Pengaruh Pemilihan Fungsi Heuristik pada Algoritma AStar Terhadap Kinerja Aplikasi Pencarian Rute Terpendek Diantara Dua Titik

Nicholas Chen 13519029
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13519029@std.stei.itb.ac.id

***Abstrak*—Beberapa aplikasi yang berkembang saat ini membutuhkan sebuah algoritma yang dapat mencari rute terpendek dari sebuah titik ke titik tujuan dengan waktu yang singkat. Oleh karena itu, Algoritma *greedy* seperti Djikstra dan *Uniform Cost Search* yang membutuhkan waktu lama perlu dikembangkan lebih lanjut. Salah satu solusinya adalah dengan menambahkan fungsi heuristik kedalam pencariannya. Namun, tidak semua fungsi heuristik dapat mempersempit lingkup pencarian dengan baik.**

Pada makalah ini fungsi-fungsi heuristik yang sudah umum seperti Fungsi *Manhattan*, *Euclidean*, dan *Chebysev* masing-masing diuji performansinya dan dibandingkan dengan satu sama lain. Berdasarkan hasil pengujian, semua algoritma yang menggunakan fungsi heruistik berhasil mengungguli algoritma seperti UCS yang tidak memakai fungsi heuristik. Selain itu didapatkan urutan peringkat performansi dari yang paling baik, yaitu: AStar dengan Fungsi *Manhattan*, AStar dengan Fungsi *Euclidean*, AStar dengan Fungsi *Chebysev*.

Keywords—pencarian rute; fungsi heuristik; AStar; Manhattan; Chebysev: Euclidian; Uniform Cost Search

I. Pendahuluan

Persoalan pencarian rute antara 2 titik adalah salah satu persoalan umum yang sering ditemui pada program maupun aplikasi yang berkembang dewasa ini. Mulai dari aplikasi peta seperti *google map* sampai aplikasi interaktif seperti *video game* membutuhkan algoritma penyelesaian untuk mencari rute terpendek. Saat ini sudah banyak algoritma yang dibuat untuk menyelesaikan persoalan ini, namun tidak semua algoritma tersebut dapat mengkomputasi solusi dari persoalan ini dengan mangkus dan cepat. Padahal kebanyakan aplikasi dan program terutama yang sudah *realtime*, menuntut hasil dari komputasi solusi dengan waktu terbatas yang relatif singkat.

Untuk menanggapi masalah tersebut, pada algoritma pencarian rute Uniform Cost Search (UCS) dan Algoritma Djikstra yang ditemukan oleh Edsger W. Djikstra dikembangkan lebih lanjut dan diberikan fungsi heuristik agar algoritma tersebut dapat lebih mangkus dalam pencarian rute menuju titik tujuan yang kemudian disebut Algoritma AStar. Algoritma yang sebelumnya perlu mengecek setiap simpul untuk mencari rute terpendek, dengan bantuan fungsi ini dapat langsung memfokuskan pencarian untuk simpul yang lebih berpotensi menjadi rute terpendek.

Menurut definisinya, fungsi heuristik adalah fungsi yang dapat memprediksi jarak dari suatu titik ke titik tujuan dan sifatnya harus *underestimate*. Dalam penerapan implementasinya, formula dari fungsi ini sangat beragam. Beberapa dari fungsi-fungsi tersebut berhasil membuat alur pencarian rute lebih efesien, sedangkan sebagian lainnya hanya memperlambat proses pencarian solusi.

Pada makalah ini akan dibahas mengenai tingkat efektivitas kinerja dari beberapa formula fungsi heuristik umum seperti fungsi *Manhattan*, *Chebysev* dan *Euclidian*, lalu dibandingkan dengan algoritma pencarian yang tidak menggunakan fungsi heuristik. Fungsi-fungsi tersebut akan diuji dengan beberapa kasus skenario untuk diketahui apakah pemilihan fungsi heuristik mempengaruhi kinerja dari aplikasi pencarian rute dan jika mempengaruhi, bagaimana pengaruhnya dalam setiap kasus skenario yang diujikan.

II. Dasar Teori

A. *Uniform Cost Search*

Uniform Cost Search (UCS) merupakan algoritma pencarian rute yang tidak menggunakan fungsi heuristik. Algoritma ini sekilas mirip dengan algoritma Djikstra yang juga merupakan algoritma pencarian rute yang tidak menggunakan fungsi heuristik, namun UCS hanya memasukkan setiap fungsi yang perlu dicek kedalam *array* simpul hidup sedangkan Djikstra memasukkan setiap simpul kedalam *array* simpul hidup dan menginisiasi jarak meraka dengan *infinite*.

	0	1	2	3	4	5	6	7	8	9
0	1.41	1.00	1.41	2.41	3.41	4.41	5.41	6.41	7.41	0.00
1	1.00	0.00	1.00	2.00	3.00	4.00	5.00	6.00	7.00	0.00
2	1.41	1.00	1.41	2.41	3.41	4.41	5.41	6.41	7.41	0.00
3	2.41	2.00	2.41	2.83	3.83	4.83	5.83	6.83	7.83	0.00
4	3.41	3.00	3.41	3.83	4.24	5.24	6.24	7.24	0.00	0.00
5	4.41	4.00	4.41	4.83	5.24	5.66	6.66	7.66	0.00	0.00
6	5.41	5.00	5.41	5.83	6.24	6.66	7.07	0.00	0.00	0.00
7	6.41	6.00	6.41	6.83	7.24	7.66	0.00	0.00	0.00	0.00
8	7.41	7.00	7.41	7.83	0.00	0.00	0.00	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

```
Jarak = 6.65685424949238
Total node entering Simpul Hidup = 69
Total visited node = 53
```

gambar 1. Pencarian rute dengan UCS dari (1,1) ke (5,6). Titik yang masuk kedalam array simpul hidup adalah titik yang tidak berwarna hitam.

