BAB 3
ANALISIS DAN KEBUTUHAN ALGORITMA

3.1 Analisis Masalah

Masalah yang diangkat dalam penulisan tugas akhir ini adalah penerapan algoritma IDA* dan IDA* SNC. Pada *game* labirin dengan banyak pilihan jalan yang bercabang dan dibatasi oleh tembok-tembok, dibutuhkan algoritma pencarian jalur untuk karakter musuh-musuh yang optimal (jalur terpendek) agar dapat mengejar karakter pemain. Algoritma IDA* (*Iterative Deepening* A*) merupakan pengembangan dari algoritma A* yang mengatasi masalah pada *game* labirin dengan melakukan serangkaian iterasi pencarian sambil terus menerus meningkatkan nilai *f-limit*. Untuk setiap iterasi, nilai *f-limit* ditingkatkan menjadi $f(n_i)$ minimum dari seluruh simpul yang dibangkitkan namun belum diekspansi pada iterasi terakhir, *f-limit* yang baru akan digunakan sebagai batasan pencarian untuk iterasi berikutnya.

Di sisi lain, algoritma IDA* memerlukan waktu eksekusi yang lama karena berkemungkinan untuk selalu kembali mengunjungi simpul yang sama. Hal ini menyebabkan proses ekspansi simpul yang banyak dan berujung pada waktu eksekusi yang lama. Untuk mengatasi masalah ini akan ditambahkan *Stochastic Node Caching* (SNC). Algoritma IDA* SNC mencoba mengatasi kelemahan yang dimiliki algoritma IDA* dengan menyimpan simpul berdasarkan nilai probabilitas sehingga dapat mengurangi kemungkinan terjadinya kunjungan kembali ke simpul yang sama dan dapat mempercepat waktu eksekusi. Berkurangnya ekspansi simpul diharapkan akan mengurangi waktu pemrosesan dari algoritma IDA*.

Oleh karena itu, yang akan diteliti untuk mengukur efektifitas penggunaan algoritma IDA* SNC adalah sebagai berikut :

1. Jumlah ekspansi simpul untuk seluruh iterasi dari proses pencarian.
2. Waktu eksekusi untuk pencarian solusi.

3.2 Analisis *Game* Labirin

Analisis pada *game* labirin dilakukan untuk melihat keefektifan dalam hal waktu eksekusi dan jumlah ekspansi simpul, algoritma IDA* dan IDA* SNC akan diimplementasikan dalam *game* labirin. Dalam mengimplementasikan algoritma IDA* dan IDA* SNC akan dibangun sebuah prototype dengan gambaran *game* labirin adalah sebagai berikut :

1. *Game* terdiri dari 1 (satu) *stage* dalam ukuran 800 x 640.
2. Kerumitan labirin yang dapat dipilih yaitu *easy, medium,* atau *hard*.
3. Penerapan algoritma yang dapat dipilih yaitu algoritma IDA* atau IDA* SNC.
4. Nilai inputan parameter IDA SNC yaitu untuk M = 0 - 60, dan p = 0 - 1.
5. Terdiri 1 (satu) pemain karakter utama dan karakter musuh berjumlah 2 (dua) karakter yang posisinya telah ditentukan.
6. Pencarian jalan terpendek pada *game* labirin diasumsikan dimulai dari simpul awal (musuh) dan simpul tujuan (pemain) dalam keadaan diam, kemudian musuh yang telah terintegrasi algoritma IDA* dan IDA* SNC bergerak secara bersamaan, mengejar dan pada akhirnya menangkap pemain.

