indonesia [3]. On the delivery of goods, we often encounter the selection of goods to be entered into the

transportation first and who held the delivery. At the time of selection there are Knapsack problems that require optimal selection of solutions. From several previous studies, knapsack problems can be solved using the Greedy Algorithm or the Dynamic Programming Algorithm [4,6,15].

Knapsack problem is a problem that we often find in everyday life [5]. Knapsack is a place used as a means of storing or loading an object. The place can only store objects with a limited total size provision of the Knapsack capacity. Each object does not have to be entered as a whole but can be only part of the object. Many stages are needed to get the problem solved [6]. The knapsack problem is a linear programming problem that has only one constraint [7]. Knapsack is a matter of combinatorial optimization in which we must seek the best solution of the many possibilities produced [8]. The resulting solution is Every object has a certain weight and gain certain advantages. Knapsack can be filled with maximum weight [9]. The purpose of the Knapsack problem is to obtain maximum benefit from the selection of goods without exceeding the capacity of the storage medium [10].

To solve the Knapsack Integer problem in Greedy Algorithm, n input data are done gradually. The first is select the possible solutions which are possible, from the set of possible solutions for optimal solutions [11]. At each step in the Greedy Algorithm, the most optimal decision is made for the step regardless of the consequences of the next step [12]. The Greedy algorithm solves the problem by making the best choice at any given moment [13].

In the Dynamic Programming Algorithm, the problem solving Integer Knapsack is mathematically designed to improve the efficiency level by splitting it into smaller parts of the problem, making it simpler in calculations. To complete the proposed mathematical programming Model Dynamic Programming approach can be used Dynamic Programming is a powerful approach through which a global optimal solution can be obtained in the case of discrete solution space [14]. There are several variables that are considered one of them is the type of goods transported into the transporter or transportation media [15].

Based on the above description, the purpose of this research is to implement the Dynamic Programming Algorithm and Greedy Algorithm on Knapsack Integer Problems in Freight Transportation to obtain the most optimal transporting solution.

2. METHODS

The methods applied in this research are dynamic programming algorithm and greedy algorithm.

2.1. Dynamic Programming Algorithm

Dynamic Programming is a problem-solving method by outlining the solution into a set of steps or steps so that the solution of the problem can be viewed from a series of interconnected decisions [16].

The first step in the Dynamic Programming Algorithm stage determines the structure

of the problem. From the known problem n goods. The maximum transport capacity of 750 kg is denoted as M. Wi is denoted as the weight of item i, and Pi is denoted as the postage of goods i. Formulated as $M \in n, W_i \leq M$, $i \in n$, with limits $\sum_{i=1}^{n} w_i x_i \leq M$. The optimum total profit is affected by the postage denoted as $Z = \sum_{i=1}^{n} p_i x_i$. To obtain the optimum total profit is calculated by forward recursive procedure.

The next step is to determine the recursive equation to maximize the optimal profit. From the equation Z can be written with a recursive equation to look for all possible haulage of goods. The value of the optimal transport solution is entered into z(i, j), so it can be written the recursive equation fk(y) = max { fk-1(y),pk + fk-1 (y – wk) } , k = 1,2,.. ,n.

The pseudecode Dynamic Programming Algorithm is shown on Figure 1.

```
Input:
 ongkoskirim (v[])
 Berat barang (w[])
 Jumlah item barang (n)
 Kapasitas Knapsack (W)
for j from 0 to W do
    m[0, j] := 0
end for
for i from 1 to n do
      for j from 0 to W do
               if w[i] <= j then
               m[i, j] := max(m[i-1, j], m[i-1, j w[i]] + v[i])
               else
               m[i, j] := m[i-1, j]
               end if
       end for
end for
```

Figure 1. Pseudecode Dynamic Programming Algorithm

Then calculate the value of the optimal solution, the calculation procedure used is forward recursive procedure. The calculation starts from finding the profit value for the first item to the last item of the total goods. The result of data calculation using Dynamic Programming Algorithm shown in Table 1.