Alur dari algoritma ini cukup sederhana. Dimulai dari simpul awal sebagai simpul yang sedang dicek, simpul tersebut akan diperluas berdasarkan *cost* terkecil sampai simpul tujuan ditemukan. Algoritma ini membutuhkan sebuah *array* simpul hidup yang menyimpan kandidat simpul yang didapat dari perluasan simpul sebelumnya. Seperti yang sudah dijelaskan sebelumnya, algoritma ini tidak menggunakan fungsi heuristik. Algoritma ini menggunakan jarak yang telah ditempuh dari simpul awal ke suatu simpul sebagai *cost* simpul.

$$f(n) = g(n)$$

f(n) = cost simpul ini
g(n) = jarak tempuh simpul awal ke simpul ini

Pertama, algoritma ini akan memperluas simpul awal dengan cara memasukkan setiap simpul tetangga dari simpul awal kedalam *array* simpul hidup terurut menaik berdasarkan jarak dari simpul tersebut ke simpul awal. Setelah itu, simpul terdepan pada *array* simpul hidup akan diperluas dengan cara yang sama, hanya saja jarak setiap simpul baru merupakan jarak simpul saat ini dengan simpul awal ditambah dengan jarak simpul saat ini dengan simpul yang baru diperluas. Proses tersebut akan diulang terus menerus sampai simpul terdepan pada *array* simpul hidup merupakan simpul tujuan. Lintasan hasil dapat diperoleh dengan melakukan pemeriksaan *bactracking* dari simpul tujuan hingga simpul awal. Jika *array* simpul hidup habis sebelum ditemukan simpul tujuan, maka disimpulkan bahwa simpul tujuan *unreachable* dari simpul awal.

```
// Berikut merupakan salah satu alternatif
pseudocode Algoritma UCS

function Double UCS(Node startNode, Node endNode){
  simpulHidup = [];
  currNode = startNode;
  simpulHidup.add(currNode);

  while (simpulHidup not empty){
    currNode.visited(true);
    simpulHidup.remove(0);
    if (currNode == endNode){
      return currNode.getTravelledDistance();
    }
    simpulHidup.add(currNode.adjNode());
    simpulHidup.sortBy(travelledDistance, asc);
    currNode = simpulHidup.head();
  }
  // case not found
  return null;
}
```

gambar 2. Salah satu alternatif pseudocode Algoritma UCS

Alasan digunakan Algoritma UCS sebagai algoritma pencarian rute yang tidak menggunakan fungsi heuristik karena algoritma ini lebih mirip dengan AStar dibanding dengan Djikstra, sehingga proses analisis Algoritma UCS dengan Algoritma AStar yang menggunakan fungsi heuristik akan lebih relevan.

B. AStar using Manhattan Heuristic Function

Algoritma AStar versi ini merupakan algoritma perkembangan lanjutan dari Algoritma UCS yang menggunakan *Manhattan Distance* sebagai fungsi heuristik. Dengan Algortima ini, *cost* dari sebuah simpul merupakan *cost* simpul dari Algoritma UCS ditambah dengan jarak heuristik simpul ini ke simpul tujuan. *Manhattan Distance* pada fungsi heuristik akan memprediksi jarak kedua titik dengan total selisih absis kedua titik dan selisih ordinat kedua titik.

$$f(n) = g(n) + h(n)$$

f(n) = cost simpul ini
g(n) = jarak tempuh simpul awal ke simpul ini
h(n) = *Manhattan Distance* simpul ini ke simpul tujuan

$$Manhattan\ Distance = |x_1 - x_2| + |y_1 - y_2|$$

(x_1, y_1) = koordinat simpul 1
(x_2, y_2) = koordinat simpul 2

Alur kerja dari algoritma ini sama seperti Algoritma UCS, hanya saja untuk penentuan simpul yang akan diperluas menggunakan teknik perhitungan *cost* yang sudah disertakan dengan fungsi heuristik *Manhattan*.

	0	1	2	3	4	5	6	7	8	9
0	11.00	10.00	9.00	8.00	7.00	6.00	5.00	6.00	7.00	8.00
1	10.00	9.00	8.00	7.00	6.00	5.00	4.00	5.00	6.00	7.00
2	9.00	8.00	7.00	6.00	5.00	4.00	3.00	4.00	5.00	6.00
3	8.00	7.00	6.00	5.00	4.00	3.00	2.00	3.00	4.00	5.00
4	7.00	6.00	5.00	4.00	3.00	2.00	1.00	2.00	3.00	4.00
5	6.00	5.00	4.00	3.00	2.00	1.00	0.00	1.00	2.00	3.00
6	7.00	6.00	5.00	4.00	3.00	2.00	1.00	2.00	3.00	4.00
7	8.00	7.00	6.00	5.00	4.00	3.00	2.00	3.00	4.00	5.00
8	9.00	8.00	7.00	6.00	5.00	4.00	3.00	4.00	5.00	6.00
9	10.00	9.00	8.00	7.00	6.00	5.00	4.00	5.00	6.00	7.00

gambar 3. Nilai Manhattan Heuristic Function setiap simpul pada peta 10x10 dengan (5, 6) sebagai simpul tujuan