3.3 Analisis Masukan

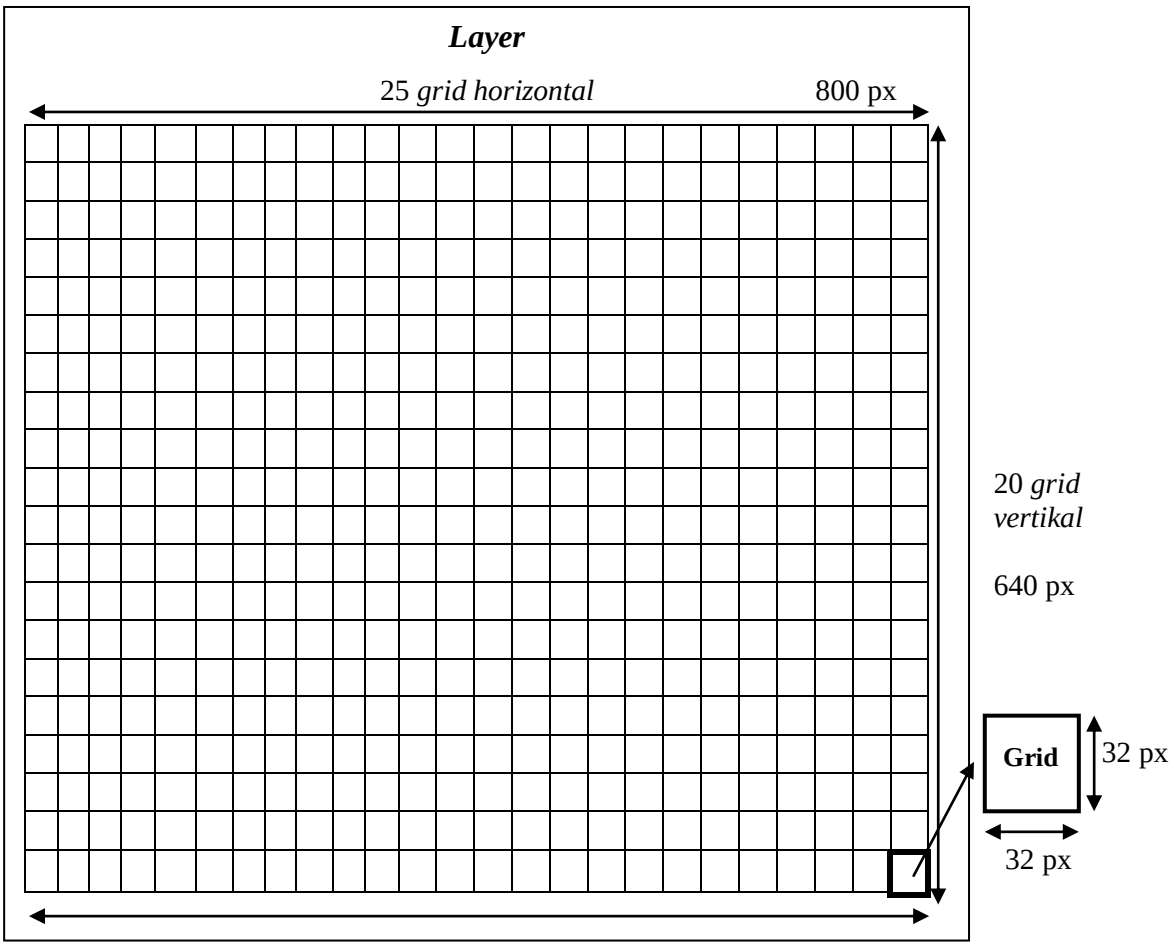
Analisis masukan menjelaskan data masukan yang diperlukan algoritma IDA* dan IDA* SNC dalam *game*.

3.3.1 Analisis Masukan Algoritma IDA*

Analisis data masukan yang dibutuhkan pada algoritma IDA* yaitu posisi awal pemain (*initial state*), posisi musuh (*goal state*). Posisi pemain dan musuh yang sudah ditentukan pada saat permainan dimulai, akan diproses untuk pencarian rute terpendek musuh agar dapat mengejar pemain. Selain itu, dalam pencarian algoritma IDA*, dibutuhkan pencarian untuk menghitung nilai $f(n_i)$. Nilai $f(n_i)$ dapat dilihat pada persamaan 2.1.

Sebelum menghitung menggunakan algoritma IDA*, akan dicari terlebih dahulu biaya (*cost*) sebenarnya yang terdapat pada jalur itu sendiri. *Layer game* labirin dibuat dengan skala *pixel* yang berukuran lebar 800 *pixel* dan tinggi 640 *pixel*. Satu *layer* tersebut disimpan beberapa *grid* (suatu kotak-kotak kecil) dengan

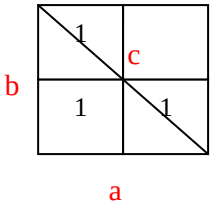
ukuran lebar 32 *pixel* dan tinggi 32 *pixel*, didapat 25 *grid horizontal* dan 20 *grid vertikal*. Jika dijumlahkan keseluruhan *grid*-nya menjadi 500 *grid*. Lihat Gambar 3.1.



Gambar 3.1 Keterangan ukuran *layer* dan *grid* (simpul) pada *game* labirin

Untuk memudahkan perhitungan maka ada beberapa nilai yang akan disederhanakan. Pertama nilai simpul (*grid*) yang tadinya 32x32 *pixel* diubah menjadi 1x1, artinya jarak simpul 1 ke simpul yang lain adalah 1. Jadi setiap simpul memiliki *cost* yang sama yaitu 1 secara *horizontal* maupun *vertikal*. Sedangkan untuk mencari biaya (*cost*) sebenarnya ketika gerakan diagonal, nilainya dicari menggunakan rumus *phytagoras*. Llihat Gambar 3.2.

$$c = \sqrt{a^2 + b^2}$$
$$= \sqrt{1^2 + 1^2}$$
$$= \sqrt{2} = 1.41$$



Gambar 3.2 Rumus *phytagoras* untuk mencari *cost* pada *game* labirin

Jadi untuk setiap perhitungan diagonalnya memiliki biaya (*cost*) 1.41. Sehingga dapat dilihat pada Gambar 3.3 merupakan koordinat *x* dan *y* dari biaya (*cost*) sebenarnya yang sudah disederhanakan. Koodinat *x* menyatakan kolom dan koordinat *y* menyatakan baris.

(0,0)	(1,0)	(2,0)	(3,0)	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	(24,0)
(0,1)																						(24,1)
(0,2)																						(24,2)
(0,3)																						(24,3)
(0,4)																						(24,4)
(0,5)																						(24,5)
.																						.
.																						.
.																						.
.																						.
.																						.
.																						.
.																						.
.																						.
.																						.
.																						.
.																						.
.																						.
.																						.
(0,19)	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	(24,19)

Gambar 3.3 Penyederhanaan nilai jarak *x* dan *y*

Setelah biaya (*cost*) sebenarnya telah diketahui, kemudian untuk menentukan nilai heuristik diantara dua jalur digunakan rumus *Manhattan Distance*, dapat dilihat pada persamaan 2.2.

