PERBANDINGAN ALGORITMA *A**, *DIJKSTRA* DAN *FLOYD WARSHALL* UNTUK MENENTUKAN JALUR TERPENDEK PADA PERMAINAN *"BACTERIA DEFENSE"*

M. Azan Cahyadi[1], M. Arif Bambang. P[2], Wijang Widhiarso[3], Yohannes[4]
STMIK GI MDP; JL. Rajawali No.14 Palembang, 0711-376400
Jurusan Teknik Informatika, STMIK GI MDP, Palembang
e-mail: [1]zancahyadi@mhs.mdp.ac.id, [2]arif_purnama@mhs.mdp.ac.id, , [3]wijang.mdp.ac.id
[4]yohannesmasterous@mdp.ac.id

Abstrak

Algoritma Pathfinding atau pencarian jalur tercepat merupakan salah satu varian algoritma yang sering diterapkan pada permainan. Penelitian ini menggunakan 3 jenis algoritma pathfinding yaiu algoritma A Star, Dijkstra dan Floyd Warshall yang nantinya akan dilakukan perbandingan untuk mendapatkan jalur terpendek pada permainan Bacteria Defense. Bacteria Defense adalah permainan yang memiliki genre Turn Base Strategy yaitu sebuah permainan yang dimainkan dua orang pemain setiap player memiliki beberapa bidak dan memiliki 100 titik untuk meletakkan bidaknya secara bergantian. Algoritma A Star, Dijkstra dan Floyd Warshall digunakan untuk menentukan jalur terpendek berdasarkan bobot yang dimiliki setiap titik. Tujuan dari penelitian ini adalah untuk menerapkan dan menganalisis algoritma A Star, Dijkstra dan Floyd Warshall untuk menentukan jalur terpendek pada permainan Bacteria Defense. Penelitian ini dilakukan untuk mengetahui algoritma mana yang bekerja lebih efektif diantara Algoritma A Star, Dijkstra dan Floyd Warshall dengan parameter perbandingan Weight, Path Count, dan Checked Node. Berdasarkan hasil pengujian menunjukkan bahwa algoritma A Star lebih efektif dari algoritma Dijkstra dan Floyd Warshall karena Algoritma A Star melakukan pengecekan Node lebih sedikit dengan 2499 pengecekan sedangkan algoritma Dijkstra melakukan 4717 pengecekkan node dan Floyd Warshall melakukan 4800 pengecekan node .

Kata kunci : *Turn Base Strategy*, Algoritma *A Star*, Algoritma *Dijkstra*, Algoritma *Floyd Warshall*

Abstract

Pathfinding Algorithm or shortest path finder is one of algorithm variant that usually use in game. Pathfinding algorithm used in "Bacteria Defense" game is AStar, Dijkstra and Floyd Warshall. Bacteria Defense is Turn Base Strategy game. This game played by 2 player, every player take turns to control a few of unit and 100 nodes to put unit on the node. AStar, Dijkstra, and Floyd Warshall used to decide shortest path based on node weight. The goal of this research is to implement and analysis on Astar, Dijkstra And Floyd Warshall Algorithm in Bacteria Defense game. This research used to find out which algorithm that work better between AStar, Dijkstra and Floyd Warshall with weight, Path count, and checked node comparasion parameter. Based on the testing result Astar Algorithm have checked node less than from Dijkstra and Floyd Warshall Algorithm but the algorithms have the same path count and weight.

Keywords : *Turn Base Strategy*, *A Star* Algorithm, *Dijkstra* Algorithm, *Floyd Warshall* Algorithm

1. PENDAHULUAN

Implementasi kecerdasan buatan banyak diterapkan untuk membantu menyelesaikan tugas- tugas manusia di bidang kesehatan, industri, penerbangan, militer, permainan/game dan bidang-bidang lain. Beberapa teknik kecerdasan buatan yang pada *game* digunakan untuk memodelkan komputer sehingga dapat bertindak seperti manusia seperti *Finite State Machines*, Logika *Fuzzy*, *Neural Networks*, *path finding,* dan lain sebagainya. Teknik *path finding* atau gan pencarian lintasan adalah teknik yang digunakan di dalam sebuah game untuk menentukan lintasan terpendek antara titik awal dan titik akhir tanpa melewati hambatan.

