IMPLEMENTATION OF ITERATIVE DEEPENING A* ALGORITHM AND PRUNING METHOD OF SOLUTION PUZZLE GAME FLOW FREE COLOR

Fahrurrozi

Informatic Engineering – Indonesian Computer University
Jln. Depati ukur 112-114 Bandung
E-mail : rozymom86@gmail.com

ABSTRACT

Flow Free Color is a puzzle-themed games popular enough it can be played on your computer or mobile. The game consists of a n x n matrix contains some colored circles spread over several grid. The game flow free color path is considered invalid if it doesn't meet with his partner, it takes a path search algorithm solution to connect the dots of color to connect in accordance with the type of color.

As for the algorithms that can be used such as brute force algorithms, greedy and others. In previous research using a brute force algorithm, but has a weakness because the execution time required longer. There is one algorithm to solve this problem that is Iterative Deepening A * algorithm.

However, in this research will be used in the search algorithm, IDA * line solution on game flow free color and method of pruning will reduce the number of nodes that are evaluated by a search tree.

Keyword : Iterative Deepening A*, algorithm A*, method pruning, flow free color

1. INTRODUCTION

Flow Free is a game type puzzles, this game consists of an n x n matrix containing several pairs of colored circles spread over several grid [2]. In this game the player must connect the dots color according to color. The problem in this game is how to connect the dots of color scattered random to connect with the appropriate color. In the processing of connecting the dots of colors, it takes the search algorithm. The search algorithm is applied to connect the dots color is brute force algorithms and greedy [15].

Application of algorithm of IDA * on game flow free color i.e. to get the path of the solution for connecting each of the dots color according to the type of color. IDA * algorithms take advantage A * in cost calculation actual cost of coupled with estimates for the cost of the examination of every simpul-simpulnya, this makes the algorithm IDA *

complete and optimal. However algorithms of IDA * done in iterative, then likely IDA * evokes the node-node over and over again and takes a long time. To resolve this issue, will use an extra method of Pruning method. With the method of pruning is trying to overcome the weakness owned IDA * algorithm by ignoring the likely path that does not produce a solution, to find out those lines produce a solution or does not produce a solution that is by comparing the values of alpha and Beta in accordance with the rules of the method of pruning.

The existence of the above problems, the algorithm is Iterative Deepening A * (IDA *) and pruning are expected to complete the above game solutions

2. THE CONTENT OF RESEARCH

2.1 Game

Games are games that use the interaction with the user interface through the image generated by the other video. Video games generally provide a system of rewards, such as a score that is calculated based on the level of success achieved in a task that is in the game.

2.2 Genre Game

The game has a lot of type or genre. Each of these genres have different characteristics. The genre of the game [7] including the following:

a. Action Game
This game is known from action therein in the form of movement, attack, reactions and other movements.

b. Adventure Games
This type of demanding games players to complete missions given all the adventures in the game.

c. Education Games
Educational games provide an emphasis on elements of education and learning.

d. Puzzle Game
Puzzle game has a very simple goal to complete the puzzle.

e. role-playing Game
In this type of game has a vast world explored by the player, either for treasures, objects and status (leveling), while destroying monsters and objects that allow it.

f. Simulation Game
Simulation games or sims game is a form of describing real-world situations, either in the form of

simulated driving environment as well as in relation to the interaction between human beings.

g. Strategy Game

In this type of game it takes thought and planning is properly to be able to win the game and usually use an overhead vantage point so that the player can see the entire game.

2.3 The Search methods

Search methods are generally divided into two groups, namely, blind or uninformed search (search blind or not given information) and heuristic or informed search (search with a guide or armed with information). Each method has characteristics that distinguish each other each other with advantages and disadvantages of each, which can be measured performansinya with four criteria [4] follows:

a. Completeness, i.e. whether the search method ensures the discovery solution if a solution does exist.

b. Time complexity, i.e., how long will it take for the search process.

c. Space complexity, i.e., how much memory is required for the search process.

d. Optimality, namely whether the search method ensures the discovery the best solution if there are a few different solutions.