3.3.2 Analisis Masukan Algoritma IDA* SNC

Data masukan yang dibutuhkan untuk pencarian algoritma IDA* SNC sama dengan data masukan algoritma IDA* ditambah dengan parameter tambahan yang harus diinputkan oleh user yaitu nilai *M* dan *p* [4]. Nilai *M* adalah banyak memori yang disediakan sebagai jumlah simpul yang dapat disimpan di memori tersebut. IDA* SNC mengalokasikan ruang memori yang dapat menyimpan *M* simpul. Alokasi ruang memori yang digunakan hanya untuk menyimpan nilai

heuristik saja, sehingga jumlah memori yang digunakan tidak terlalu signifikan. Karena pada akhirnya IDA* SNC juga sama seperti IDA*, yaitu akan menghapus simpul yang bukan jalurnya. Sedangkan *p* adalah probabilitas yang menentukan bagaimana sebuah simpul dalam *cache* setiap kali diekspansi. Nilai *p* yang diinputkan yaitu dalam rentang nilai 0 sampai 1 sesuai dengan nilai peluang. Untuk menentukan nilai *M* dan *p* yang efektif, maka dapat dilakukan dengan percobaan nilai inputannya.

Pemilihan untuk inputan nilai parameter dengan hasil jumlah ekspansi dan waktu pencarian yang lebih sedikit dapat meningkatkan performa eksekusi algoritma tersebut. Sehingga diharapkan kedua parameter tambahan ini dapat mempengaruhi algoritma terhadap cara kerja algoritma IDA* yaitu dalam hal jumlah ekspansi simpul sehingga dapat mengurangi kemungkinan terjadinya kunjungan kembali ke simpul yang sama dan waktu pencarian yang lebih cepat.

3.4 Analisis Algoritma

Pada bagian ini akan dijelaskan sebuah kasus dengan nilai-nilai parameter yang telah ditetapkan, kemudian diproses secara manual untuk menghasilkan *output*.

3.4.1 Analisis Algoritma IDA*

Untuk lebih memahami algoritma IDA*, bisa dilihat dari contoh kasus seperti terlihat pada Gambar 3.4. Diasumsikan dalam *game* terdapat 2 karakter yaitu pemain (P) dan musuh (E), selain itu juga ada tembok penghalang (t). Karakter P dan E digunakan untuk menunjukkan lokasi simpul awal (E) dan simpul tujuan (P). Sedangkan labirin yang digunakan untuk contoh kasus adalah bagian *medium* dari kerumitan labirin dan tidak semua labirin yang ditampilkan.

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	t	t	t	t	t	t	t	t	t	t	t	t	t
1	t				t						t		
2	t								P				t
3	t			t	t	t	t	t	t	t	t		t
4	t						t				E		t
5	t		t	t			t	t	t	t	t	t	t

Keterangan :

t Tembok/penghalang

E Simpul awal

P Simpul tujuan

Gambar 3.4 Kondisi awal pencarian IDA* pada *game* labirin

Proses pertama yang dilakukan musuh adalah mencari tahu jalan menuju pemain, kemudian mencari dan memutuskan langkah yang akan diambil. Apabila langkah menuju pemain terhalang tembok, maka ulangi pencarian dengan *f-limit* baru, sampai menemukan jalan terpendek menuju pemain. Untuk lebih jelas, berikut adalah langkah-langkah perhitungan dalam pencarian jalur menuju pemain. Rumus untuk perhitungan $f(n_i)$ dan heuristiknya dapat dilihat pada persamaan 2.1 dan 2.2.

1. Tentukan simpul awal dan simpul tujuan.

   Simpul awal (n_a) = (10,4)

   Simpul tujuan (n_t) = (8,2)