Dalam teknik *path finding* ada beberapa pilihan algoritma yaitu antara lain *A Star*, *Dijkstra* dan *Floyd Warshall*. *A Star* adalah algoritma menggunakan sifat *greedy* dan fungsi heuristik untuk mendapatkan rutenya[1]. Algoritma *Dijkstra* adalah sebuah algoritma greedy (*greedy algorithm*) dalam memecahkan permasalahan jarak terpendek (*shortest path problem*) untuk sebuah graf berarah[2]. Algoritma *Floyd-Warshall* memiliki input graf berarah dan berbobot (*V*,*E*), yang berupa daftar titik (*node*/*vertex V*) dan daftar sisi (*edge E*).[3]

Pada pada paper ini digunakan tiga (3) algoritma untuk teknik *path finding* yaitu *A Star, Dijkstra* dan *Floyd Warshalls*. Rio, dkk (2016) telah menggunakan Algoritma *A Star dan Fuzzy Logic Sugeno* untuk mengembangkan game yang diberi nama "Game Battle City". Berdasarkan hasil uji (menggunakan 30 data uji) terhadap algoritma yang digunakan diperoleh hasil bahwa *fuzzy logic sugeno* mendapatkan 100% keberhasilan dalam menentukan perilaku NPC dan algoritma *A Star* didapat keberhasilan 100% dalam mencari jalur terpendek [4].

Algoritma *Dijkstra* juga diterapkan pengembangan untuk *Game* "Hijaiyah" oleh Zaidah (2016), hasilnya diperoleh bahwa keberhasilan 100% pada uji coba sistem [5].

Untuk mengambangkan game "*Maze Treasure"* Yusuf, dkk (2016) menggunakan Algoritma *Fuzzy State Machine* dan *Floyd Warshall*. Hasil penelitiannya menunjukan bahwa dari 20 data uji metode *Finite State Machine* mendapatkan keberhasilan 100% dalam menentukan perilaku NPC dan *floyd warshall* mendapatkan keberhasilan 100% dalam mencari jalur terpendek[6].

Berdasarkan penelitian di atas dapat dapat diketahui bahwa algoritma *A Star*, *Dijkstra* dan *Floyd Warshall* dapat digunakan secara maksimal untuk pengembangan game yang berbasis pencarian jalur tercepat. Tantangannya adalah menentukan algoritma mana yang terbaik diantara ketiganya jika digunakan untuk mengembangkan game atau permainan berbasis pencarian tercepat. Atas dasar tersebut, paper ini akan memamparkan perbandingan algoritma *A Star*, *Dijkstra* dan *Floyd Warshall* untuk menentukan jalur tercepat menggunakan dengan melukakan pengujian saat mengembangkan *game* yang diberi rnama "*Bacteria Defense*". Perbandingan algoritma untuk menentukan jalur terpendek dan mendapatkan algoritma terbaik berdasarkan besar bobot yang diperlukan dan banyak *turn* ke tujuan.

2. METODE PENELITIAN

Adapun tahapan-tahapan yang dilakukan di dalam penelitian ini, meliputi :

2.1 Identifikasi masalah

Pada tahap ini dilakukan identifikasi masalah penelitian mengenai perbandingan tiga algoritma dengan menggunakan *game* sebagai media pengujian.

2.2 Studi Pustaka

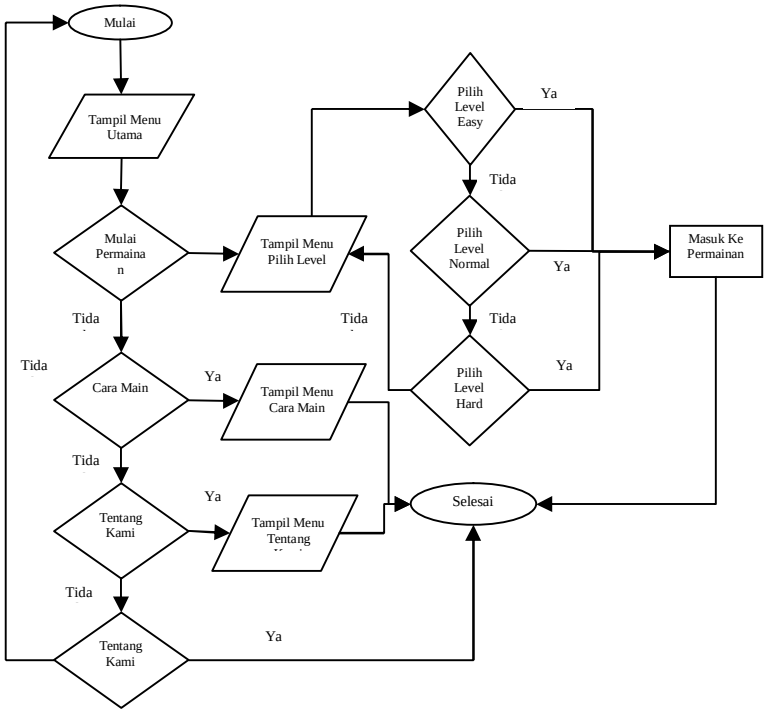
Pada tahap ini, dikumpulkan bahan dan buku-buku yang berhubungan dengan algoritma *A*, Dijkstra* dan *Floyd Warshall,* kemudian mempelajari buku tentang pengembangan *game* yang digunakan sebagai media pengujiannya. Tahapan ini bermanfaat untuk membantu penulis dalam membuat landasan teori dan menambah pemahaman algoritma *A*, Dijkstra* dan *Floyd*.