2.4 Iterative Deepening A* Algorithm

Iterative Deepening A * algorithm is a combination of algorithm IDS and A *. Iterative Deepening Search algorithm is a technique that combines the advantages of a search using Breadth First Search (BFS) in terms of complete and optimal advantage of search techniques as well as Depth First Search (DFS) in case Space Complexity [3]. While the Algorithm A * (A Star) is a graph search algorithm/tree looking for the line from a start point to a predetermined end point [4]. A * algorithm i.e. excess fees taken into account actual cost of coupled with cost estimates. While the advantages of the algorithm IDS combines the advantages of Breadth First Search (complete and optimal) and excess Depth First Search (requires less memory) [10]. IDS is a Blind Search algorithm that performs iterative search use the Search Depth Limited Search (DSL) begins with the restrictions limit 0 [5].

If the path of the solution has not been found, each iteration returns the value f-the new limit will be used as a limitation of the search for the next iteration. F-value limit itself obtained from combined estimated cost and actual costs as found in equation (1), the math is expressed as:

f (n) = g (n) + h (n) ... (1)

With:

n = knot

g (n) = distance from the original location to the location early to n

h (n) = the distance from the beginning to the node location n

f (n) = value evaluation

As for searching h (n) as found in equation (2) is performed in the following way:

$$h(n) = \sqrt{(x - x1)^2 + (y - y1)^2} \quad ...(2)$$

With:

h (n) = the minimum cost estimate of the destination node to node n

X = X position of the initial node

Y = Y position of the initial node

X 1 = X position of node location to the destination node

Y 1 = Y position of node location to the destination node

2.5 Pruning Method

Pruning is a method that overrides state that might not produce solutions [11].

Suppose under the tree represents the search space which must be searched to find the solution can be seen in Figure 1.

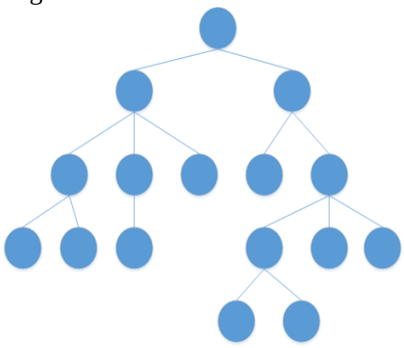


Figure 1 An Example Of A Tree Search Space

With pruning can be proven mathematically, can trim the search with no trace state of unnecessary search can be seen in Figure 2.

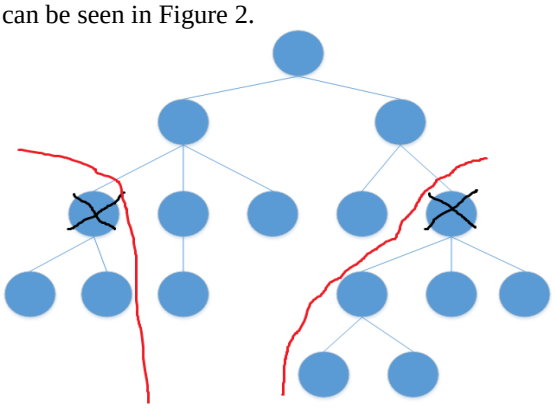


Figure 2 Example of pruning on the tree space search.

Pruning method of Karekteristik is:

a. the higher state or the smaller the depth of the state that we're pruning the more state searches we prune

b. With proper pruning, trimming gains from this can decrease the search time exponentially.

With this method, the search time can be reduced. One of the applications of the method of pruning that is famous for is the use of Alpha-Beta pruning on computer chess [5]. Alpha-beta pruning eliminate the

possibility of a step ahead of set of analysis results if they proved no better than have been found. The developer of the chess computer Deep Blue using this algorithm and generates innovative theory is referred to as "singular extensions" [5].