2. Hitung $f(n_a)$ simpul awal.

   g(10,4) = 0

   c(10,4;10,4) = 0

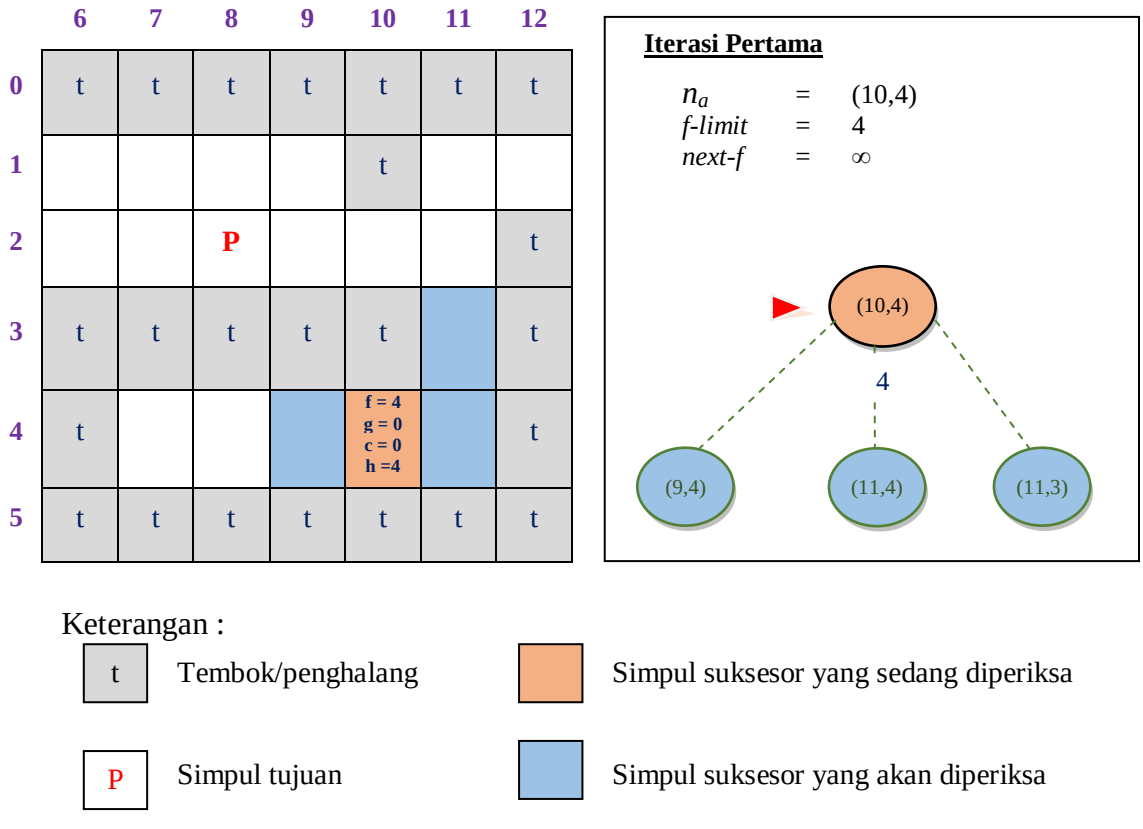
   h(10,4) = (abs|10-8| + abs|4-2|)

   = 2 + 2 = 4

   f(10,4) = g(10,4) + c(10,4;10,4) + h(10,4)

   = 0 + 0 + 4 = **4**

3. Tentukan *f-limit* dari $f(n_a)$ simpul awal.
   f-limit = $f(10,4)$ = **4**
4. Inisialisasi *next-f* = ∞.
5. Lakukan perulangan untuk setiap suksesor n_i dari simpul (10,4). Pencarian dilakukan berlawanan arah jarum jam mulai dari kiri atas, kiri, kiri bawah, bawah, kanan bawah, kanan, kanan atas dan atas. Simpul (10,4) mempunyai 3 simpul suksesor yaitu simpul (9,4), (11,4), dan (11,3). Maka lakukan perulangan sebanyak 3 kali. Lihat Gambar 3.5.



Keterangan :

t — Tembok/penghalang

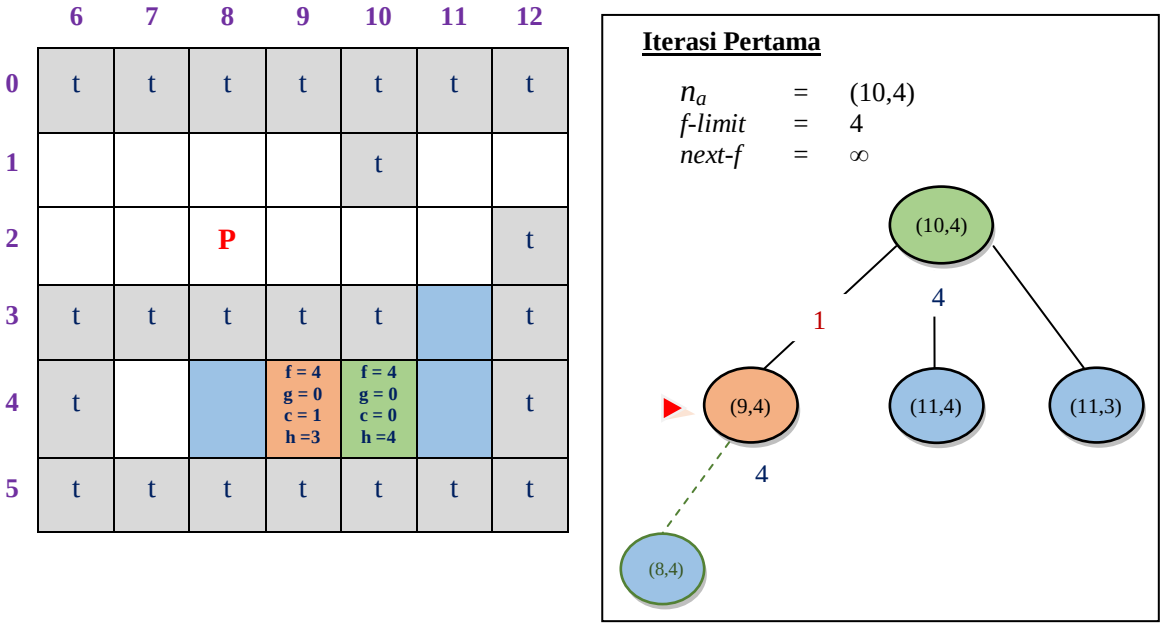
Simpul suksesor yang sedang diperiksa

P — Simpul tujuan

Simpul suksesor yang akan diperiksa

Gambar 3.5 Langkah 1-5 pencarian IDA* pada *game* labirin

6. Hitung nilai $f(n)$ suksesor pertama dari simpul (10,4) yaitu simpul (9,4), diperoleh nilai $f(9,4) \leq$ *f-limit* yaitu **4**, oleh karena itu lakukan perulangan untuk mengekspansi simpul suksesor dari simpul (9,4). Lihat Gambar 3.6.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

