BASIS PATH TESTING OF ITERATIVE DEEPENING SEARCH
AND HELD-KARP ON PATHFINDING ALGORITHM

aI Gede Surya Rahayuda, bNi Putu Linda Santiai
a,bSTMIK STIKOM Bali, Jl. Raya Puputan Renon No. 86 Denpasar, 80234
E-mail: surya_rahayuda@stikom-bali.ac.id

Abstract
This research is a continuation of previous research, where in previous research discussed about the implementation of Iterative Deepening Search and Held-Karp Method. Both methods are compared based on the results obtained. While in the current study, research is more focused on the analysis of the program flow that has been made. Evaluation is done by using basis path method, there are several processes performed on the method, such as: flowgraph, independent path, cyclomatic complexity and graph matrix. In addition to the evaluation of program flow, evaluation is also done based on program performance. Performance tests are based, time, cpu and memory. Based on the evaluation using the base path, obtained flowgraph structure and independent path different, but obtained the result of cyclomatic complexity and graph matrix calculation of the same between Iterative Deepening Search and Held-Karp method is 4. Based on evaluation in terms of performance, process the program from entering data and until getting the result, the Held-Karp method takes a longer time than the Iterative Deepening Search method. The Iterative Deepening Search method takes 2.7 seconds while the Held-Karp method takes 2.8 seconds.

Keywords: Held-Karp, Iterative Deepening Search, Basis Path, Cyclomatic complexity, Graph Matrix
INTRODUCTION

This research is extended research from previous research, where in previous research more discussed about implementation of Iterative Deepening Search and Held-Karp method [1][2]. In that research also discussed about the comparison of the results of both methods. In this research will be more about the evaluation or analysis of the script program that has been made.

Of the two methods that have been implemented into the program in previous research, will be evaluated using the basis path method [3][4]. The basis path method is one of the white box testing methods, where in the testing process needed to create a flowgraph of the script program and also determine the value of cyclomatic complexity. This test aims to analyze the correctness of the structure of the program created and the performance of the program [5][6].

In addition to testing using the basis path method, evaluation is also done based on the speed of processing the program to get the results, the processor and memory performance required during the program.

Of all the tests performed, then the results of the evaluation data are then compared in the form of tables, so that later can be analyzed about the differences of both methods based on various tests that have been done.

Regarding similar research that has been done, the author will begin to explain about previous research that has been done by the author, where the paper has been published in national seminars and journals, research ever conducted on path search method, the method of Iterative Deepening Search and Held-Karp.

My first research on path search methods, the study was titled “Penerapan Pemrograman Dinamis pada Manajemen Pengiriman Produk Menggunakan Metode Held-Karp”. The research is my preliminary study on the use of path search algorithms. In the field of informatics science, various path search methods are called dynamic programming. In that study I chose to use the Held-Karp method used to determine the closest route of a product delivery trip. The method is very appropriately applied to the case of product delivery. Because the Held-Karp method find for the shortest path from the starting point to all of the route point and then back to the starting point. In contrast to other path search algorithms, where generally the method only looks for the shortest path from the starting point to the destination point. In that research developed Held-Karp method using visual basic language in console based desktop program [7].

Publication of the second phase of research on pathfinding method entitled “Implementasi dan Perbandingan Metode Iterative Deepening Search dan Held-Karp pada Manajemen Pengiriman Produk”. The research is a continuation of previous research, whereas in this research developed Held-Karp method on visual based desktop program [8]. In addition to that research also use other pathfinding method, Iterative Deepening Search method, Iterative Deepening Search method is the development of Depth First Search method, where the Iterative Deepening Search method applied a recursive process or repetition if in a case the path is not found, the process will then repeated and done until the paths are actually found. But if the method is applied to the product delivery will have some deficiency, because the Iterative Deepening Search method only finds the path from the starting point to the destination point. To overcome this the authors add some additional scripts for the path from the end point to the starting point, so that will get the total results of travel distance from the starting point to all points and back to the starting point. Both programs are compared by inputting the same test data, with the same test data, the two programs produce different routes, when compared to the results obtained, the Held-Karp method gets better route results than the Iterative Deepening Search method.