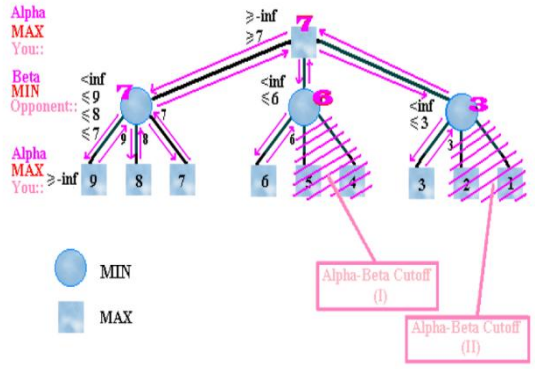


Figure 3 Examples Of Alpha-Beta Pruning Algorithm

The algorithm will either have two values: alpha and beta, which will represent the minimum value to the maximum value, and MAX to MIN. Initial state value of alpha is – infinity, and the initial value + infinity beta. While the alpha-beta Cutoff are alpha-beta nodes that do not need to be evaluated. During the course of the recursion, the difference between the alpha and beta will be progressively decreased. When the value of the beta became less than the alpha, it means that the current state is not the best result of both players, thus does not need to be explored further as there is in Figure 3.

Pruning method steps:

1. The first node with the value 9, up to the parent node level min enter the value 9, approch 8, because the value 9 8 and replace node < parent 8, approch 7, because the values 7 and 8 < 7 k ride level min.
2. After the third node first approached, up to max level, insert the value 7.
3. The fourth node on with a value of node 6, up to the parent node at level min enter the value 6. If the next node, then went in search of a higher value of 6, while at level max looking smaller than 7, then node 5
and 4 in pruning (not approximated).
4. After that seventh node value of node 3, up to the parent node at level min enter the value 3. If it does come up next, then the node will be looking for higher grades of 3, while at max level searching is smaller than 7, then node 2 and 1 in the pruning (not approximated). Based on the above steps, as for the flowchart of pruning is present on the



Figure 4 Flowchart Pruning

While the pseudocode for pruning as follows:

```
evaluate (node, alpha, beta)
    if node is a leaf
        return the heuristic value of node
    if node is a minimizing node
        for each child of node
            beta = min (beta, evaluate (child,
            if beta <= alpha
                return beta
        return beta
    if node is a maximizing node
        for each child of node
            alpha = max (alpha, evaluate (child
            if beta <= alpha
                return alpha
        return alpha
```

2.6 Game Analysis of flow free color

Game Flow Free Color is a puzzle-based game that has been developed by big duck games for mobile devices and ios, in this game there is a challenge, a challenge that was given on this game is very interesting with the challenged various difficulty levels shown in the various grid game i.e. ranging from 5 x 5 up to the highest levels of 14 x 14.

Rules of the game flow free color as follows:

1. It is assumed there is a 4 x 4 grid that contains 6 color point consisting of two of the same color at any point color
2.6 point color is made up of 2 red, 2 yellow and 2 green color
3. Color – this color will be scattered in the
4 x 4 grid and color – this color must be connected in accordance with the same color
4. In linking the colors we have to form a path and browse the grid so that the colors will be connected can be connected.
5. In tracing the grid are not allowed to cut the path that has been formed, and if there is a grid that is empty or is not passed then the path is considered invalid.

2.7 Analysis of Data Input

Input Data analysis conducted in this study is analyzing the input algorithm in IDA * on game flow free color. The data input parameters that are inputed by the user i.e. the dots color is represented with figures of 0-6 who are composed of an empty color, green, red, blue, yellow, orange, purple. The color dots done in a random grid of laptop in the deployment. As for the stages to do random as follows:

The number of grid-number of colors x 2

↓

The number of grid – 1

As for the random pseudocode is as follows:

```
function genColor (seed) {
   color = Math.floor((Math.abs(Math.sin(seed) *
16777215)) % 16777215);
   color = color.toString(16);
   // pad any colors shorter than 6 characters with
leading 0s
   while(color.length < 6) {
      color = '0' + color;
   }
   return color;
}
for (chr in chars) {
   if (chars.hasOwnProperty(chr)) {
      color = genColor(chars[chr].charCodeAt(0));
      colorloop.append('<div style="background:#'
+ color + '">' + chars[chr] + ' = #' + color +
'</div>');
      ccount[color] = ccount[color] === undefined
? 1 : ccount[color] + 1;
   }
}
for(itm in ccount) {
   if (ccount.hasOwnProperty(itm)) {
      if (ccount[itm] > 1) {
         repeatCount += 1;
      }
   }
}
```