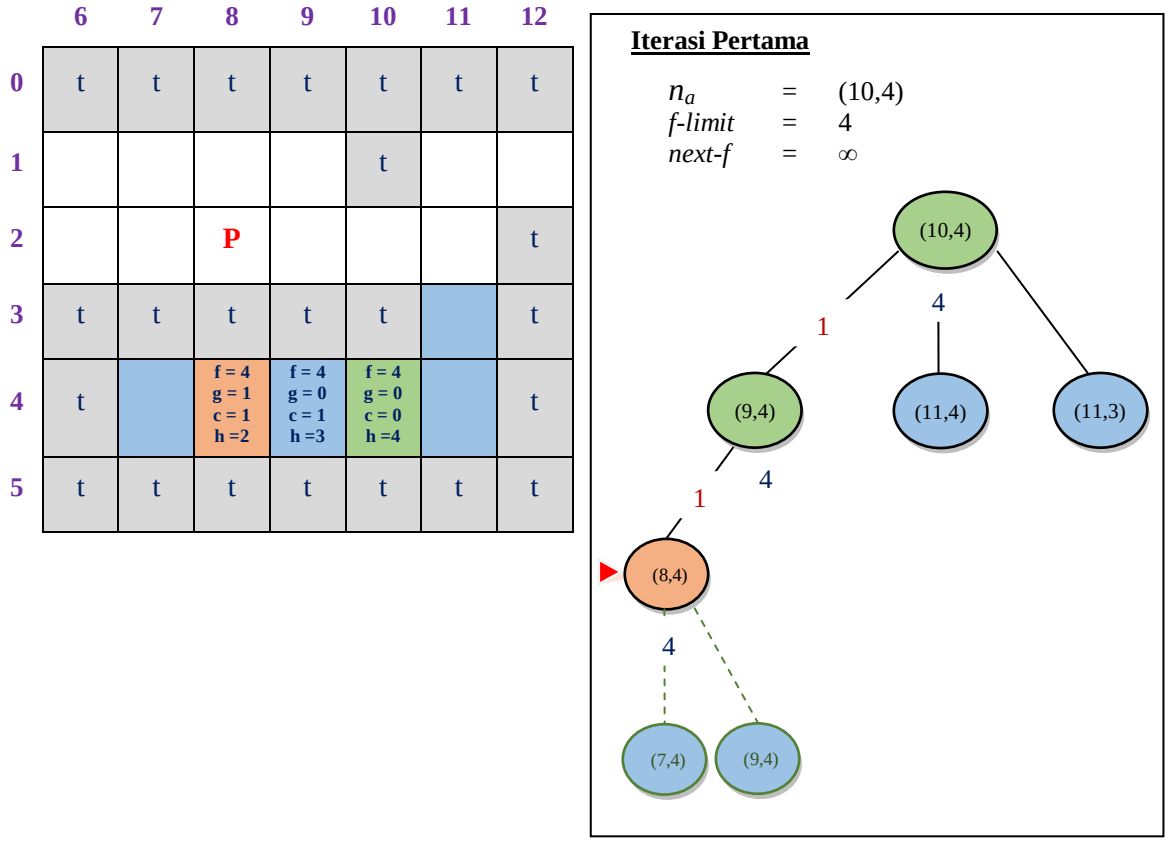
P — Simpul tujuan

Simpul suksesor yang akan diperiksa

Simpul suksesor yang sedang diperiksa

Gambar 3.6 Langkah 6 pencarian IDA* pada *game* labirin

7. Diperoleh simpul (8,4) sebagai suksesor dari simpul (9,4) dengan nilai *f*(8,4) = **4**. Karena nilai *f*(8,4) masih lebih kecil sama dengan nilai *f-limit*, maka ekspansi simpul (8,4). Simpul (8,4) mempunyai 2 suksesor yaitu simpul (7,4) dan (9,4). Lihat Gambar 3.7.