2.3 Studi Literatur

Kata *Game* berasal dari kata bahasa Inggris, Dalam kamus bahasa Indonesia istilah "Game" adalah permainan. Permainan merupakan media atau virtual digital yang didalamnya terdapat sekumpulan interaksi antar pemain dan memiliki peraturan yang dibuat oleh permainan tersebut. Permainan dalam hal ini merujuk pada pengertian kelincahan intelektual (*Intellectual Playable Game)* yang juga dapat diartikan sebagai keputusan dan aksi pemainnya. Untuk mengembangaka sebuah game berbasis pencarian dibutuhkan algortima untuk mengembangkannya. Salah satu algoritma yang sering digunakan adalah Algoritma Dijkstra dinamakan dengan nama penemunya yaitu seorang ilmuwan komputer yang bernama Edsger Dijkstra. Dijkstra adalah algoritma yang digunakan untuk mencari lintasan terpendek pada sebuah graf berarah. Ian Millington (2006) pada bukunya yang berjudul *Artificial Inteligence for Games* menyebut algoritma Dijkstra sebagai versi ringkas dari algoritma A Star. Selain algoritma Dijkstra, Algoritma A Star juga digunakan sebagai salah satu algoritma pencarian terbaik dalam mencari jalur terpendek dengan melakukan perhitungan pada jalur *node* awal menuju *node* akhir. Algoritma ini pertama kali dijelaskan pada tahun 1968 oleh Peter Hart, Nils Nilsson dan Bertram Raphael. Selain dua lagoritma diatas Algoritma Floyd-Warshall juga digunakan. Algoritma Floyd-Warshall adalah salah satu varian dari pemrograman dinamis yang melakukan pemecahan masalah dengan memandang solusi yang akan diperoleh sebagai suatu keputusan yang saling terkait. Artinya solusi-solusi tersebut dibentuk dari solusi yang berasal dari tahap sebelumnya dan ada kemungkinan solusi lebih dari satu.