2.8 Algorithm Analysis Of IDA*

Algorithm of IDA * not considering the current shortest path and the cost node-node you've visited, but only given one single path at a time. In the end algorithm IDA * will find lines with minimum cost to the destination node because it is only updating the cache (save in memory) the nodes that are in the search path. Therefore the amount of memory that is needed to become a lot less.
The following is a flowchart steps search (IDA *),

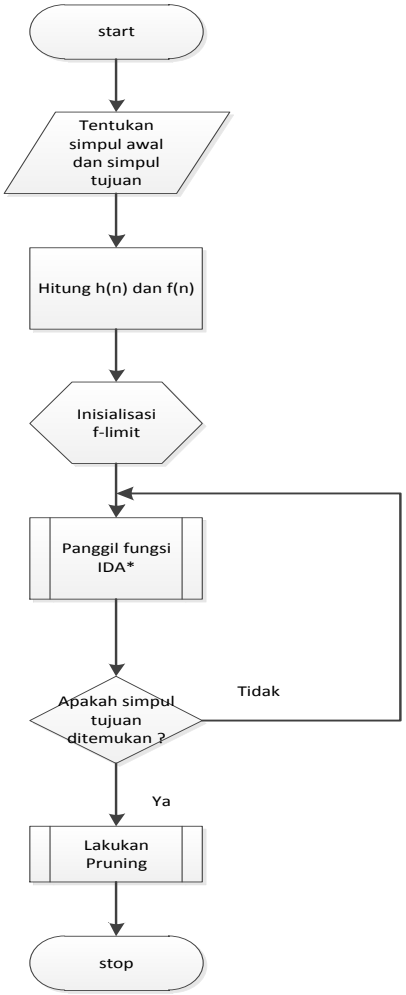


Figure 5 Flowchart IDA* Algorithm

While the pseudo-code algorithm of IDA * is as follows:

```
function  depth_first_search  (n,
limit)
    If f(n) > limit
    then return f(n)
    if h(n) = 0 then BERHASIL
    return  nilai  terendah  dari
depth_first_search    (ni;,limit)
bagi semua successor n; dari n
end
prosedure IDA* (n)
    limit ≥ h(n)
```

```
repeat
    limit = depth_first_search
(n,limit)
until BERHASIL
end
```

2.9 Application Of Heuristic Iterative Deepening A*

The game has a 4 x 4 grid there are 6 point color pattern consists of 2 same color i.e. 2 red, 2 yellow, 2 green color. Where each letter is represented by a color:
A = red color
B = yellow
C = Green

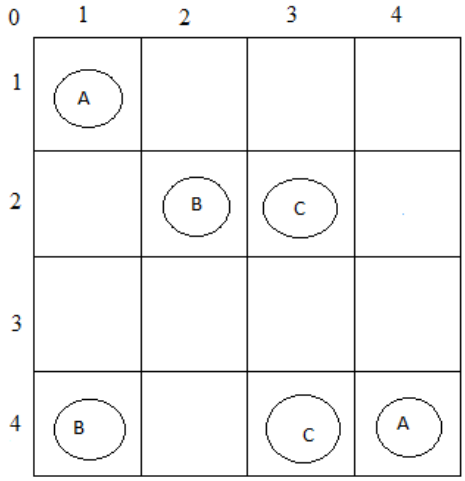


Figure 6 Case flow free color

The game flow free color, there are some patterns of color in each grid-grid. Color patterns are contained in the following grid
:

Table 1 Table color patterns

The pattern of the game point color flow free color			
No	The Name Of The Pattern	The Initial Coordinates	The Final Coordinates
1.	Pattern color A	1,1	4,4
2.	Pattern color B	2,2	1,4
3.	Pattern color B	3,2	3,4