PATH FINDING USING ITERATIVE DEEPENING SEARCH AND HELD-KARP METHOD

In the previous research has been discussed and solved two desktop pathfinding programs using Iterative Deepening Search and Held-Karp Method. In the program as depicted in Figure 1, provides 5 pieces of combobox to determine which subscription stores that order gas to be delivered. It is assumed that in the case of product shipments there are 5 orders. Administrator will determines the delivery route of the product using the program and
sends the product according to the route specified in the program, as seen in Figure 2.

The same program is done using the Held-Karp method, as shown in Figure 3. In some cases of product delivery Held-Karp and Iterative Deepening Search get different results. The search route can be seen in Figure 4.

The use of both programs, tested in some cases of product delivery, is the comparison table from the previous study, shown in Table 1. There are 20 product shipments, where each shipment contains 5 product shipping locations.

<table>
<thead>
<tr>
<th>pase</th>
<th>Start Route</th>
<th>IDS Route</th>
<th>dist.</th>
<th>HK Route</th>
<th>dist.</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>R.Gane Subravo</td>
<td>R.Gane Subravo</td>
<td>21 Km</td>
<td>R.Gane Subravo</td>
<td>30 Km</td>
</tr>
<tr>
<td>P2</td>
<td>R.Mortari</td>
<td>R.Karang</td>
<td>32 Km</td>
<td>R.Karang</td>
<td>40 Km</td>
</tr>
<tr>
<td>P3</td>
<td>R.Antara</td>
<td>R.Kebabayan</td>
<td>38 Km</td>
<td>R.Kebabayan</td>
<td>46 Km</td>
</tr>
<tr>
<td>P4</td>
<td>R.Kebabayan</td>
<td>R.Mortari</td>
<td>59 Km</td>
<td>R.Mortari</td>
<td>66 Km</td>
</tr>
<tr>
<td>P5</td>
<td>R.Kebabayan</td>
<td>R.Antara</td>
<td>59 Km</td>
<td>R.Antara</td>
<td>65 Km</td>
</tr>
<tr>
<td>P6</td>
<td>R.Antara</td>
<td>R.Gane Subravo</td>
<td>22 Km</td>
<td>R.Gane Subravo</td>
<td>30 Km</td>
</tr>
<tr>
<td>P7</td>
<td>R.Mortari</td>
<td>R.Kebabayan</td>
<td>22 Km</td>
<td>R.Kebabayan</td>
<td>30 Km</td>
</tr>
<tr>
<td>P8</td>
<td>R.Kebabayan</td>
<td>R.Mortari</td>
<td>39 Km</td>
<td>R.Mortari</td>
<td>45 Km</td>
</tr>
<tr>
<td>P9</td>
<td>R.Mortari</td>
<td>R.Antara</td>
<td>44 Km</td>
<td>R.Antara</td>
<td>50 Km</td>
</tr>
<tr>
<td>P10</td>
<td>R.Antara</td>
<td>R.Mortari</td>
<td>44 Km</td>
<td>R.Mortari</td>
<td>50 Km</td>
</tr>
<tr>
<td>P11</td>
<td>R.Mortari</td>
<td>R.Kanada</td>
<td>50 Km</td>
<td>R.Kanada</td>
<td>60 Km</td>
</tr>
<tr>
<td>P12</td>
<td>R.Kanada</td>
<td>R.Mortari</td>
<td>50 Km</td>
<td>R.Mortari</td>
<td>60 Km</td>
</tr>
<tr>
<td>P13</td>
<td>R.Mortari</td>
<td>R.Kanada</td>
<td>50 Km</td>
<td>R.Kanada</td>
<td>60 Km</td>
</tr>
<tr>
<td>P14</td>
<td>R.Kanada</td>
<td>R.Mortari</td>
<td>50 Km</td>
<td>R.Mortari</td>
<td>60 Km</td>
</tr>
</tbody>
</table>

Table 1. Test Results on Some Data Test
methods determine the same path on the transmission. The mean difference of the two methods based on the distance is 11.35 km. Overall the Held-Karp method gets better results from the Iterative Deepening Search method, see Table 2, the total overall difference of the two methods is 19.66%.

BASIS PATH TESTING

The evaluation was performed using the basis path method [9], the processes performed on the basis path method, as seen in Figure 5, are:
1. Flowgraph
2. Independent Path
3. Cyclical Complexity
4. Graph Matrix

In addition to using the base path method, evaluation is also performed based on the performance of the computer when the program is run [10], some performance tests performed are:
1. Speed Performance
2. CPU Performance
3. Memory Performance

After basis path testing and performance then next do the comparison of test results that have been done.