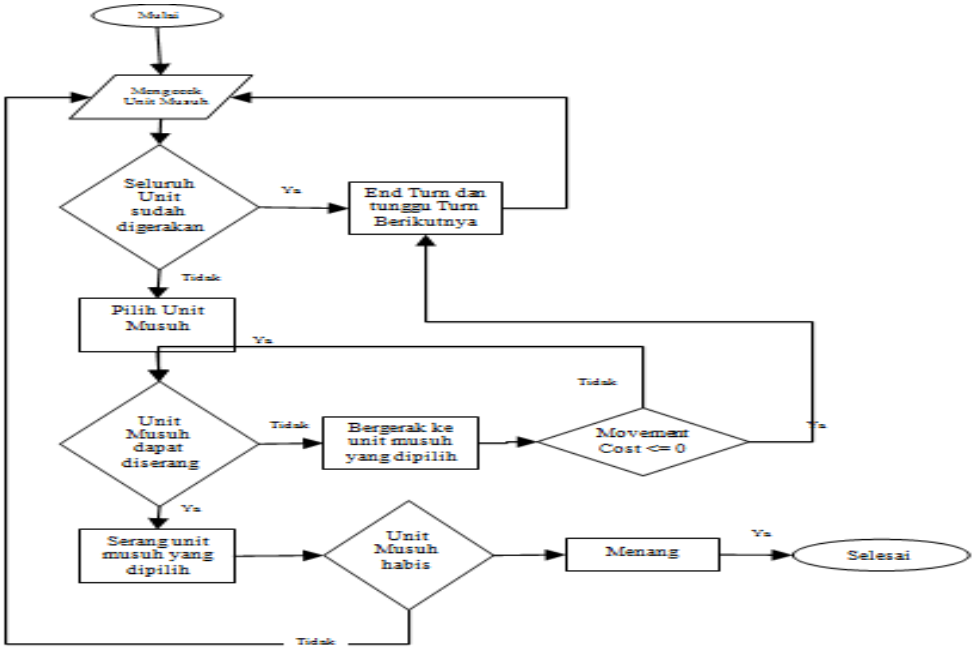
2.4 Implementasi Algoritma

Implementasi berarti suatu tindakan atau pelaksanaan rencana yang telah disusun dengan cermat dan rinci. Implementasi ini biasanya selesai setelah dianggap permanen. Sedangkan Algoritma dapat dikatakan sebagai urutan langkah-langkah logis yang sistematis dalam mencari suatu solusi dari suatu permasalahan yang ada. Pada program komputer, algoritma terdiri dari sekumpulan langkah-langkah untuk mencapai suatu tujuan, seperti logika *if-then-else* maupun pengulangan suatu tindakan atau langkah dengan *loop*. Dalam paper ini untuk menggambarkan bagaimana langkah dan skenario Impelemnatasi algrotima Pada tahap ini, akan di mulai proses perbandingan algoritma A*, Dijkstra dan Floyd Warshall dibuat dalam bentuk diagram alur serta diimplemntasi dalam bentuk perancangan output dilayar komputer. Gambar 2.1 menjelaskan alur dari menu utama permainan 'bacteria defence' yang memilik tombol mulai permainan, tombol cara main, tombol tentang kami, dan tombol keluar.



Gambar 2.1 Diagram Alir Menu Utama pada Bacteria Defense

Gambar 2.2 merupakan diagram alir permainan yang menjelaskan prosedur permainan. Langkah pertama yang dilakukan adalah dengan memeriksa unit yang dapat diserang. Terdapat beberapa unit yang pemain miliki, jika pemain masih memiliki unit yang dapat digerakkan maka unit tersebut bisa menyerang tergantung dari apakah unit musuh ada di jangkauan serang unit pemain.



Gambar 2.2 Diagram Alir Permainan

Perancangan Permainan

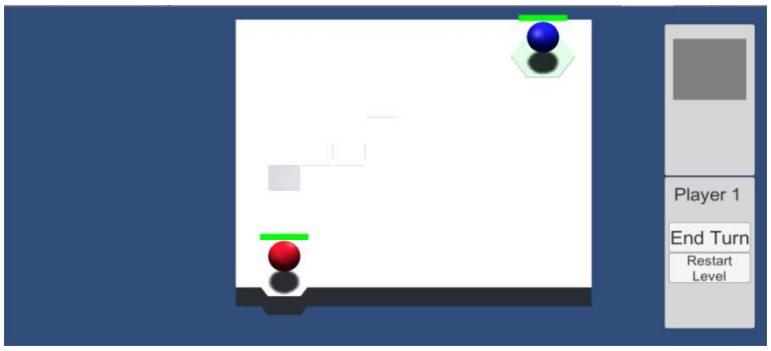
a. Tampilan Menu Utama Permainan

Pada gambar 2.3 merupakan tampilan saat penegguna menjalankan aplikasi. Dari Tampilan halaman pembuka ini menampilkan 4 tombol yaitu tombol start game, tombol cara bermain, tombol tentang pembuat, dan tombol keluar.

Gambar 2.3 Tampilan Awal Game

b. Tampilan Game

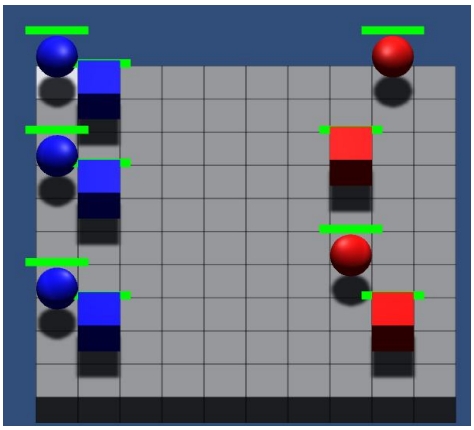
Gambar 2.4 akan muncul pada saat player memilih tombol Mulai Permainan. Player akan mendapatkan misi untuk mengalahkan semua Enemy atau NPC. Dalam permainan *player* bisa memilih grid yang ingin di lewati untuk tiap unit yang dimiliki, memilih musuh untuk diserang.