	0	1	2	3	4	5	6	7	8	9
0	12.41	11.00	10.41	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	11.00	9.00	9.00	9.83	0.00	0.00	0.00	0.00	0.00	0.00
2	10.41	9.00	8.41	8.41	9.24	0.00	0.00	0.00	0.00	0.00
3	0.00	9.83	8.41	7.83	7.83	8.66	0.00	0.00	0.00	0.00
4	0.00	0.00	9.24	7.83	7.24	7.24	8.07	0.00	0.00	0.00
5	0.00	0.00	0.00	8.66	7.24	6.66	6.66	0.00	0.00	0.00
6	0.00	0.00	0.00	0.00	10.07	8.66	8.07	0.00	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

```
Jarak = 6.656854249492381
Total node entering Simpul Hidup = 29
Total visited node = 6
```

gambar 4. Hasil pencarian rute dari titik (1, 1) ke (5, 6) dengan AStar using Manhattan Heuristic Function

C. AStar using Chebysev Heuristic Function

Algoritma AStar versi ini juga merupakan algoritma perkembangan lanjutan dari Algoritma UCS, namun kali ini menggunakan *Chebysev Distance* sebagai fungsi heuristik. Dengan Algortima ini, *cost* dari sebuah simpul merupakan *cost* simpul dari Algoritma UCS ditambah dengan jarak heuristik simpul ini ke simpul tujuan. *Chebysev Distance* pada fungsi heuristik akan memprediksi jarak kedua titik dengan mencari nilai terbesar diantara selisih absis kedua titik dan selisih ordinat kedua titik.

$$f(n) = g(n) + h(n)$$

f(n) = cost simpul ini
g(n) = jarak tempuh simpul awal ke simpul ini
h(n) = *Chebysev Distance* simpul ini ke simpul tujuan

$$Chebysev\ Distance = \max(|x_1 - x_2|, |y_1 - y_2|)$$

(x_1, y_1) = koordinat simpul 1
(x_2, y_2) = koordinat simpul 2

Alur kerja dari algoritma ini juga sama seperti Algoritma UCS, hanya saja untuk penentuan simpul yang akan diperluas menggunakan teknik perhitungan *cost* yang sudah disertakan dengan fungsi heuristik *Chebysev*.

	0	1	2	3	4	5	6	7	8	9
0	6.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
1	6.00	5.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
2	6.00	5.00	4.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00
3	6.00	5.00	4.00	3.00	2.00	2.00	2.00	2.00	2.00	3.00
4	6.00	5.00	4.00	3.00	2.00	1.00	1.00	1.00	2.00	3.00
5	6.00	5.00	4.00	3.00	2.00	1.00	0.00	1.00	2.00	3.00
6	6.00	5.00	4.00	3.00	2.00	1.00	1.00	1.00	2.00	3.00
7	6.00	5.00	4.00	3.00	2.00	2.00	2.00	2.00	2.00	3.00
8	6.00	5.00	4.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00
9	6.00	5.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00

gambar 5. Nilai Chebysev Heuristic Function setiap simpul pada peta 10x10 dengan (5, 6) sebagai simpul tujuan

	0	1	2	3	4	5	6	7	8	9
0	7.41	6.00	6.41	7.41	8.41	0.00	0.00	0.00	0.00	0.00
1	7.00	5.00	5.00	6.00	7.00	8.83	0.00	0.00	0.00	0.00
2	7.41	6.00	5.41	5.41	6.41	7.41	0.00	0.00	0.00	0.00
3	8.41	7.00	6.41	5.83	5.83	6.83	8.66	0.00	0.00	0.00
4	0.00	8.83	7.41	6.83	6.24	6.24	7.24	0.00	0.00	0.00
5	0.00	0.00	0.00	8.66	7.24	6.66	6.66	0.00	0.00	0.00
6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

```
Jarak = 6.65685424949238
Total node entering Simpul Hidup = 34
Total visited node = 15
```

gambar 6. Hasil pencarian rute dari titik (1, 1) ke (5, 6) dengan AStar using Chebysev Heuristic Function

D. AStar using Euclidian Heuristic Function

Pada Algoritma AStar versi ini, formula dari fungsi heuristik yang digunakan adalah *Euclidean Distance* antara 2 titik. Dengan Algortima ini, *cost* dari sebuah simpul merupakan *cost* simpul dari Algoritma UCS ditambah dengan jarak heuristik simpul ini ke simpul tujuan. *Euclidean Distance* pada fungsi heuristik akan memprediksi jarak kedua titik dengan total selisih absis kedua titik dan selisih ordinat kedua titik.

$$f(n) = g(n) + h(n)$$

f(n) = cost simpul ini
g(n) = jarak tempuh simpul awal ke simpul ini
h(n) = *Euclidean Distance* simpul ini ke simpul tujuan

$$Euclidean\ Distance = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2|}$$

(x_1, y_1) = koordinat simpul 1
(x_2, y_2) = koordinat simpul 2

Tidak berbeda dengan Konsep Algoritma AStar versi sebelumnya, algoritma ini menggunakan teknik perhitungan *cost* yang sudah disertakan dengan fungsi heuristik *Euclidean* untuk menentukan simpul berikutnya yang akan diperluas.