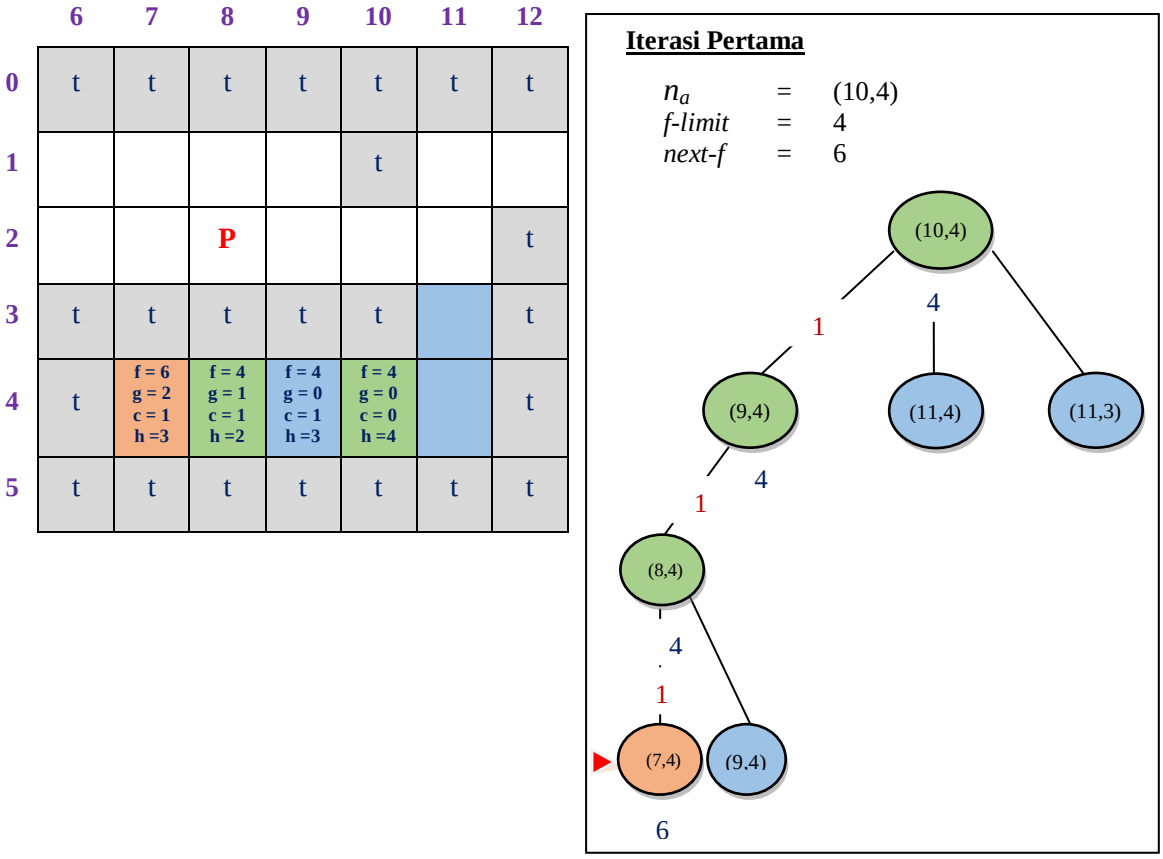


Keterangan :

t	Tembok/penghalang		Simpul suksesor yang sudah diperiksa
P	Simpul tujuan		Simpul suksesor yang diekspansi
	Simpul suksesor yang sedang diperiksa		

Gambar 3.7 Langkah 7 pencarian IDA* pada *game* labirin

8. Periksa simpul suksesor pertama dari simpul (8,4) yaitu simpul (7,4). Diperoleh nilai *f*(7,4) = **6**. Namun karena simpul (7,4) bukan simpul tujuan dan nilai *f*(7,4) > *f-limit*, maka simpul (7,4) tidak diekspansi tetapi jadikan nilai *f*(7,4) menjadi nilai *next-f*. Lihat Gambar 3.8.