2.10 Search Pattern Color A

Table 2 Search Pattern Color A

Search Pattern Color A		
Knot The initial	Displacement of the end	Iteration
1,1	4,4	1
1,1	2,1	2
2,1	3,1	3
3,1	4,1	4
4,1	4,2	5
4,2	4,3	6
4,3	4,4	7
1,1	4,4	8
1,1	1,2	9
1,2	1,3	10
1,3	2,3	11
2,3	3,3	12
3,3	4,3	13
4,3	4,4	14
1,1	4,4	15

After the analysis is performed by using the algorithm of IDA * value evaluations, results obtained with regard to the value of the evaluation are as follows:

Table 3 The value of color A pattern evaluation

Pattern color A		
Knot The initial	Displacement of the end	Value evaluation
(1,1)	(2,1)	2
(2,1)	(3,1)	2
(3,1)	(4,1)	2
(4,1)	(4,2)	2
(4,2)	(4,3)	2
(4,3)	(4,4)	2
(1,1)	(1,2)	1
(1,2)	(1,3)	2
(1,3)	(2,3)	2

(2,3)	(3,3)	2
(3,3)	(4,3)	2

After A pattern of calculated coordinates will then be done searching algorithm steps IDA * in A pattern that can produce solutions and measures that do not produce a solution by using the following method of pruning pruning method process :

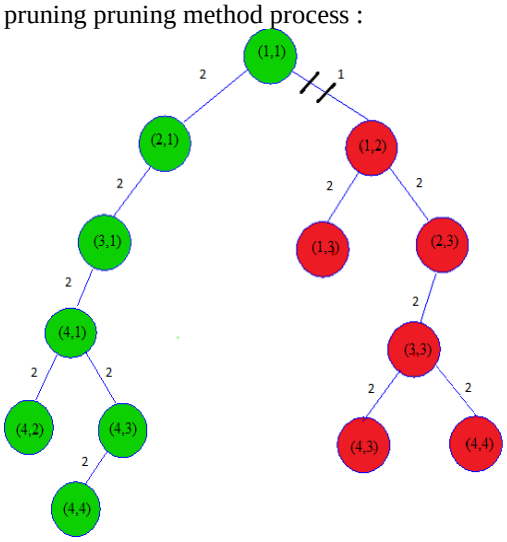


Figure 7 The set of possibilities pattern A trajectory line

The method of pruning will register two values namely alpha and beta, alpha values are initial state – infinity, and the initial value + infinity beta. When the value of the beta became less than the alpha, it means the State is not a result of a solution and there is no need to do further searches. So the conclusion in Figure 7 is the green line is a line of solutions in the formation lap for the color patterns with due compliance with the analysis method of pruning and the rules of the game. While the red line represents the path berwana not produce solutions because it considered cutting other colors patterns and lines also carried out an analysis of the method of pruning with generate a value $\beta < \alpha$.

2.11 Search Pattern Color B

Table 4 Search Pattern Color B

Pattern color B		
Knot The initial	Displacement of the end	Iteration
2,2	1,4	1
2,2	1,2	2
1,2	1,3	3
1,3	1,4	4
2,2	1,4	5
2,2	2,3	6
2,3	2,4	7
2,4	1,4	8
2,2	1,4	9

After the analysis is performed by using the algorithm of IDA * value evaluations, results obtained with regard to the value of the evaluation are as follows:

Table 5 The value of color B pattern evaluation

Pattern Color B		
The initial coordinates	The final Coordinate	Value evaluation
(2,2)	(1,2)	2
(1,2)	(1,3)	2
(1,3)	(1,4)	2
(2,2)	(2,3)	1
(2,3)	(2,4)	2
(2,4)	(1,4)	2

After the coordinates Pattern B counted the next search will be done step-by-step algorithm IDA * in the pattern B which can produce solutions and measures that do not produce a solution by using the method of pruning pruning methods process the following:

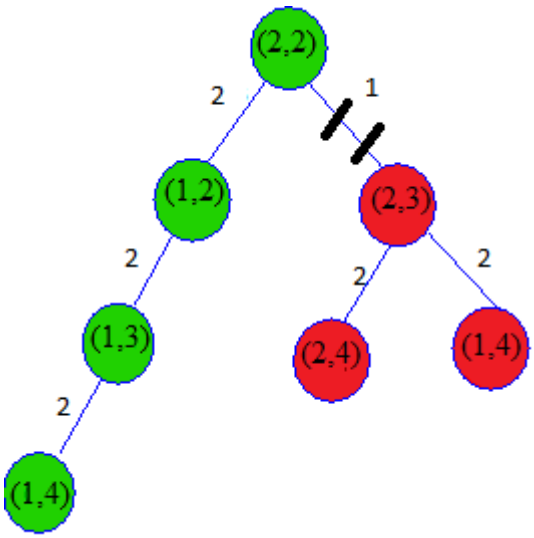


Figure 8 The set of possibilities of line trajectory pattern B

The method of pruning will register two values namely alpha and beta, alpha values are initial state – infinity, and the initial value + infinity beta. When the value of the beta became less than the alpha, it means the State is not a result of a solution and there is no need to do further searches. So the conclusion in Figure 8. is the green line is a line of solutions in the formation lap for the color pattern B due in accordance with the analysis method of pruning and the rules of the game. While the red line represents the path berwana not produce solutions because it considered cutting other colors patterns and lines also carried out an analysis of the method of pruning with generate β value < α

2.12 Search Pattern Color C

Table 6 Search Pattern Color C

Pattern color C		
Knot The initial	Displacement of the end	Iteration
3,2	3,4	1
3,2	3,3	2
3,3	2,3	3
2,3	2,4	4
2,4	3,4	5
3,2	3,4	6

After the analysis is performed by using the algorithm of IDA * value evaluations, results obtained with regard to the value of evaluation as follows:

Table 7 The value of color A pattern evaluation

Pattern Color C		
The initial coordinates	The final Coordinate	Value evaluation
(3,2)	(3,3)	2
(3,3)	(2,3)	2
(2,3)	(2,4)	2
(2,4)	(3,4)	2

After the coordinates Pattern C calculated would henceforth be conducted a search algorithm steps IDA * in the pattern C can produce solutions and measures that do not produce a solution by using the method of pruning pruning methods process the following:

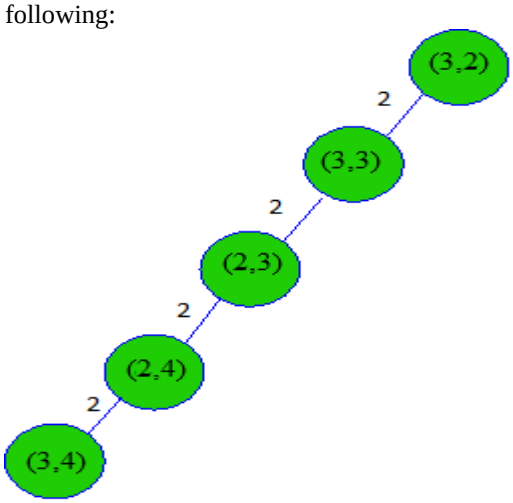


Figure 9 The set of possibilities of line trajectory pattern B

Line C pattern is a path that produces solutions and in accordance with the rules of the game.

2.13 Non Functional Requirments Analysis

Non-functional analysis is the analysis required to determine system requirements specification. This specification also consists of elements or components needed by the system to be built up with the system implemented. This needs analysis also determines the required input specification system, output to be generated and system process that is needed for processing the input

to produce a desired output. As for the requirement that is used in this implementation is as follows:
1. Hardware Requirements
2. Software Requirements
Needs analysis of non functional aims so that implementation can be used in accordance with needs.