	0	1	2	3	4	5	6	7	8	9
0	7.81	7.07	6.40	5.83	5.39	5.10	5.00	5.10	5.39	5.83
1	7.21	6.40	5.66	5.00	4.47	4.12	4.00	4.12	4.47	5.00
2	6.71	5.83	5.00	4.24	3.61	3.16	3.00	3.16	3.61	4.24
3	6.32	5.39	4.47	3.61	2.83	2.24	2.00	2.24	2.83	3.61
4	6.08	5.10	4.12	3.16	2.24	1.41	1.00	1.41	2.24	3.16
5	6.00	5.00	4.00	3.00	2.00	1.00	0.00	1.00	2.00	3.00
6	6.08	5.10	4.12	3.16	2.24	1.41	1.00	1.41	2.24	3.16
7	6.32	5.39	4.47	3.61	2.83	2.24	2.00	2.24	2.83	3.61
8	6.71	5.83	5.00	4.24	3.61	3.16	3.00	3.16	3.61	4.24
9	7.21	6.40	5.66	5.00	4.47	4.12	4.00	4.12	4.47	5.00

gambar 7. Nilai Euclidean Heuristic Function setiap simpul pada peta 10x10 dengan (5, 6) sebagai simpul tujuan

	0	1	2	3	4	5	6	7	8	9
0	9.22	8.07	7.82	8.25	0.00	0.00	0.00	0.00	0.00	0.00
1	8.21	6.40	6.66	7.00	8.30	0.00	0.00	0.00	0.00	0.00
2	8.12	6.83	6.41	6.66	7.02	8.40	0.00	0.00	0.00	0.00
3	0.00	8.21	6.89	6.43	6.66	7.06	8.66	0.00	0.00	0.00
4	0.00	0.00	8.37	6.99	6.48	6.66	7.24	0.00	0.00	0.00
5	0.00	0.00	0.00	8.66	7.24	6.66	6.66	0.00	0.00	0.00
6	0.00	0.00	0.00	0.00	9.31	8.07	8.07	0.00	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

```
Jarak = 6.656854249492381
Total node entering Simpul Hidup = 33
Total visited node = 10
```

gambar 8. Hasil pencarian rute dari titik (1, 1) ke (5, 6) dengan AStar using Euclidean Heuristic Function

Dibandingkan dengan fungsi Manhattan dan Fungsi Chebysev, Fungsi Euclidean dapat lebih akurat dalam memprediksi jarak di antara kedua titik, terlebih lagi untuk kasus simpul saling bertetangga dengan simpul diagonal disekitarnya.

III. Pengujian dan Analisis

Untuk menguji performansi dari Algoritma dengan masing-masing fungsi heuristik, penulis membuat sebuah program kecil CLI (*Command Line user Interface*) menggunakan bahasa Java yang dapat memecahkan persoalan pencarian rute menggunakan algoritma-algoritma yang telah dibahas sebelumnya. Pada program tersebut disediakan peta *tiles* berukuran 10x10 yang dibaca dari file eksternal. Setiap simpul pada peta bertetangga dengan simpul lain yang ada disekitarnya, termasuk yang posisinya diagonal terhadap simpul tersebut.

Setiap algoritma tersebut akan diuji-cobakan kedalam 3 kasus persoalan berbeda yang sudah dirancang oleh penulis. Performansi dari algoritma akan ditentukan dari *Total Search Cost* yang didapat dari banyaknya simpul yang harus dikunjungi (*Visited Node*) dan banyaknya simpul yang pernah dimasukkan kedalam *array* simpul hidup. Semakin tinggi nilai *Total Search Cost* maka tingat performansi dari algoritma tersebut akan semakin rendah. Pada pengujian ini, Algoritma UCS akan digunakan sebagai variabel kontrol.

```
Computation Result Map:
     0     1     2     3     4     5     6     7     8     9
0  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
1  0.00  0.00  0.00  0.00  8.00  7.41  7.41  8.41  9.41  0.00
2  0.00  0.00  0.00  0.00  7.00  0.00  7.83  7.83  8.83 10.66
3  0.00  0.00  0.00  0.00  6.00  0.00  8.83  8.24  8.24  9.24
4  0.00  0.00  0.00  0.00  5.00  0.00 10.66  9.24  8.66  8.66
5  0.00  0.00  0.00  0.00  6.00  0.00  0.00 11.07  9.66 10.07
6  0.00  0.00  0.00  0.00  7.00  0.00  0.00  0.00  0.00  0.00
7  0.00  0.00  0.00  0.00  8.00  7.41  7.41  8.41  9.41  0.00
8  0.00  0.00  0.00  0.00  9.00  8.41  8.83  9.83 10.83  0.00
9  0.00  0.00  0.00  0.00 10.83 10.41 10.83  0.00  0.00  0.00
```

gambar 9. Hasil pencarian rute dari titik (4, 4) ke (4, 9) dengan AStar using Chebysev Heuristic Function

Gambar 9 merupakan contoh hasil peta yang di-*generate* secara otomatis oleh program. Pada peta tersebut, simpul awal dan simpul akhir merupakan simpul yang diberi warna hijau dan merah. Simpul yang diwarnai ungu merupakan simpul yang tidak dapat dilalui (*Obstacle*). Rute hasil pencarian dari simpul awal ke simpul akhir diberi warna biru muda. Setiap simpul yang dikunjungi (*visited*) untuk diperluas diberi warna biru tua, sedangkan simpul yang ada di *array* simpul hidup namun tidak pernah diperluas akan diberi warna putih. Simpul yang diberi warna hitam adalah simpul yang tidak pernah dicek oleh algoritma. angka pada setiap titik merepresentasikan nilai f(n) untuk simpul tersebut