Tabel 1. The Calculation Results Using Dynamic Programming Algorithm

Date	Total Weight	Total Shipping Cost	Amount of Data
1st july 2017	772 kg	Rp. 25.495.500	31 data
2nd juli 2017	581 kg	Rp. 6.857.000	10 data
3rd juli 2017	750 kg	Rp. 35.730.500	35 data
4th juli 2017	750 kg	Rp. 40.233.000	39 data

5th juli 2017	744 kg	Rp. 31.698.000	34 data
6th juli 2017	725 kg	Rp. 37.829.000	32 data
7th juli 2017	750 kg	Rp. 32.292.500	25 data
8th juli 2017	719 kg	Rp. 11.457.500	15 data
9th juli 2017	723 kg	Rp. 7.297.000	2 data
10th juli 2017	516 kg	Rp. 4.881.000	2 data
11th juli 2017	278 kg	Rp. 2.000.000	1 data

Based on the calculation results in Table 1 it is found that the Dynamic Programming Algorithm of the remaining shipment requires 4 times of transport, That is, 8.9,10 and 11 July 2017.

2.2. Greedy Algorithm

The first step in the Greedy Algorithm phase is to sort the data. Greedy used in this study is Greedy by density data sorted by density (bulk sharing divided weight), then taken one by one object that can be accommodated by the storage until the place where the storage is full or no more objects are inserted into the storage . The result of data calculation using Greedy Algorithm shown in Table 2.

Table 2. The Calculation Results Using Greedy Algorithm

Date	Total Weight	Total Shipping Cost	Amount of Data
1st july 2017	722 kg	Rp. 25.495.500	31 data
2nd july 2017	581 kg	Rp. 6.857.000	10 data
3rd july 2017	560 kg	Rp. 30.270.500	37 data
4th july 2017	659 kg	Rp. 34.609.000	34 data
5th july 2017	706 kg	Rp. 32.937.000	32 data
6th july 2017	570 kg	Rp. 30.562.000	28 data
7th july 2017	698 kg	Rp. 36.651.500	30 data
8th july 2017	672 kg	Rp. 14.456.000	5 data
9th july 2017	724 kg	Rp. 8.781.000	10 data
10th july 2017	706 kg	Rp. 8.526.000	4 data
11th july 2017	660 kg	Rp. 6.625.000	5 data

Based on calculations in Table 2 it is found that the Greedy Algorithm of the remaining shipment requires 4 times of transport, That is, 8.9,10 and 11 July 2017.The Greedy Algorithm is an Algorithm to solve problems gradually [6]. The completion stage are:

1. Take the best options that can be obtained at that moment regardless of the consequences of the future.
2. Hope that by choosing the local optimum at each step will end up with global optimum.

The pseudecode Greedy Algorithm is shown on Figure 2.

```
Input:
p(1:n), w(1:n), x(1:n), m, isi,n
x(1:n) = 0
isi = m
for i = 1 to n do
       if w(i) > isi then
              Exit
        endif
       x(i) = 1
       isi = isi – w(i)
repeat
       if i ≤ n then
               x(i) = isi/w(i)
       endif
```

Figure 2. Pseudecode Greedy Algorithm

2.3. Process Design

The Steps of Making The System:

1. The data collection stage
   The data collection stage is the step done to get the data related and needed in the research [17].
2. Literature Studies
   This stage is the stage through the literature relevant to the research. This phase is carried out to obtain the relevant information related to the research [18]. At this stage also collected materials, information, explanations, and theory in the book and consultation with experts or interviewees and references from articles, journals, and other scientific papers relevant and related to the object of research, and methods used in research [19].
3. Field Studies
   This stage is the search data in the field. this stage is done to understand, study, and obtain information. This method of observation is useful for researchers to collect data in various ways [20].
4. Implementation
   This stage is the process of building a system based on the results of analysis and system design that has been done before. Once the system is built, the data obtained on the collection of data is processed on a system that has been built [21].
5. Testing
   This stage is a testing process conducted by using data and implementation results.

2.4. System Development

The design of this system using waterfall model. Waterfall is an approach method based on the assumption that major decisions must be made before encoding Begins [22]. Waterfall is a software development methodology that proposes a systematic and sequential software approach, starting at the system advance level, all of the

analysis, design, code, testing and maintenance [23].This model is named "Waterfall" because its diagrammatic representation looks like a cascade (flow) of Waterfall. This is also known as classical lifecycle model [24]. This model is often used by systems analysts in general [25]. There are four stages in the waterfall model. The stages are, requirement analysis, design, implementation and testing [26].