2.14 Needs Hardware Analysis

Iterative deepening algorithm to implement A *, required hardware specifications that support the work processes of a system. As for the hardware specifications needed in the application development game flow free color as follows:

Table 8 Hardware specifications

No	Hardware	Specifications
1	CPU	Intel Dual core 1.80 GHz
2	VGA	Intel(R) HD Graphics
3	Harddisk	80 GB
4	RAM	2 GB

2.15 A functional Needs Analysis

Modeling used in the game to be made is the analysis of the design and development of object-based software, where the tools will be used for software development that is modeling the Unified Modeling Language (UML). Modeling in the analysis include the making of use case diagrams, definitions, definitions of use case use case, scenario, activity diagrams, sequence diagrams, and class diagrams..

A. Use Case Diagram

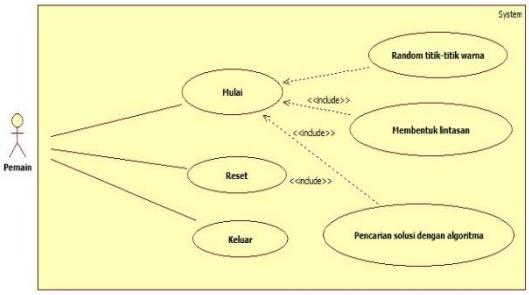


Figure 10 Use Case Diagram

B. Class Diagram

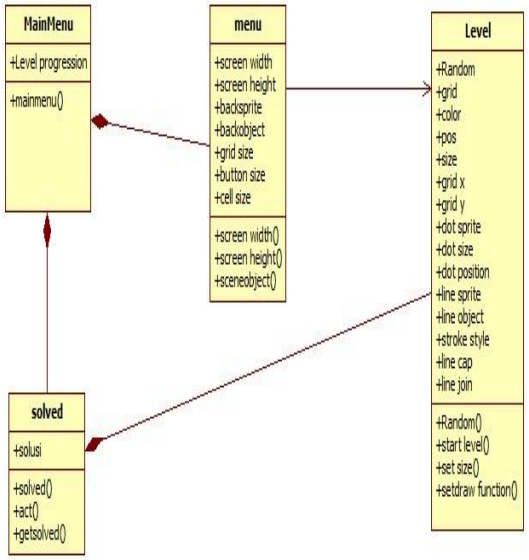


Figure 11 Class Diagram

2.16 Implementation and Testing

Stages of implementation are made after the analysis and design of the system was completed, and the next will be implemented into programming languages and tested into the application. An application that has been built will be implemented in order to know the system can run as appropriate or not. The objective of the implementation is to implement a design that has been done to the system, which consists of a description of the implementation environment, be it hardware environment, as well as the environment of the software, as well as the implementation of the program.

2.17 Hardware Implementation

Hardware specifications (hardware) used in iterative deepening algorithm implementation of A * to specify the path of the solution and pruning methods on game flow free color is as follows:

Table 9 Hardware Specifications

Hardware	Spesifications Hardware
Processor	Intel(R) Celeron (R) @ 2.16 GHz
Resolusi Layar	1366 x 768 pixel
Memori	Memori 2 GB
Hardisk	Hardisk 320 GB
Keyboard dan touchpad	Standar

2.18 Software Implementation

Specifications (software) software that is used in the implementation of the iterative deepening A *

algorithms and methods on game flow free color as follows:

Table 10 Software Specifications

Software	Software Specifications
Sistem Operasi	Windows 7
Tools Program	JetBrains Webstrom

2.19 Implementation Class

The following is the implementation of the entity class in the .js file flow as follows:

Table 11 Implementation Class Entity

Class	Stereo Type
Menu.js	Entity
Level.js	Entity

2.20 Implementation Testing

At this stage of testing against the game flow free color done with two different kinds of techniques, namely Black-box testing and testing methods. Black box testing focuses on the errors contained in the program, while the testing methods focus on testing the implementation of the method in the simulation.