![Figure 5. Research Method](image)

RESULT AND DISCUSSION

Iterative Deepening Search

Iterative deepening search or often referred to as iterative deepening depth first search is

### Table 2. Comparison of Both Method on Some Data Test

<table>
<thead>
<tr>
<th>Process</th>
<th>Distance HK</th>
<th>Distance IDS</th>
<th>Average</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>41 Km</td>
<td>60 Km</td>
<td>19 Km</td>
<td>32%</td>
</tr>
<tr>
<td>P2</td>
<td>63 Km</td>
<td>70 Km</td>
<td>7 Km</td>
<td>10%</td>
</tr>
<tr>
<td>P3</td>
<td>33 Km</td>
<td>48 Km</td>
<td>15 Km</td>
<td>31%</td>
</tr>
<tr>
<td>P4</td>
<td>32 Km</td>
<td>44 Km</td>
<td>12 Km</td>
<td>27%</td>
</tr>
<tr>
<td>P5</td>
<td>81 Km</td>
<td>93 Km</td>
<td>12 Km</td>
<td>13%</td>
</tr>
<tr>
<td>P6</td>
<td>57 Km</td>
<td>78 Km</td>
<td>21 Km</td>
<td>27%</td>
</tr>
<tr>
<td>P7</td>
<td>37 Km</td>
<td>62 Km</td>
<td>25 Km</td>
<td>40%</td>
</tr>
<tr>
<td>P8</td>
<td>55 Km</td>
<td>75 Km</td>
<td>20 Km</td>
<td>27%</td>
</tr>
<tr>
<td>P9</td>
<td>38 Km</td>
<td>51 Km</td>
<td>13 Km</td>
<td>25%</td>
</tr>
<tr>
<td>P10</td>
<td>60 Km</td>
<td>68 Km</td>
<td>8 Km</td>
<td>12%</td>
</tr>
<tr>
<td>P11</td>
<td>37 Km</td>
<td>66 Km</td>
<td>29 Km</td>
<td>44%</td>
</tr>
<tr>
<td>P12</td>
<td>59 Km</td>
<td>66 Km</td>
<td>7 Km</td>
<td>11%</td>
</tr>
<tr>
<td>P13</td>
<td>40 Km</td>
<td>43 Km</td>
<td>3 Km</td>
<td>7%</td>
</tr>
<tr>
<td>P14</td>
<td>28 Km</td>
<td>28 Km</td>
<td>0 Km</td>
<td>0%</td>
</tr>
<tr>
<td>P15</td>
<td>35 Km</td>
<td>35 Km</td>
<td>0 Km</td>
<td>0%</td>
</tr>
<tr>
<td>P16</td>
<td>65 Km</td>
<td>69 Km</td>
<td>4 Km</td>
<td>6%</td>
</tr>
<tr>
<td>P17</td>
<td>42 Km</td>
<td>42 Km</td>
<td>0 Km</td>
<td>0%</td>
</tr>
<tr>
<td>P18</td>
<td>27 Km</td>
<td>37 Km</td>
<td>10 Km</td>
<td>27%</td>
</tr>
<tr>
<td>P19</td>
<td>67 Km</td>
<td>80 Km</td>
<td>13 Km</td>
<td>16%</td>
</tr>
<tr>
<td>P20</td>
<td>50 Km</td>
<td>59 Km</td>
<td>9 Km</td>
<td>15%</td>
</tr>
</tbody>
</table>

Regarding the difference of path search result from Held-Karp method and Iterative Deepening Search method can be seen in Table 2. In the table it can be seen that there is not much difference between the two methods. The greatest distance difference is seen in the delivery of P11 about 43%. The lowest distance difference is seen in the delivery of P14, P15 and P17, there is a difference of 0, or on the delivery both methods get the same result. The Iterative Deepening Search and Held-Karp

### Performance

**Script Program:**
- Iterative Deepening Search
- Held-Karp

**Flowgraph**
- Independent Path
- Cyclical Complexity

**Comparison Table**

**Performance**
- Speed Test
- CPU Usage
- RAM Usage
one of the pathfinding method which is the development of depth first search method [6][11].