1. The requirement analysis stage is defining the entire software format, identifying all the needs, and outlines of the created system [27]. This stage intends to identify and evaluate the problems and needs required, this stage includes the analysis of hardware requirements, software requirements analysis, user requirements analysis and requirements analysis process [28].
2. The design stage is to design applications including interface design, and database structure design [29]. The design stage is the process of translation systems in accordance algorithm used [30].
3. The implementation stage is designing software which realized as a series of program or program unit [31].
4. The testing stage is to test whether the system is ready and feasible to use. The tester can define the set of input conditions and perform testing on functional specifications of the program [32]. To check the success of the system, performed the testing stage. This stage is to test the errors that exist in making the program [33]. Verification is a software evaluation Process to Determine wether a product of given development phase statisfy the conditions imposed at the begining of that stage [34].

3. RESULTS AND DISCUSSION

3.1. Data Collecting

In this research, the data used are the record of goods delivery in PT POS Indonesia in Semarang. Data obtained from PT POS Indonesia in Semarang as much as 226 data in a week in July. The following table presents data on the delivery records of goods from 1 to 7 July 2017.

Table 3. Data Record of Goods Delivery 1 – 7 July 2017

Date	Total Weight	Total Shipping Cost	Amount of Data
1st july 2017	802 kg	Rp. 26.581.500	32 data
2nd july 2017	501 kg	Rp. 5.771.000	9 data
3rd july 2017	1057 kg	Rp. 42.397.500	43 data
4th july 2017	739 kg	Rp. 39.318.000	36 data
5th july 2017	1764 kg	Rp. 46.103.000	38 data
6th july 2017	1329 kg	Rp. 44.748.500	33 data
7th july 2017	616 kg	Rp. 30.851.500	35 data

In Knapsack problem, there are problem in the storage. Where it can only store objects with a limited total size provision of the Knapsack capacity. The situation can be described as a state of overload. In this research record data delivery of goods in PT POS Indonesia Semarang there Knapsack problems. Because if known Maximum transportation capacity of 750 kg while the total weight of delivery on 1,3, and July 6, 2017 exceeds the maximum transportation capacity. So in the data there are Knapsack

problems . as long as there is a state of overload that can be in search for the optimal solution using Greedy Algorithm or Dynamic Programming Algorithm.

3.2. Data Calculating

In the application of Greedy Algorithm, using data from July 1, 2017 until July 7, 2017. Obtained total weight remaining shipment of 2762 kg. While the application of Dynamic Programming Algorithm obtained the total weight of the remaining shipments of 2236 kg. Comparison of retrieval solutions from both algorithms shown in Table 4.

Tabel 4. Comparison of Goods Selection Solutions from Both Algorithms.

Date	Total Calculation Results		
	Algorithm	Weight	Shipping Cost
1st july 2017	*Greedy*	722 kg	Rp. 25.495.500
	Dynamic Programming	722 kg	Rp. 25.495.500
2nd july 2017	*Greedy*	581 kg	Rp. 6.857.000
	Dynamic Programming	581 kg	Rp. 6.857.000
3rd july 2017	*Greedy*	560 kg	Rp. 30.270.500
	Dynamic Programming	750 kg	Rp. 35.730.500
4th july 2017	*Greedy*	659 kg	Rp. 34.609.000
	Dynamic Programming	750 kg	Rp. 40.233.000
5th july 2017	*Greedy*	706 kg	Rp. 32.937.000
	Dynamic Programming	744 kg	Rp. 31.698.000
6th july 2017	*Greedy*	570 kg	Rp. 30.562.000
	Dynamic Programming	725 kg	Rp. 37.829.000
7th july 2017	*Greedy*	698 kg	Rp. 36.651.500
	Dynamic Programming	750 kg	Rp. 32.292.500
8th july 2017	*Greedy*	672 kg	Rp. 14.456.000
	Dynamic Programming	719 kg	Rp. 11.457.500
9th july 2017	*Greedy*	724 kg	Rp. 8.781.000
	Dynamic Programming	723 kg	Rp. 7.297.000
10th july 2017	*Greedy*	706 kg	Rp. 8.526.000
	Dynamic Programming	516 kg	Rp. 4.881.000
11th july 2017	*Greedy*	660 kg	Rp. 6.625.000
	Dynamic Programming	278 kg	Rp. 2.000.000

Based on the results of the calculation by two algorithms, it can be seen that the solution of making goods using Dynamic Programming Algorithm is better when compared with Greedy Algorithm or choose randomly without Algoritma. This can be seen from the calculation result obtained by Dynamic Programming Algorithm, that is total weight of 5022 kg in 7 days. While the calculation result obtained by Greedy Algorithm, that is total weight of delivery equal to 4496 kg in 7 days.