Figure 12 The Main Menu Interface

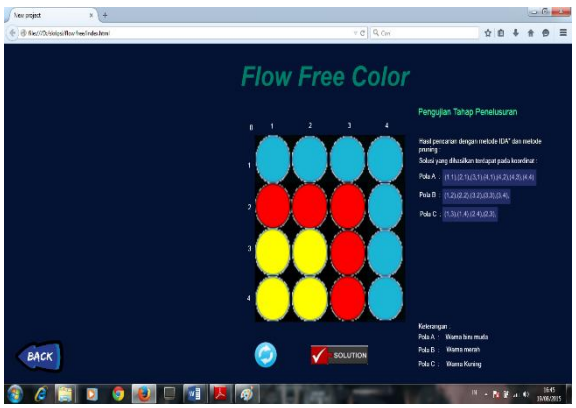


Figure 13 Testing Interface

The resulting solution is based on A pattern is at the coordinate (1,1), (2.1), (3.1), (4.1), (4.2), (4.3), (4.4), on the pattern of B is at coordinates (1, 2), (2.2), (3.2), (3,3), (3,4), sedngkan for pattern C is at coordinates (1.3), (1.4), (1.5), (2,3).

3. CONCLUSION

Based on the results obtained in the research and drafting of this thesis as well as adapted to the purpose, then it can be concluded that:

1. for the implementation of the algorithm of iterative deepening A * on game flow free color can be used with the function search heuristic formula as the value of the path, but the solution path search using search tree depth first search, it can reproduce the search node-node which is contained in every pattern search path, but in this case added a method i.e. pruning, this method is used to ignore lines which do not produce a solution.
2. iterative deepening A * Algorithm can do a search the dots color according to the type of color with the spread of color dots are random, and the level of accuracy that is right on the color dots distributed random color dots so that it can be connected in accordance with the type of color.

Based on the results of this research, then be given advice to be given in future development i.e.

1. You can add an algorithm to do random dots of color that have the number of a grid search algorithm so that many IDA * pruning methods can run well, but for the addition algorithm has to be payed in advance because the algorithm IDA * in this case is to find the path of the solution.
2. In the event that testing should be added the search feature points pertahap color, because on this system do all the search process the dots color simultaneously. And also display the calculation process pertahap.
3. for more accuracy in the calculation of the points of color developer can add a method to calculate the accuracy of color dots.

REFERENCE

[1] L. Hermawan dan R. Kristoforus Jawa Bendi, "Penerapan Algoritma A* pada Aplikasi Puzzle," dalam Seminar Nasional Teknologi Informasi dan Komunikasi, Palembang, 2013.
[2] Mike, "Flow Free Color," Big Duck Games, 13 07 2012. [Online]. Available: http://blog.bigduckgames.com. [Diakses 12 02 2015].
[3] S. Kusumadewi, Artificial Intelligence, Yogyakarta: Graha Ilmu, 2003.
[4] Suyanto, Artificial Intelligence, Bandung: Informatika, 2011.
[5] J. S. Russell dan P. Norvig, Artificial Intelligence – A Modern Approach 3rd Edition., New Jersey: Prentice hall, 2010.
[6] R. S. Pressman, Rekayasa Perangkat Lunak Pendekatan Praktis, Yogyakarta: Andi, 2012.
[7] A. Ismail, Education Game, Yogyakarta: Pro-u Media, 2009.
[8] A. Rollings dan D. Morris, Game Architecture and Design, New Rider: A New Edition, 2004.
[9] A. Nilwan, Pemrograman Animasi dan Game Profesional, Jakarta: Elex Media Komputindo, 2010.
[10] W. Budiharto dan D. Suhartono, Artificial Intelligence Konsep dan Penerapannya, Yogyakarta: Andi, 2014.
[11] S. J. Russell dan P. Norvig, Artificial Intelligence – A Modern Approach 2rd Edition., New Jersey: Prentice hall, 2010.
[12] B. Hariyanto, Struktur Data : Pondasi Membuat Program Elegan dan Efisien, Bandung: Informatika, 2008.
[13] S. Rosa dan M. Shalahuddin, Rekayasa Perangkat Lunak, Bandung: Informatika, 2013.
[14] F. S. J. Hill, Computer Graphics Using OpenGL (2nd ed.), New Jersey: Prentice hall, 2001.
[15] M. Rahmat, "Pencarian Solusi Permainan Flow Free Menggunakan Brute Force dan Pruning," Jurnal Teknik Informatika, 2013.