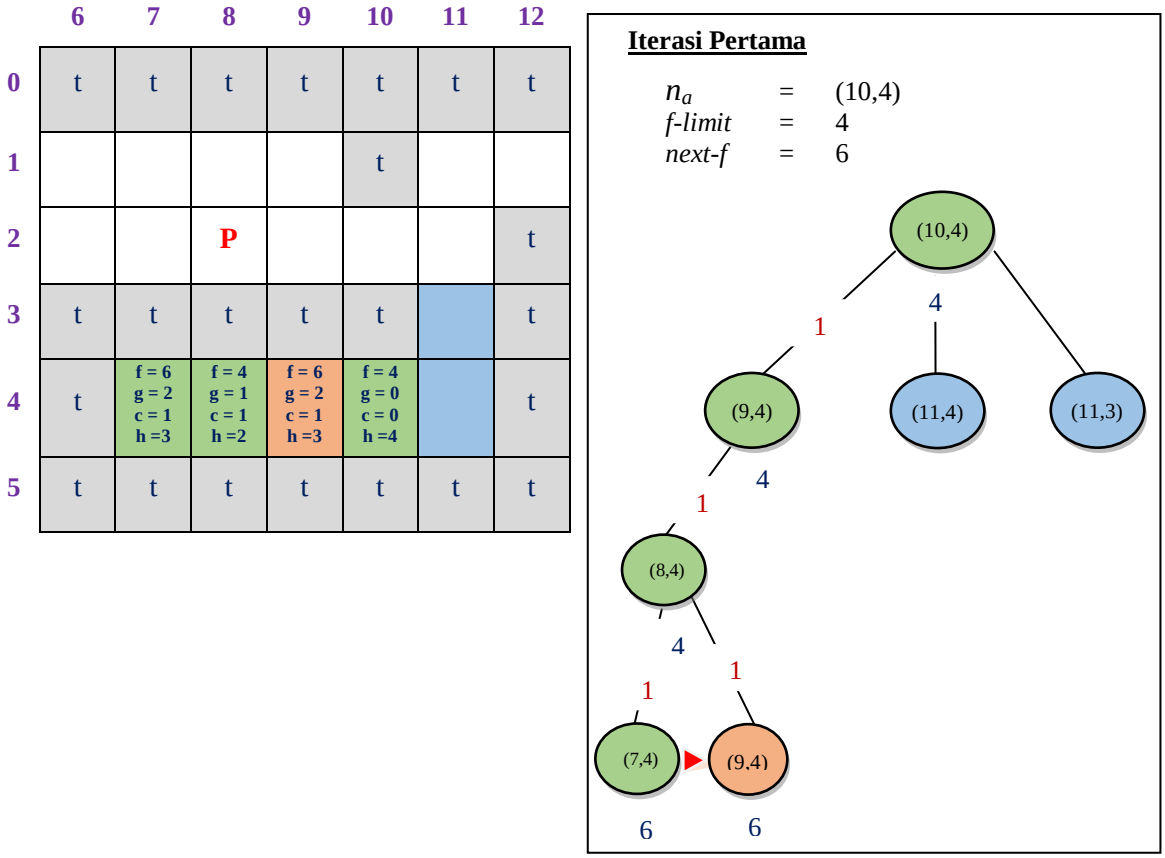


Keterangan :

t	Tembok/penghalang		Simpul suksesor yang sudah diperiksa
P	Simpul tujuan		Simpul suksesor yang diekspansi
	Simpul suksesor yang sedang diperiksa		

Gambar 3.8 Langkah 8 pencarian IDA* pada *game* labirin

9. Periksa simpul suksesor kedua dari simpul (8,4) yaitu simpul (9,4). Diperoleh nilai *f*(9,4) = **6**. Karena nilai *f*(9,4) lebih besar dari nilai *f-limit*, maka simpul (9,4) juga tidak diekspansi dan nilai *next-f* tetap karena nilai *f*(9,4) tidak lebih kecil dari nilai *next-f*. Lihat Gambar 3.9.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

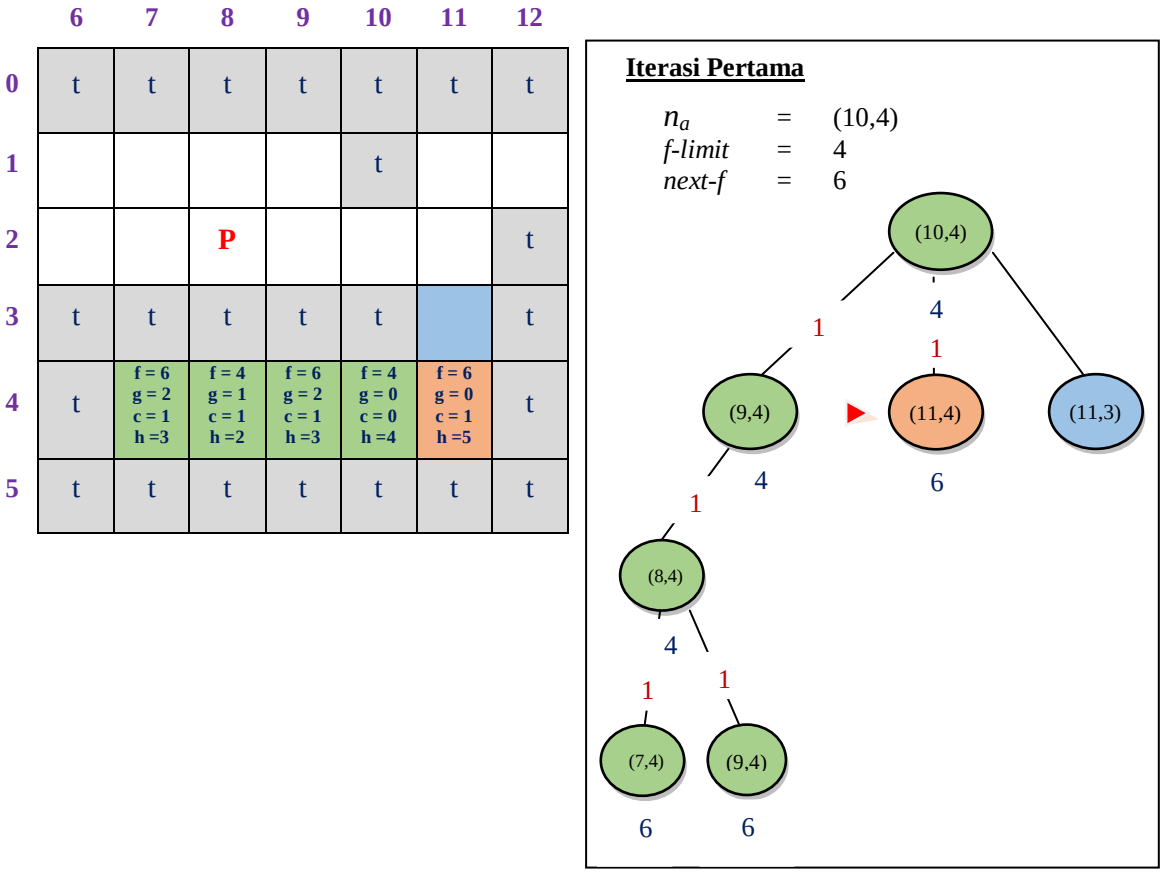
P — Simpul tujuan

Simpul suksesor diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.9 Langkah 9 pencarian IDA* pada *game* labirin