4. CONCLUSION

The solution of goods using Dynamic Programming Algorithm is better when compared with Greedy algorithm or choose randomly without Algorithm. This can be seen from the calculation result obtained by Dynamic Programming Algorithm, that is total weight of 5022 kg in 7 days. While the calculation result obtained by Greedy Algorithm, that is total weight of delivery equal to 4496 kg in 7 days. the retrieval

solution using the Dynamic Programming Algorithm in 7 days has a total shipping weight of 526 kg larger when compared to the Greedy Algorithm.

5. REFERENCES

[1] Alamsyah & Putri, I. T. (2014). Penerapan Algoritma Greedy Pada Mesin Penjual Otomatis (Vending Machine). *Scientific Journal of Informatics*, 1(2), 201-209.

[2] Trianto, E., & Revina, W. Perancangan Sistem Iinformasi Pencatatan Pengiriman Barang di PT. TIKI Jalur Nugraha Ekakurir Cabang Bandung.

[3] Fitriana, E. N., & Sugiharti, E. (2015). Implementasi Algoritma Genetika dengan Teknik Kendali Logika Fuzzy untuk Mengatasi Travelling Salesman Problem Menggunakan Matlab. *Unnes Journal of Mathematics*, 4(2).

[4] Husni, T. S. & Arief, A. (2008). Implementasi 0-1 Knapsack Menggunakan Algoritma Dynamic Programming Pada Aplikasi Perhitungan Harga Satuan Produk Percetakan Berbasis Web (Studi Kasus: Cv Tunas Utama). *Jurnal InforSAINS* 2(3): 31-35.

[5] Supriana, I. W. (2016). Optimalisasi Penyelesaian Knapsack Problem Dengan Algoritma Genetika. Lontar Komputer: Jurnal Ilmiah Teknologi Informasi, 182-192.

[6] Paryati, P. (2015). Optimasi Strategi Algoritma Greedy untuk Menyelesaikan Permasalahan Knapsack 0-1. In Seminar Nasional Informatika (SEMNASIF), 1(1): 101-110.

[7] Isnaeni, S., Supriyono, S., & Arini, F. Y. (2014). Implementasi Algoritma Pemrograman Dinamik Untuk Penyelesaian Persoalan Knapsack Dalam Penentuan Keuntungan Optimal Penjualan Barang. *Unnes Journal of Mathematics*, 3(2): 97-102.

[8] Puspa, S. K., Mrunal, T. V. & Suhas. C. (2016). A Study of Performance Analysis on Knapsack Problem. *International Journal of Computer Applications*, 0975 – 8887:1

[9] Pajjan, S. P., Roogi, R., Badiger, V., & Amaragatti, S. (2014). A New Approach to Solve Knapsack Problem. *Oriental Journal Of Computer Science & Technology*, 7(2):219.

[10] Supriadi, D. (2016). Perbandingan Penyelesaian Knapsack Problem Secara Matematika, Kriteria Greedy Dan Algoritma Greedy. *Indonesian Journal on Computer and Information Technology*,1(2):91.

[11] V. Springer. (2005). Knapsack 0-1 Problem. Yogyakarta: Graha Ilmu.

[12] Faisal. (2014). Penerapan Metode Greedy Knapsack Dalam Menentukan Komposisi Buah Pada Masalah Keranjang. *Jurnal Teknologi Informasi*, 10(2): 32-37.

[13] Malik, A., Sharma, A., Saroha, V. (2013). Greedy Algorithm. *International Journal of Scientific and Research Publications*, 3(8):1.

[14] Tari, F.G. (2016). A Hybrid Dynamic Programming for Solving Fixed Cost Transportation with Discounted Mechanism. *Hindawi Publishing Corporation Journal of Optimization*, 2016(8518921):1-9.