Gambar 2.4 Tampilan dalam Permainan

c. Tampilan game Level Easy

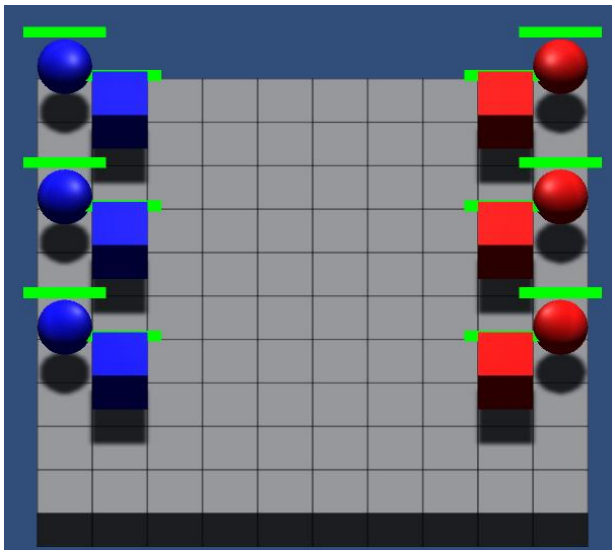
Gambar 2.5 merupakan tampilan pada *level Easy*. Dimana pada level ini terdapat 6 unit yang bisa di gerakkan oleh *player* dan 4 unit NPC yang akan menyerang unit player.



Gambar 2.5 Tampilan Game pada Level Easy

d. Tampilan game Level Normal

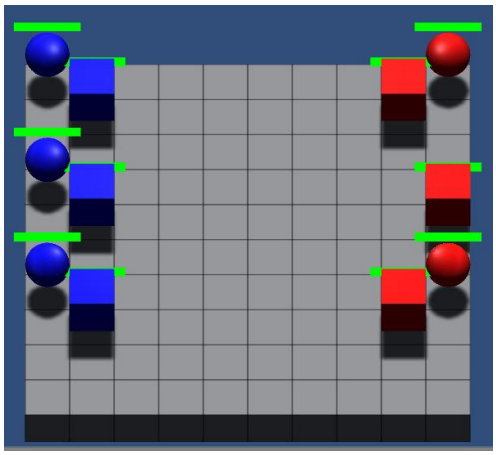
Gambar 2.6 merupakan tampilan pada level Normal. Dimana pada level ini terdapat 6 unit yang bisa di gerakkan oleh player dan 6 unit NPC yang akan menyerang unit player.



Gambar 2.6 Tampilan Game pada Level Normal

e. Tampilan game Level Hard

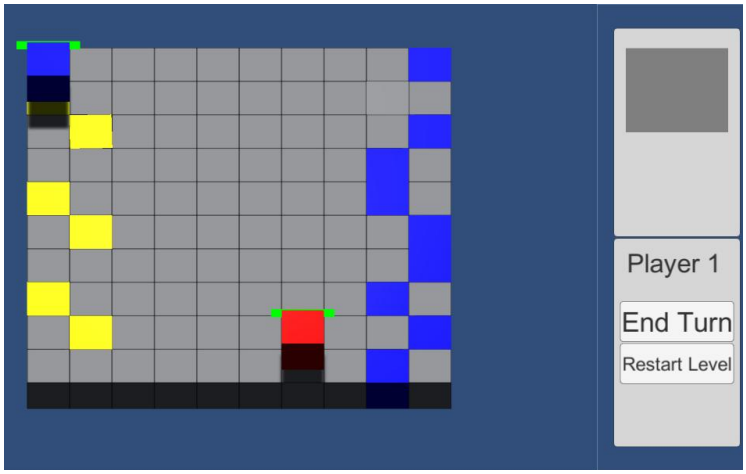
Gambar 2.7 merupakan tampilan pada level hard. Dimana pada level ini terdapat 6 unit yang bisa di gerakkan oleh player lalu 3 unit enemy dan 1 unit bos enemy yang akan menyerang unit player.



Gambar 2.7 Tampilan Game pada Level Hard

2.5 Hasil Uji

Pada tahap ini dilakukan pengujian dengan cara tiap titik awal akan dipilih titik akhir yang berbeda kemudian dari titik akhir tersebut akan didapatkan nilai path count, weight dan checked node. Terdapat 6 titik awal yang akan diuji kemudian setiap titik awal akan dilakukan 10 kali pengulangan dengan titik akhir yang berbeda. pengujian ini dilakukan secara bergantian menggunakan algoritma A Star, Dijkstra, dan Floyd Warshall dengan titik yang sama kemudian dilakukan perbandingan ketiga algoritma tersebut (gambar 2.8).