The first version of the depth first search method was discovered around 1900, by the French mathematical scientist Charles Pierre Tremaux as a strategy to overcome the problem of searching the path on the labyrinth. Expressed from several sources, that the logic of the Depth First Search method is to do a preliminary search of the deepest boundary, or searched from the end of a graph, from the last graph sought the smallest number until it is found, then proceeded to the previous boundary and done the same thing, initial limit [12][13].

Iterative deepening search method is the development of depth first search method. Where in the Iterative deepening search method adds several possible repetitions on each search until a route is found. In contrast to the depth first search method, in the depth first search method there is a possibility that the route can not be found, whereas in the Iterative deepening depth first search method if in a certain depth the route is not found, it will be repeated by looking back for the second smallest number at the depth, the next limit, and so on until the route is found [14][15].

Iterative deepening search method has been implemented in previous research [6][16], the flowgraph will be analyzed in order to be evaluated.

**Flowgraph Iterative Deepening Search**

From the script program is then analyzed and recreated in the form of flowgraph, so that later can be calculated cyclomatic complexity and searched independent path [15][17], program flow from Iterative Deepening Search is:

a. Initialize the point list as many as the number of points, and enter each point into the list

b. Initialize the path list according to the available data

c. Specify the start point of the search path

d. Perform percountan process starting from the depth or first depth to the limit or maximum limit of depth

e. For each pre-defined limit, Perform a path search counter through all available points, an explanation of this function will be described in the following table (points 5a - 5e)

e.1 Enter the starting point as the first point to look for the path

e.2 Then enter the other points in sequence from the last point to the first point

e.3 Initialize the depth or depth variable to know the depth at the time of search path

e.4 Enter the first point into the path to be discounted

e.5 Perform percountan as long as there are still points that have not been mounted (points e.5.1 - e.5.4)

e.5.1 Enter the next point into the current discount path

e.5.2 Count depth at the point being counter, If the depth value is 0, then give the initial value of depth, besides it subtract the depth value by 1

e.5.3 If there is a path between the point before and the next point, and all points have been mounted all or the number of points that are being mounted has not exceeded the limit or maximum limit of depth, then check whether all points have been mounted all, If the condition is true, then stop percountan

e.5.4 If the above conditions are not met, then do the following percountan (points 5e4a - 5e4g)

e.5.4.1 Show error message if the number of points being discount has exceeded the limit or maximum limit of depth

e.5.4.2 Show error message if there is no path between point before and point next

e.5.4.3 Reduce depth in the position of the counter points for each remaining point that is not mounted

e.5.4.4 And for the rest of the point, give the depth value to 0

e.5.4.5 For all points with a depth value of 0, Eliminate those points from the path that is being mounted, and re-enter the points into the list of points that have been counter

e.5.4.6 Determine the next point with the order of Nearby after the last point is mounted, That is the point with the lowest negative index difference

e.5.4.7 After finding the Near point, Sort all uneded points besides point Nearby from point last to first point, Then enter this Nearby point in the first sequence before the sequence
f. After finding the path, then count distance to that path

Then all the points on the program script then put in a step count of numbers as seen in Table 3, and translated into structured tree of flowgraph as shown in Figure 6.

**Independent Path Iterative Deepening Search**

Based on the flowgraph that has been made then can be determined independent path from the Iterative Deepening Search method [18]. independent path that formed 4 lines, such as:

Path 1 : 1, 2, 3, 4, 5, 6, 7, 8, 9, 22
Path 2 : 1, 2, 3, 4, 5, 10, 11, 12, 13, 22
Path 3 : 1, 2, 3, 4, 5, 10, 14, 15, 16, 17, 18, 19, 20, 22
Path 4 : 1, 2, 3, 4, 5, 10, 14, 21, 22

**Cyclomatic Complexity Iterative Deepening Search**

Furthermore, it can be determined the value of cyclomatic complexity, the value of cyclomatic complexity is determined in order to know whether the algorithm used is relevant or not [5][19]. The components used to calculate cyclomatic complexity can be seen from the previously calculated Flowgraph and Independent Path. Here is a process to calculate the cyclomatic complexity value of the method.

\[
VG = 4 \\
VG = 24 \text{ edge} - 22 + 2 = 4 \\
VG = 3 \text{ predicate node} + 1 = 4
\]

**Graph Matrix Iterative Deepening Search**

In addition to calculating cyclomatic complexity, basis path evaluation can also be done by finding the value of the graph matrix [20]. Here is the process of calculating the matrix graph of the Iterative Deepening Search algorithm as seen in Table 4.