A. *Test Case 1*

Pada test case ini, didalam peta tidak diberikan *obstacle* apapun. Pada persoalan dari test ini, setiap algoritma harus mencari rute terpendek dari titik (1, 0) ke titik (6 ,9).

```
Computation Result Map:
     0     1     2     3     4     5     6     7     8     9
0  1.00  1.41  2.41  3.41  4.41  5.41  6.41  7.41  8.41  9.41
1  0.00  1.00  2.00  3.00  4.00  5.00  6.00  7.00  8.00  9.00
2  1.00  1.41  2.41  3.41  4.41  5.41  6.41  7.41  8.41  9.41
3  2.00  2.41  2.83  3.83  4.83  5.83  6.83  7.83  8.83  9.83
4  3.00  3.41  3.83  4.24  5.24  6.24  7.24  8.24  9.24 10.24
5  4.00  4.41  4.83  5.24  5.66  6.66  7.66  8.66  9.66 10.66
6  5.00  5.41  5.83  6.24  6.66  7.07  8.07  9.07 10.07 11.07
7  6.00  6.41  6.83  7.24  7.66  8.07  8.49  9.49 10.49 11.49
8  7.00  7.41  7.83  8.24  8.66  9.07  9.49  9.90 10.90 11.90
9  8.00  8.41  8.83  9.24  9.66 10.07 10.49 10.90 11.31 12.31
Distance = 11.071067811865476
Total node entering Simpul Hidup = 100
Total visited node = 96
Total Search Cost = 196
```

gambar 10. Hasil test case 1: rute dari titik (1, 0) ke (6, 9) dengan UCS

```
Computation Result Map:
     0     1     2     3     4     5     6     7     8     9
0 16.00 15.41  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
1 14.00 14.00 14.83  0.00  0.00  0.00  0.00  0.00  0.00  0.00
2 14.00 13.41 13.41 14.24  0.00  0.00  0.00  0.00  0.00  0.00
3 14.83 13.41 12.83 12.83 13.66  0.00  0.00  0.00  0.00  0.00
4  0.00 14.24 12.83 12.24 12.24 13.07  0.00  0.00  0.00  0.00
5  0.00  0.00 13.66 12.24 11.66 11.66 12.49 12.49 12.49 12.49
6  0.00  0.00  0.00 13.07 11.66 11.07 11.07 11.07 11.07 11.07
7  0.00  0.00  0.00  0.00 14.49 13.07 12.49 12.49 12.49 12.49
8  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
9  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
Distance = 11.071067811865476
Total node entering Simpul Hidup = 40
Total visited node = 10
Total Search Cost = 50
```

gambar 11. Hasil test case 1: rute dari titik (1, 0) ke (6, 9) dengan AStar using Manhattan Function

```
Computation Result Map:
     0     1     2     3     4     5     6     7     8     9
0 11.82 11.41 11.63 11.90 12.22 12.63  0.00  0.00  0.00  0.00
1 10.30 10.43 10.60 10.81 11.07 11.40 12.66  0.00  0.00  0.00
2 10.85 10.36 10.48 10.63 10.82 11.07 11.41 12.71  0.00  0.00
3 11.49 10.96 10.44 10.54 10.66 10.83 11.07 11.43 12.82  0.00
4 13.05 11.66 11.11 10.57 10.63 10.71 10.85 11.07 11.48 13.07
5  0.00  0.00 12.73 11.33 10.76 10.78 10.82 10.89 11.07 11.66
6  0.00  0.00  0.00 13.07 11.66 11.07 11.07 11.07 11.07 11.07
7  0.00  0.00  0.00  0.00 13.58 12.19 11.65 11.72 11.90 12.49
8  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
9  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
Distance = 11.071067811865476
Total node entering Simpul Hidup = 61
Total visited node = 32
Total Search Cost = 93
```

gambar 12. Hasil test case 1: rute dari titik (1, 0) ke (6, 9) dengan AStar using Euclidean Function

```
Computation Result Map:
     0     1     2     3     4     5     6     7     8     9
0 10.00  9.41  9.41  9.41 10.41 11.41 12.41 13.41  0.00  0.00
1  9.00  9.00  9.00  9.00  9.00 10.00 11.00 12.00  0.00  0.00
2 10.00  9.41  9.41  9.41  9.41  9.41 10.41 11.41 13.24  0.00
3 11.00 10.41  9.83  9.83  9.83  9.83  9.83 10.83 11.83  0.00
4 12.00 11.41 10.83 10.24 10.24 10.24 10.24 10.24 11.24 13.07
5  0.00 13.24 11.83 11.24 10.66 10.66 10.66 10.66 10.66 11.66
6  0.00  0.00  0.00 13.07 11.66 11.07 11.07 11.07 11.07 11.07
7  0.00  0.00  0.00  0.00 13.49 12.07 11.49 11.49 11.49 12.49
8  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
9  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
Distance = 11.071067811865476
Total node entering Simpul Hidup = 66
Total visited node = 43
Total Search Cost = 109
```

gambar 12. Hasil test case 1: rute dari titik (1, 0) ke (6, 9) dengan AStar using Chebysev Function

Pada pengujian ini didapatkan hasil peringkat performansi terbaik dari Fungsi Manhattan (50), Fungsi Euclidean (93), Fungsi Chebysev (109), dan yang terakhir Algoritma UCS (196). Setiap algoritma yang menggunakan fungsi heuristik berhasil melampaui performansi algoritma tanpa fungsi heuristik (UCS).

Walaupun Fungsi Euclidean dapat memprediksi jarak sebenarnya diantara 2 titik lebih baik dibanding Fungsi Manhattan, pada kasus ini Fungsi Manhattan lebih unggul karena Algoritma dengan Fungsi Manhattan hanya fokus mengecek salah satu kemungkinan jalur terpendek, tidak seperti Algoritma dengan Fungsi Euclidean yang mengecek lebih banyak kemungkinan jalur terpendek.