Gambar 2.8 Tampilan awal pengujian

3. HASIL DAN PEMBAHASAN

3.1 Perbandingan Algoritma

Ada 4 parameter yang dilakukan di dalam pengujian algoritma yaitu *Path Count*, *Weight, Checked Node* dan *Play Turn. Path Count* merupakan berapa banyak node yang di lewati oleh *NPC*, *Weight* adalah bobot dari *Node* yang dilewati oleh *NPC*, *Checked Node* adalah banyak *Node* yang diperiksa oleh algoritma tersebut dan *Play Turn* adalah berapa banyak giliran bermain untuk menuju player. Setelah melakukan pengujian pada algoritma

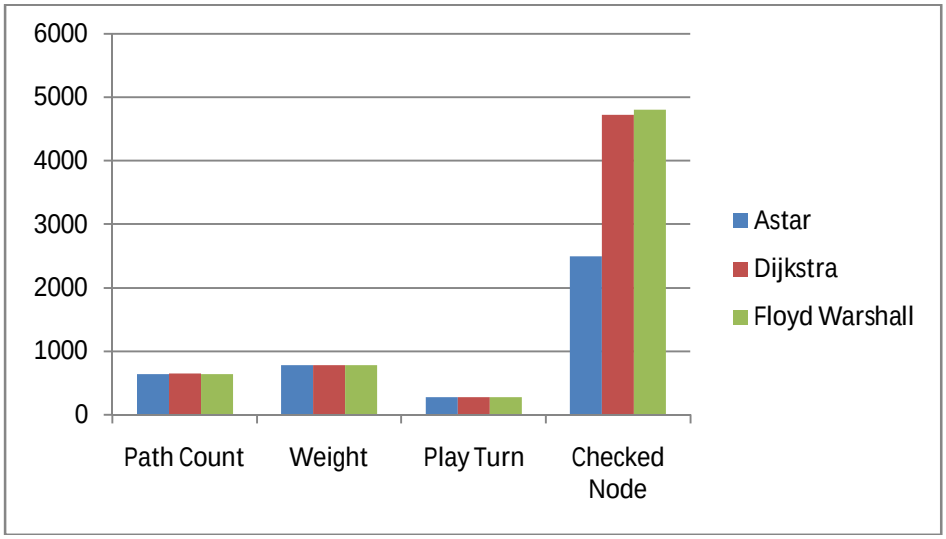
A Star, Dijkstra dan *Floyd Warshall* maka dilakukan perbandingan terhadap hasil uji dari ketiga algoritma tersebut yang dapat dilihat pada Tabel 3.1.