**Table 3. Iterative Deepening Search Program Flow and Point List**

<table>
<thead>
<tr>
<th>a = 1</th>
<th>e.2 = 7</th>
<th>e.3 = 13</th>
<th>e.5.4.5 = 19</th>
</tr>
</thead>
<tbody>
<tr>
<td>b = 2</td>
<td>e.3 = 8</td>
<td>e.5.4 = 14</td>
<td>e.5.4.6 = 20</td>
</tr>
<tr>
<td>c = 3</td>
<td>e.4 = 9</td>
<td>e.5.4.1 = 15</td>
<td>e.5.4.7 = 21</td>
</tr>
<tr>
<td>d = 4</td>
<td>e.5 = 10</td>
<td>e.5.4.2 = 16</td>
<td>f = 22</td>
</tr>
<tr>
<td>e = 5</td>
<td>e.5.1 = 11</td>
<td>e.5.3 = 17</td>
<td></td>
</tr>
<tr>
<td>e.1 = 6</td>
<td>e.5.2 = 12</td>
<td>e.5.4.4 = 18</td>
<td></td>
</tr>
</tbody>
</table>

**Table 4. Graph Matrix Iterative Deepening Search**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |

**Figure 6. Flowgraph Iterative Deepening Search**
Flowgraph Held-Karp

The program flow from Held-Karp is:

a. Initialize the point list as many as the number of points, and enter each point into the list
b. Initialize the path list according to the available data
c. Specify the start point of the search path
d. Determine whether the invitations that are made only until all the points have been passed, or until the return to the starting point. If the input is yes, then the path sought must be back to the starting point. If the input is not, then the path searched only to all points already passed
e. Perform a search for the path with the lowest distance. Explanation of the function will be explained in percountan below (points e.1 - e.3)
   e.1 Initialize the initial point list alongside the initial point, then enter all the points alongside the initial point into the list
   e.2 Perform a reply to the lowest distance search on all points. The explanation of the function will be explained in the following table (points e.2.1 - e.2.4)
      e.2.1 Perform sums on all point purposes
      e.2.2 Add a branch of all point purposes at the starting point
      e.2.3 Determine distance path awal dari point purpose
      e.2.4 Determine the initial distance path from the starting point to point purpose
      e.2.4.1 Create a new point list for all points beside starting point and point purpose
      e.2.4.2 If the distance path on the point to the new point is more than 0, then add that distance to the initial distance
e.3 Find the path of points with the lowest distance that has been found before. Explanation of the function will be explained in percountan below
f. Record the final result for each line answer found. Then show the total distance path

Step count number and flowgraph can be seen in Table 5 and Figure 7.

Independent Path Held-Karp

Similar to what happens with Iterative Deepening Search methods, in the Held-Karp method, the flowgraph from the Held-Karp method will be analyzed in order to be evaluated.
method there are also 4 independent paths was formed [26][27], as:

Path 1: 1, 2, 3, 4, 5, 6, 14, 15
Path 2: 1, 2, 3, 4, 5, 7, 8, 9, 10, 15
Path 3: 1, 2, 3, 4, 5, 7, 11, 12, 15
Path 4: 1, 2, 3, 4, 5, 7, 11, 13, 15

*Cyclomatic Complexity Held-Karp*

After creating the flowgraph and determining the independent path, then cyclomatic complexity can be searched [28]. Here is the process to find the cyclomatic complexity value of the method.

\[ VG = 4 \]
\[ VG = 17 \text{ edge} - 15 + 2 = 4 \]
\[ VG = 3 \text{ predicate node} + 1 = 4 \]

*Graph Matrix Held-Karp*

The search for cyclomatic complexity values can also be done by creating a graph matrix [25]. Here is the process of calculating the matrix graph of Iterative Deepening Search algorithm, as seen in Table 6.