B. Test Case 2

Pada test case ini, didalam peta diberikan *obstacle* seperti pada gambar 13. Pada persoalan dari test ini, setiap algoritma harus mencari rute terpendek dari titik (3, 2) ke titik (5, 8).

```
     0       1       2       3       4       5       6       7       8       9
0 (0, 0) (0, 1) (0, 2) (0, 3) (0, 4) (0, 5) (0, 6) (0, 7) (0, 8) (0, 9)
1 (1, 0) (1, 1) (1, 2) (1, 3) (1, 4) (1, 5) (1, 6) (1, 7) (1, 8) (1, 9)
2 (2, 0) (2, 1) (2, 2) (2, 3) (2, 4) (2, 5) (2, 6) (2, 7) (2, 8) (2, 9)
3 (3, 0) (3, 1) (3, 2) (3, 3) (3, 4) (3, 5) (3, 6) (3, 7) (3, 8) (3, 9)
4 (4, 0) (4, 1) (4, 2) (4, 3) (4, 4) (4, 5) (4, 6) (4, 7) (4, 8) (4, 9)
5 (5, 0) (5, 1) (5, 2) (5, 3) (5, 4) (5, 5) (5, 6) (5, 7) (5, 8) (5, 9)
6 (6, 0) (6, 1) (6, 2) (6, 3) (6, 4) (6, 5) (6, 6) (6, 7) (6, 8) (6, 9)
7 (7, 0) (7, 1) (7, 2) (7, 3) (7, 4) (7, 5) (7, 6) (7, 7) (7, 8) (7, 9)
8 (8, 0) (8, 1) (8, 2) (8, 3) (8, 4) (8, 5) (8, 6) (8, 7) (8, 8) (8, 9)
9 (9, 0) (9, 1) (9, 2) (9, 3) (9, 4) (9, 5) (9, 6) (9, 7) (9, 8) (9, 9)
```

gambar 13. Peta untuk test case 2

```
Computation Result Map:
     0     1     2     3     4     5     6     7     8     9
0  3.83  3.41  3.00  3.41  3.83  4.83  0.00  9.66 10.07 10.49
1  2.83  2.41  2.00  2.41  3.41  4.41  0.00  8.66  9.07 10.07
2  2.41  1.41  1.00  0.00  3.83  4.83  0.00  7.66  8.66  9.66
3  2.00  1.00  0.00  0.00  4.83  5.24  6.24  7.24  8.24  9.24
4  2.41  1.41  1.00  0.00  3.83  4.83  5.83  0.00  8.66  9.66
5  2.83  2.41  2.00  2.41  3.41  4.41  5.41  0.00  9.66 10.07
6  3.83  3.41  3.00  3.41  3.83  4.83  5.83  0.00  8.66  9.66
7  4.83  4.41  4.00  4.41  4.83  5.24  6.24  7.24  8.24  9.24
8  5.83  5.41  5.00  5.41  5.83  6.24  6.66  7.66  8.66  9.66
9  6.83  6.41  6.00  6.41  6.83  7.24  7.66  8.07  9.07 10.07
Distance = 9.65685424949238
Total node entering Simpul Hidup = 91
Total visited node = 85
Total Search Cost = 176
```

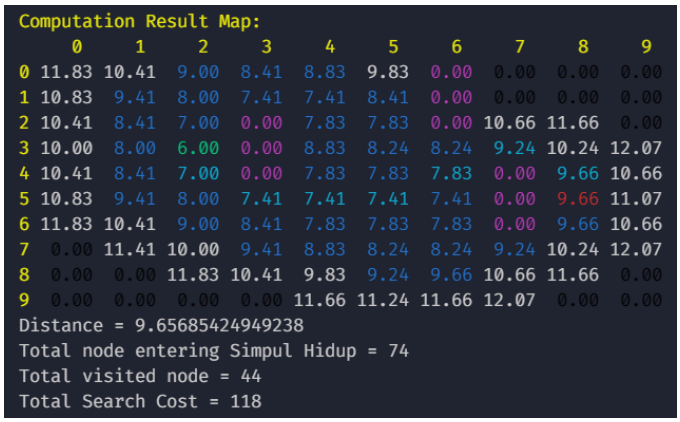
gambar 14. Hasil test case 2: rute dari titik (3, 2) ke (5, 8) dengan UCS

```
Computation Result Map:
     0     1     2     3     4     5     6     7     8     9
0  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
1  0.00 13.41 12.00 11.41  0.00  0.00  0.00  0.00  0.00  0.00
2 13.41 11.41 10.00  0.00 13.66 12.24  0.00 11.66 11.66  0.00
3 12.00 10.00  8.00  0.00 10.83 10.24 10.24 10.24 10.24 13.07
4 11.41  9.41  8.00  0.00  8.83  8.83  8.83  0.00  9.66 11.66
5 10.83  9.41  8.00  7.41  7.41  7.41  7.41  0.00  9.66 11.07
6 12.83 11.41 10.00  9.41  8.83  8.83  8.83  0.00  0.00  0.00
7  0.00 13.41 12.00 11.41 10.83 10.24 10.24 10.24  0.00  0.00
8  0.00  0.00  0.00  0.00 13.66 12.24 11.66 11.66  0.00  0.00
9  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
Distance = 9.65685424949238
Total node entering Simpul Hidup = 54
Total visited node = 26
Total Search Cost = 80
```

gambar 15. Hasil test case 2: rute dari titik (3, 2) ke (5, 8) dengan AStar using Manhattan Function

```
Computation Result Map:
     0     1     2     3     4     5     6     7     8     9
0  0.00 12.02 10.81 10.49 10.23 10.66  0.00  0.00  0.00  0.00
1 11.77 10.48  9.21  8.82  9.07  9.41  0.00  0.00  0.00  0.00
2 10.96  9.03  7.71  0.00  8.83  9.07  0.00 10.82 11.66  0.00
3 10.25  8.28  6.32  0.00  9.30  8.85  9.07  9.48 10.24 12.31
4 10.48  8.49  7.08  0.00  7.95  7.99  8.06  0.00  9.66 11.07
5 10.83  9.41  8.00  7.41  7.41  7.41  7.41  0.00  9.66 11.07
6 11.89 10.49  9.08  8.51  7.95  7.99  8.06  0.00  9.66 11.07
7  0.00 11.69 10.32  9.80  9.30  8.85  9.07  9.48 10.24 12.31
8  0.00  0.00  0.00 12.07 10.83 10.49 10.26 10.82 11.66  0.00
9  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
Distance = 9.65685424949238
Total node entering Simpul Hidup = 68
Total visited node = 37
Total Search Cost = 105
```

gambar 16. Hasil test case 2: rute dari titik (3, 2) ke (5, 8) dengan AStar using Euclidean Function