Tabel 3.19 Perbandingan *A Star, Dijkstra* dan *Floyd Warshall*

No. Pengujian	Titik Awal	Titik Akhir	Astar				Dijkstra				Floyd Warshall			
			Path Count	Weight	Play Turn	Checked Node	Path Count	Weight	Play Turn	Checked Node	Path Count	Weight	Play Turn	Checked Node
1	1.0	0.9	9	10	4	19	9	10	4	51	9	10	4	51
2	1.0	1.8	9	11	4	38	9	11	4	63	9	11	4	56
3	1.0	2.9	9	13	5	56	9	13	5	70	11	13	5	53
4	1.0	3.8	10	11	4	38	9	11	4	53	11	12	4	78
5	1.0	4.8	10	13	5	62	10	13	5	72	12	15	5	88
6	1.0	5.9	12	15	5	74	12	15	5	86	12	14	5	78
7	1.0	6.9	13	16	6	76	13	16	6	90	13	18	6	86
8	1.0	7.8	13	15	5	79	13	15	5	96	13	15	5	96
9	1.0	8.9	15	18	6	91	15	19	7	98	15	18	6	99
10	1.0	9.8	15	17	6	83	15	18	6	95	15	18	6	95
11	2.1	0.9	9	11	4	19	9	11	4	71	9	11	4	71
12	2.1	1.8	7	9	3	11	7	10	4	48	7	10	4	48
13	2.1	2.9	7	9	3	11	7	9	3	48	7	9	3	48
14	2.1	3.8	9	11	4	31	8	11	4	54	7	9	3	81
15	2.1	4.8	8	11	4	29	8	11	4	86	8	11	4	74
16	2.1	5.9	12	14	5	39	10	14	5	84	12	15	5	84
17	2.1	6.9	11	14	5	40	11	14	5	89	11	14	5	89
18	2.1	7.8	11	13	5	40	11	13	5	97	12	13	5	97
19	2.1	8.9	13	13	5	53	15	17	6	97	13	16	6	98
20	2.1	9.8	13	15	5	33	13	17	6	97	13	15	5	97
21	4.0	0.9	12	13	5	50	12	13	5	85	12	13	5	85
22	4.0	1.8	10	12	4	40	10	12	4	71	10	12	4	71
23	4.0	2.9	10	12	4	40	10	11	4	71	10	11	4	71
24	4.0	3.8	10	12	4	35	10	12	4	70	10	12	4	70
25	4.0	4.8	7	10	4	37	7	10	4	77	7	10	4	77
26	4.0	5.9	9	12	4	44	13	15	5	80	13	15	5	80
27	4.0	6.9	10	13	5	64	10	13	5	94	10	13	5	94
28	4.0	7.8	10	12	4	44	10	12	4	95	10	12	4	95
29	4.0	8.9	12	15	5	67	12	15	5	98	12	16	5	98
30	4.0	9.8	12	15	5	67	12	15	5	94	12	15	5	94
31	5.1	0.9	15	17	6	51	12	13	5	92	12	13	5	92
32	5.1	1.8	13	15	5	44	10	12	4	77	10	13	5	96
33	5.1	2.9	13	15	5	44	10	12	4	77	10	12	4	77
34	5.1	3.8	11	14	5	49	8	10	4	77	8	10	4	78
35	5.1	4.8	10	14	5	52	7	10	4	71	7	10	4	72
36	5.1	5.9	10	13	5	37	9	12	4	72	9	12	4	72
37	5.1	6.9	11	12	4	28	8	11	4	92	8	11	4	79
38	5.1	7.8	9	12	4	37	8	10	4	79	8	11	4	79
39	5.1	8.9	9	10	4	11	10	13	5	96	12	14	5	98
40	5.1	9.8	7	8	3	25	10	13	5	95	10	12	5	85
41	7.0	0.9	15	16	6	47	15	17	6	98	15	17	6	97
42	7.0	1.8	14	15	5	38	13	16	6	93	13	15	5	90
43	7.0	2.9	14	15	5	38	14	15	5	90	13	15	5	90
44	7.0	3.8	11	14	5	41	12	14	5	82	11	15	5	81
45	7.0	4.8	10	14	5	37	13	15	5	83	10	14	5	82
46	7.0	5.9	13	15	5	24	13	15	5	70	10	13	5	83
47	7.0	6.9	9	13	5	36	13	14	5	76	14	15	5	83
48	7.0	7.8	9	12	4	22	13	14	5	65	7	11	4	58
49	7.0	8.9	11	12	4	15	11	12	4	62	11	12	4	62
50	7.0	9.8	9	10	4	35	9	10	4	52	9	10	4	52

No. Pengujian	Titik Awal	Titik Akhir	Astar				Dijkstra				Floyd Warshall			
			Path Count	Weight	Play Turn	Checked Node	Path Count	Weight	Play Turn	Checked Node	Path Count	Weight	Play Turn	Checked Node
51	8.1	0.9	15	17	6	51	15	17	6	97	16	17	6	97
52	8.1	1.8	13	15	5	44	14	16	6	93	13	16	6	89
53	8.1	2.9	13	15	5	44	13	15	5	89	14	15	5	89
54	8.1	3.8	11	14	5	49	14	16	6	81	11	14	5	84
55	8.1	4.8	10	14	5	52	12	13	5	79	10	13	5	90
56	8.1	5.9	10	13	5	37	12	13	5	79	12	15	5	79
57	8.1	6.9	11	12	4	28	11	12	4	66	11	14	5	66
58	8.1	7.8	9	12	4	37	9	10	4	63	11	12	4	63
59	8.1	8.9	9	10	4	11	9	10	4	54	9	10	4	98
60	8.1	9.8	7	8	3	25	7	8	3	37	8	8	3	37
Jumlah			648	781	277	2499	653	784	283	4717	647	785	279	4800

Untuk mempermudah pembacaan data dibuatlah diagram yang diambil dari Tabel 3.1 yang dapat dilihat pada Gambar 3.1



Gambar 3.1 Diagram Perbandingan Algoritma *A Star, Dijkstra* dan *Floyd Warshall*

Algoritma *Dijkstra* dan *Algoritma Floyd Warshall* bekerja dengan cara mengecek seluruh *node* tetangga yang ada pada graph maka semakin jauh jarak titik awal ke titik akhir maka *node* yang dicek akan semakin banyak. Algoritma *A Star* menggunakan fungsi *heuristic* yaitu fungsi untuk mengurangi *node* yang dicek sehingga kinerja dari algoritma tersebut lebih efektif.