**Table 5. Held-Karp Program Flow and Point List**

<table>
<thead>
<tr>
<th>a = 1</th>
<th>e = 5</th>
<th>c.2.2 = 9</th>
<th>e.2.4.2 = 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>b = 2</td>
<td>e.1 = 6</td>
<td>c.2.3 = 10</td>
<td>e.3 = 14</td>
</tr>
<tr>
<td>c = 3</td>
<td>e.2 = 7</td>
<td>e.2.4 = 11</td>
<td>f = 15</td>
</tr>
<tr>
<td>d = 4</td>
<td>e.2.1 = 8</td>
<td>e.2.4.1 = 12</td>
<td></td>
</tr>
</tbody>
</table>

![Figure 7. Flowgraph Held-Karp](image)

**Table 6. Graph Matrix Held-Karp**

\[
\begin{array}{cccccccccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\
\end{array}
\]

Row 1: 1 - 1 = 0
Row 2: 1 - 1 = 0
Row 3: 1 - 1 = 0
Row 4: 1 - 1 = 0
Row 5: 2 - 1 = 1
Row 6: 1 - 1 = 0
Row 7: 2 - 1 = 1
Row 8: 1 - 1 = 0
Row 9: 1 - 1 = 0
Row 10: 1 - 1 = 0
Row 11: 2 - 1 = 1
Row 12: 1 - 1 = 0
Row 13: 1 - 1 = 0
Row 14: 1 - 1 = 0
Row 15: 1 - 1 = 0
Row 16: 1 - 1 = 0
Row 17: 1 - 1 = 0
Row 18: 1 - 1 = 0
Row 19: 1 - 1 = 0
Row 20: 1 - 1 = 0
Row 21: 1 - 1 = 0
Row 22: 1 - 1 = 0

\[ 3 + 1 = 4 \]
(cyclomatic complexity)

**Comparison Result**

After all the evaluation process is done, then Iterative Deepening Search and Held-Karp methods are also evaluated based on the performance of programs that have been implemented.

Evaluation based on program performance is done by testing, Speed Process, CPU Usage and RAM or Memory Usage. Testing is done by running programs that have been made, with the same sample data input value as the sample data used at the time of implementation.

All data on speed, cpu and ram recorded starting from program run until get result. Here is the comparison result of the test that has been done as seen in Table 7.
Table 7. Comparison of Iterative Deepening Search and Held-Karp Evaluation

<table>
<thead>
<tr>
<th></th>
<th>IDS</th>
<th>HK</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Independent Path</strong></td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>C. Complexity (relevan)</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td><strong>Graph Matrix</strong></td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Open Program</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td>5.2 s</td>
<td>3.1 s</td>
</tr>
<tr>
<td>CPU Usage</td>
<td>15 %</td>
<td>25 %</td>
</tr>
<tr>
<td>Memory Usage</td>
<td>6 Mb</td>
<td>1 Mb</td>
</tr>
<tr>
<td><strong>Processing Program</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td>2.7 s</td>
<td>2.8 s</td>
</tr>
<tr>
<td>CPU Usage</td>
<td>58 %</td>
<td>65 %</td>
</tr>
<tr>
<td>Memory Usage</td>
<td>1 Mb</td>
<td>1 Mb</td>
</tr>
</tbody>
</table>

**CONCLUSION**

From the evaluation that has been done, it was found that Iterative Deepening Search and Held-Karp method, have different flowgraph images, as well as independent paths, each method has a different path, but when calculated about cyclomatic complexity and Graph Matrix obtained the same result between Iterative Deepening Search and Held-Karp method. Both methods produce the relevant cyclomatic complexity. Based on the performance test conducted obtained a fairly varied results, at the time the program starts to run Iterative Deepening Search method takes a longer time and larger memory. Iterative Deepening Search takes 5.2 seconds while Held-Karp method takes 3.1 seconds. If analyzed based on the program process of entering data and until the program gets results, the Held-Karp method takes a longer time than the Iterative Deepening Search method. The Iterative Deepening Search method takes 2.7 seconds while the Held-Karp method takes 2.8 seconds.

**ACKNOWLEDGMENT**

Thank to my lovely wife and daughter, my family and all of my friends who always give support from the start process of the research and until writing articles to publish. Also thank to my institution, STMIK STIKOM Bali, which always give motivation and funding support, thus make me always get spirit to learn on research, and make good publication.

**REFERENCES**