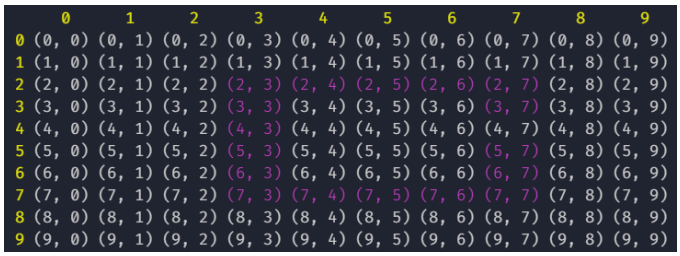
gambar 17. Hasil test case 2: rute dari titik (3, 2) ke (5, 8) dengan AStar using Chebysev Function

Pada pengujian kedua ini, didapatkan hasil peringkat performansi terbaik dari Fungsi Manhattan (80), Fungsi Euclidean (105), Fungsi Chebysev (118), dan yang terakhir Algoritma UCS (196). Setiap algoritma yang menggunakan fungsi heuristik pada pengujian ini juga berhasil melampaui performansi algoritma tanpa fungsi heuristik (UCS).

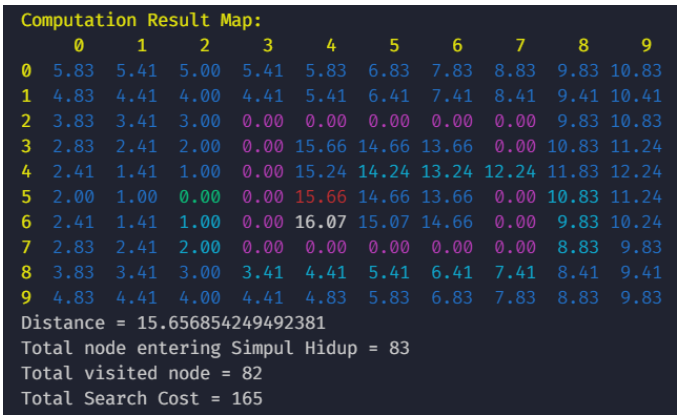
Penyebab algoritma dengan Fungsi *Manhattan* lebih tinggi tingkat performansinya dibandingkan algoritma dengan Fungsi *Euclidean* sama seperti pada *test case* 1. Pada algoritma dengan Fungsi *Manhattan* hanya fokus mengecek salah satu rute solusi (lewat bawah melalui simpul (5, 3)) sedangkan pada algoritma dengan Fungsi *Euclidean* mengecek rute lewat atas (simpul (1,3)) dan bawah (simpul (5,3)).

C. *Test Case 3*

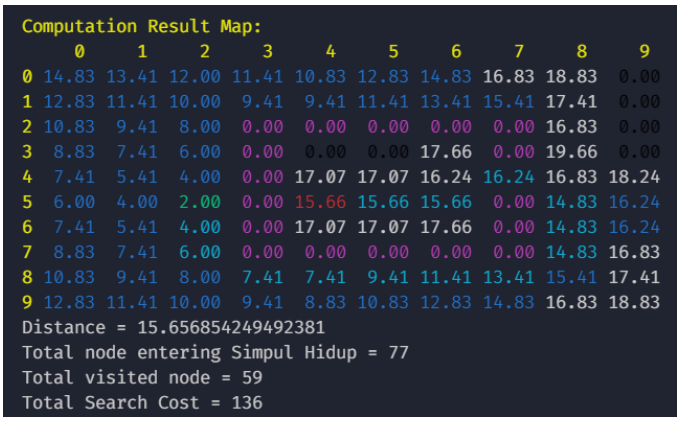
Pada test case terakhir ini, didalam peta diberikan *obstacle* berbentuk kotak dengan sebuah celah sempit seperti pada gambar 18. Pada persoalan dari test ini, setiap algoritma harus mencari rute terpendek dari titik (5, 2) ke titik (5, 4).