Pada Gambar 3.1 didapatkan bahwa dalam 60 kali percobaan dengan titik awal dan akhir yang berbeda. Selama pengujian algoritma *A Star* menampilkan data dengan nilai 781 *weight*, 648 *Path Count*, 277 *Play turn* dan 2499 pengecekkan node sedangkan algoritma *Dijkstra* menampilkan data dengan nilai 784 *weight*, 653 *Path Count*, *283 Play Turn* dan 4717 pengecekkan node dan *Floyd Warshall* menunjukan data dengan nilai 785 *weight*, 647 *Path Count*, 279 dan 4800 pengecekan node.

Dari hasil pengujian sebanyak 60 kali, ketiga algoritma tersebut berhasil menemukan titik akhir pada game Bacteria Defense yang memiliki graph beraturan. Dari ketiga Algoritma tersebut dapat disimpulkan bahwa Algoritma *A star* cocok diterapkan pada permainan Bacteria Defense karena Algoritma *A star* melakukan Weight, Play Turn dan

pengecekan node lebih sedikit dari algoritma *Dijkstra* dan *Floyd Warshall* sedangkan Algoritma *Floyd Warshall* memilih *path count* lebih sedikit dari Algoritma *A Star* dan *Dijkstra*.

4. KESIMPULAN

Dari hasil implementasi dan pengujian yang telah dilakukan sebelumnya, maka dapat ditarik kesimpulan bahwa Algoritma *A star* cocok diterapkan pada permainan Bacteria Defense karena Algoritma *A star* mendapatkan Weight, Play Turn dan pengecekan node lebih sedikit dari algoritma *Dijkstra* dan *Floyd Warshall* sedangkan Algoritma *Floyd Warshall* memilih *path count* lebih sedikit dari Algoritma *A Star* dan *Dijkstra*.

5. SARAN

Berdasarkan hasil analisis dan pembahasan yang telah dilakukan sebelumnya, maka terdapat beberapa saran yang digunakan untuk penelitian selanjutnya, antara lain :

1. Melakukan Perbandingan Algoritma *A Star* dengan algoritma lain seperti Algoritma *D**, *Theta** dan *Lazy Theta** untuk mencari algortima yang bekerja lebih baik.
2. Melakukan penerapan algoritma lain pada *game* yang bergenre *turn base strategy* dengan *graph* yang berbeda

DAFTAR PUSTAKA

[1] Attoriq, Rio. M, Anggara, Fajar, Jumeila, Selva. Fithri, 2016, Penerapan Algoritma Fuzzy Logic Sugeno dan Algoritma A Star pada Game Battle City, , Teknik Informatika, STMIK GI MDP PALEMBANG

[2] Fitria, Triansyah Apri, 2013, Implementasi Algoritma Dijkstra Dalam Aplikasi untuk menentukan Lintasan Terpedek Jalan Darat Antar Kota Di Sumatera Bagian Selatan, InstitudeDarmajaya.

[3] Kriswanto, Y Rudi, R, Kristoforus, Jawa, Bendi dan Aliyanto, Arif, 2014, Penentuan Jarak Terpendek Rute Transmusi Dengan Algoritma Floyd Warshall,Teknik Informatika, Sekolah TeknikMusi.

[4] Barnouti, Hazim. Nawaf, Al-Dabbagh, Mahmod Sameer Sinan, Naser, Sahib.Abdul. Mustafa, 2016, Pathfinding in Strategy Games and Maze Solving Using A Star Search Algorithm, Journal of Computer and Communication. Al-Mansour University Colle

[5] Ni Kai, Zhang Yao- ting, Ma Yue-peng, 2014, Shortest Path Analysis Based on Dijkstra's Algorithm in Emergency Response System, Shanghai Institute of Work Safety Science

[6] Febrian, Yusuf. M., Umri, Zainul. M, Pradesan Iis, Yohannes, 2016 "Penerapan *Algoritma Floyd Warshall* dan Metode FSM pada Permainan *MAZE TREASURE*", Teknik Informatika, STMIK GI MDP PALEMBANG