[15] Pratiwi, A & Rochmad M. (2014). Implementasi Algoritma Branch And Bound Pada 0-1 Knapsack Problem Untuk Mengoptimalkan Muatan Barang. *Unnes*

Journal of Mathematics, 3(2): 90-96.
[16] Surjawan, D, J & Susanto I. (2015). Aplikasi Optimalisasi Muat Barang Dengan Penerapan Algoritma Dynamic Programming Pada Persoalan Integer Knapsack. *Jurnal Teknik Informatika dan Sistem Informasi*, 1(2):151-162.
[17] Ashari A. I., Muslim, M.A., & Alamsyah. (2016). Comparison Performance of Genetic Algorithm and Ant Colony Optimization in Course Scheduling Optimizing. *Scientific Journal of Informatics*, 3(2): 150.
[18] Sugiharti, E., Firmansyah, S., & Devi, F.R. (2017). Redictive Evaluation Of Performance Of Computer Science Students Of Unnes Using Data Mining Based On Naïve Bayes Classifier (Nbc) Algorithm. *Journal of Theoretical and Applied Information Technology*, 95(4): 905.
[19] Amin, S., Muslim, M.A., & Alamsyah. (2012). Sistem Deteksi Dini Hama Wereng Batang Coklat Menggunakan Jaringan Syaraf Tiruan Backpropagation. *Unnes Journal of Mathematics*, 1(2): 119.
[20] Alamsyah & Arus, A.A. (2014). Analisis Sistem Pendaftaran pada Web Forum Ilmiah Matematika Unnes. *Scientific Journal of Informatics*, 1(1): 110.
[21] Sugiharti, E., & Muslim, M.A. (2016). On-Line Clustering Of Lecturers Performance Of Computer Science Department Of Semarang State University Using K-Means Algorithm. *Journal of Theoretical and Applied Information Technology*, 83 (1): 66.
[22] Patel, U.A., & Jain N.K. (2013). New Idea In Waterfall Model For Real Time Software Development. *International Journal of Engineering Research & Technology (IJERT)*, 2(4): 115.
[23] Rukmana, S.H., & Muslim, M. A. (2016). Sistem Pendukung Keputusan Tender Proyek Menggunakan Metode Benefit Cost Ratio. *Jurnal Sains & Teknologi*, 5(2): 817-822.
[24] Saxena, A., & Upadhyay, P. (2016). Waterfall vs. Prototype: Comparative Study of SDLC. *Imperial Journal of Interdisciplinary Research (IJIR)*, 2(6):1012-1013.
[25] Roviaji, R., & Muslim M.A. (2017. Pembuatan Sistem Informasi Gardu Induk PT. Pln (Persero) App Semarang Se-Kota. Prosiding Seminar Ilmu Komputer dan Teknologi Informasi. Samarinda: Universitas Mulawarman.
[26] Pressman, R.S. (2001). Software Engineering. Online. Tersedia di http://www.resource.mitfiles.com/ [diakses 18-2-2017].
[27] Nugroho, Z.A., & Arifudin, R. Sistem Informasi Tracer Study Alumni Universitas Negeri Semarang Dengan Aplikasi Digital Maps. *Scientific Journal of Informatics*, 1(2): 154.
[28] Rukhansah, N., Muslim, M. A., & Arifudin, R. (2016). Peramalan Harga Emas Menggunakan Fuzzy Time Series Markov Chain Model. *KOMPUTAKI*, 1(1): 68.
[29] Putra, A.T. 2014. Pengembangan E-Lecture menggunakan Web Service Sikadu untuk Mendukung Perkuliahan di Universitas Negeri Semarang. *Scientific Journal of Informatics*, 1(2): 170.

[30] Muslim, M.A., Kurniawati I., & Sugiharti E. (2015). Expert System Diagnosis Chronic Kidney Disease Based on Mamdani Fuzzy Inference System. *Journal of Theoretical and Applied Information Technology*, 78(1): 71.
[31] Purwinarko, A. (2014). Model Expertise Management System di Universitas Negeri Semarang. *Scientific Journal of Informatics*, 1(2): 178.
[32] Mustaqbal, M.S., Firdaus, F.S., Rahmadi, H. (2015). Pengujian Aplikasi Menggunakan Black Box Testing Boundary Value Analysis. *JITTER*, 1(3): 35.
[33] Sugiharti E & Triliani, S. E. (2014). Perancangan Aplikasi Surat Masuk dan Keluar pada PT. Angkasa Pura 1 Semarang. *Scientific Journal of Informatics*, 1(1): 41.
[34] Hardyanto, W., Purwinarko, A., Sujito, F., Masturi., Alighiri., D. (2016). Applying an MVC Framework For The System Development Life Cycle With Waterfall Model Extended. *Journal of Physics: Conference Series*, 824 (1): 3.