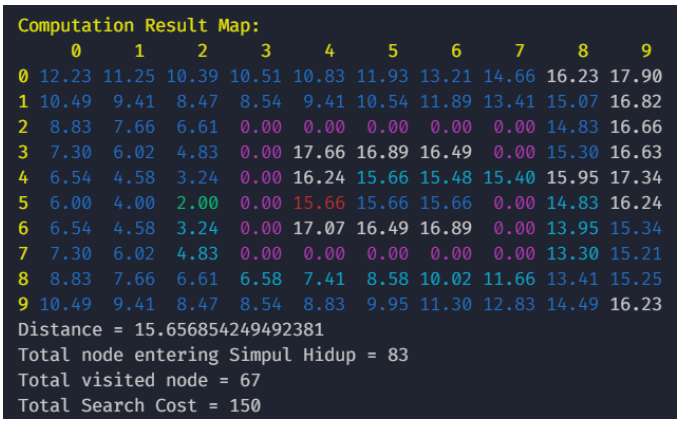
gambar 18. Peta untuk test case 3



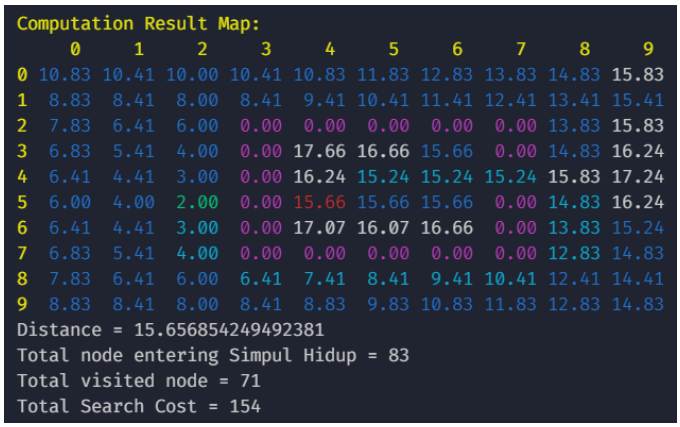
gambar 19. Hasil test case 2: rute dari titik (5, 2) ke (5, 4) dengan UCS



gambar 20. Hasil test case 2: rute dari titik (5, 2) ke (5, 4) dengan AStar using Manhattan Function



gambar 21. Hasil test case 2: rute dari titik (5, 2) ke (5, 4) dengan AStar using Euclidean Function



gambar 22. Hasil test case 2: rute dari titik (5, 2) ke (5, 4) dengan AStar using Chebysev Function

Pada pengujian terakhir ini, didapatkan hasil peringkat performansi terbaik yang sama dengan sebelumnya dengan urutan Fungsi Manhattan (136), Fungsi Euclidean (150), Fungsi Chebysev (154), dan yang terakhir Algoritma UCS (165). Setiap algoritma yang menggunakan fungsi heuristik pada pengujian ini juga berhasil melampaui performansi algoritma tanpa fungsi heuristik (UCS).

Untuk algoritma dengan Fungsi *Manhattan*, ketika algoritma ini menemukan celah rute untuk masuk kedalam kotak, algoritma ini tidak mengecek rute yang melalui jalur atas sehingga mengurangi jumlah simpul yang perlu diproses. Walaupun demikian, hasil rute terpendek yang diberikan algoritma ini tetap tepat.

Pada kasus ini, ketika algoritma dengan Fungsi *Chebysev* dan Fungsi *Euclidean* menemukan celah untuk masuk kedalam kotak, algoritma-algoritma ini tetap meng-*expand* rute bagian atas untuk dilakukan pengecekan. Hal ini membuat nilai *Total Search Cost* kedua algoritma ini mendekati algoritma tanpa Fungsi Heuristik (UCS). Bahkan, jika dilihat dari sisi waktu yang diperlukan untuk pemrosesan, algoritma dengan Fungsi *Chebysev* dan Fungsi *Euclidean* mungkin akan memakan waktu lebih lama dari Algoritma UCS karena proses pengecekan Algoritma UCS pada setiap simpulnya tidak serumit kedua algoritma tersebut.

IV. Kesimpulan

Algoritma yang menggunakan bantuan fungsi heuristik seperti algoritma AStar dengan fungsi heuristik *Manhattan*, *Euclidean*, dan *Chebysev* pada sebagian besar kasus selalu lebih efektif dalam mencari rute terpendek dibangingkan dengan algoritma seperti UCS (*Uniform Cost Search*) yang tidak menggunakan fungsi heuristik. Untuk kasus fungsi heuristik *Manhattan* dan *Euclidean*, walaupun algoritma dengan Fungsi *Euclidean* dapat memprediksi jarak antar simpul dengan lebih baik dibanding Fungsi Manhattan, algoritma dengan Fungsi Manhattan dapat mencari rute terpendek dengan komputasi jumlah simpul yang lebih sedikit dibandingkan algoritma dengan Fungsi Euclidean.

Secara keseluruhan, urutan tingkat performansi berdasarkan jumlah simpul yang diproses jika diurutkan dari yang paling baik adalah sebagai berikut: AStar dengan Fungsi *Manhattan*, AStar dengan Fungsi *Euclidean*, AStar dengan Fungsi *Chebysev*, dan yang terakhir Algoritma *Uniform Cost Search.*

Link Video Youtube

https://youtu.be/9xI57gUgUrQ .

Ucapan Terima Kasih

Rasa syukur saya panjatkan kepada Tuhan Yang Maha Esa karena rahmatnya sehingga penulis dapat menyelesaikan makalah ini dengan baik dan lengkap. Selain itu, penulis mengucapkan terima kasih kepada bapak dan ibu dosen pengampu kelas IF2211 Strategi Algoritma yang sudah dengan sabar mengajar penulis selama satu semester serta orang tua penulis yang tidak henti-hentinya memberi dukungan penuh terhadap studi penulis. Dengan makalah ini, penulis berharap agar dapat memberikan manfaat dan wawasan baru bagi para pembaca.

Referensi

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Route-Planning-Bagian1-2021.pdf - diakses pada tanggal 11 Mei 2021

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Route-Planning-Bagian2-2021.pdf - diakses pada tanggal 11 Mei 2021

[3] https://iq.opengenus.org/euclidean-vs-manhattan-vs-chebyshev-distance/#:~:text=Usage%20in%20Chess&text=the%20distance%20between%20squares%20on,its%20diagonals%20as%20coordinate%20axes. – diakses pada tanggal 11 Mei 2021

[4] https://www.youtube.com/watch?v=g024lzsknDo&t=1s – diakses pada tanggal 11 Mei 2021

[5] https://github.com/NicholasChen11/AStar-Simulator.git - link github program yang digunakan untuk pengujian

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 11 Mei 2021

Nicholas Chen - 13519029