10. Karena simpul tujuan belum ditemukan, maka pencarian dilanjutkan ke simpul suksesor berikutnya dari simpul (10,4) yaitu simpul (11,4) dan diperoleh dengan nilai *f*(11,4) = **6**. Karena nilai *f*(11,4) lebih besar dari nilai *f-limit*, maka simpul (11,4) juga tidak diekspansi dan nilai *next-f* tetap karena nilai *f*(11,4) tidak lebih kecil dari nilai *next-f*. Lihat Gambar 3.10.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

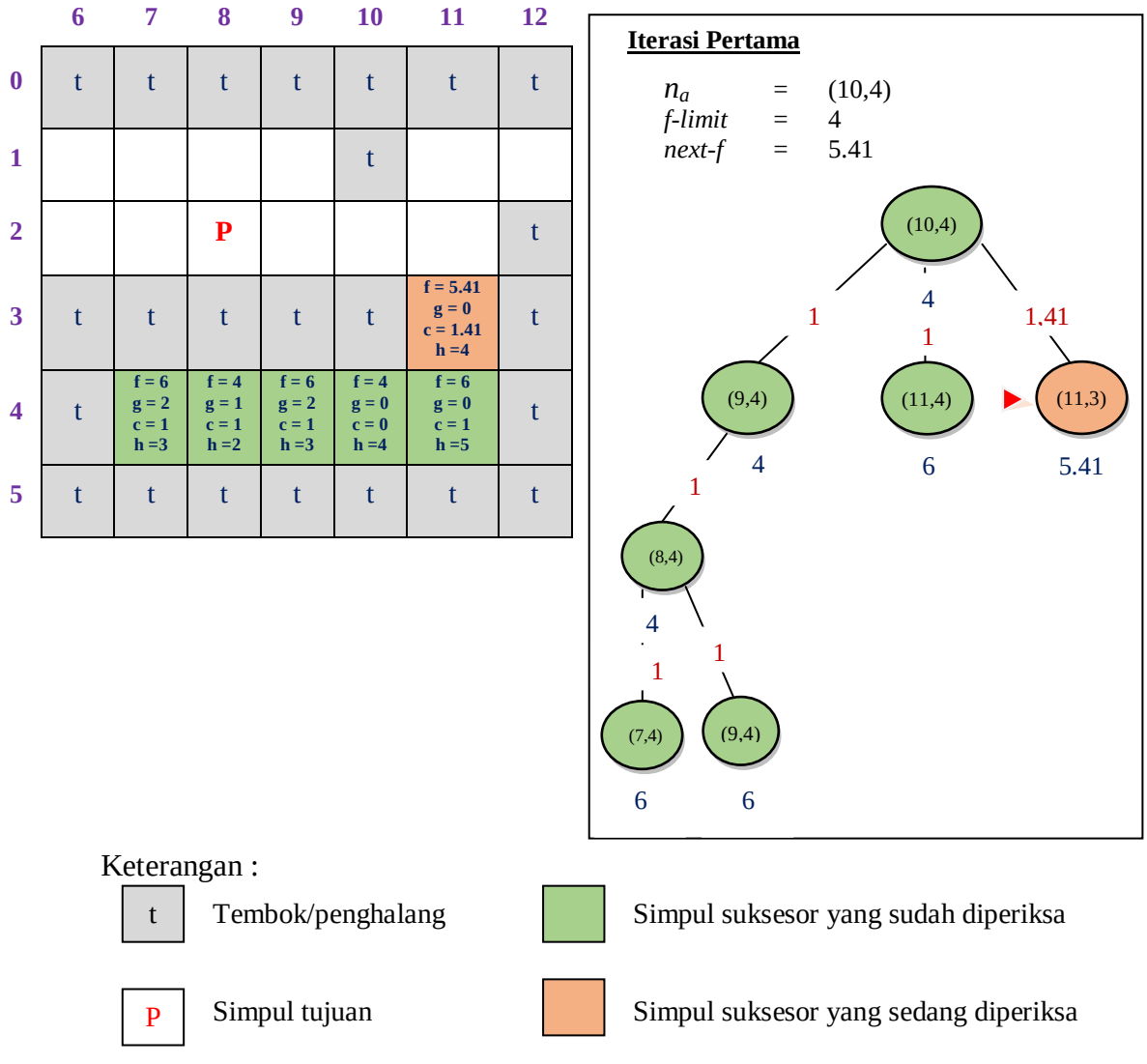
P — Simpul tujuan

Simpul suksesor yang diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.10 Langkah 10 pencarian IDA* pada *game* labirin

11. Karena simpul tujuan masih belum ditemukan, maka pencarian dilanjutkan ke simpul suksesor (11,3). Pada simpul suksesor (11,3) diperoleh nilai *f*(11,3) = **5.41**. Namun karena simpul (11,3) bukan simpul tujuan dan nilai *f*(11,3) > *f-limit*, maka simpul (11,3) tidak diekspansi tetapi jadikan nilai *f*(11,3) menjadi nilai *next-f* karena nilai *f*(11,3) < nilai *next-f*. Lihat Gambar 3.11.



Keterangan :

t — Tembok/penghalang

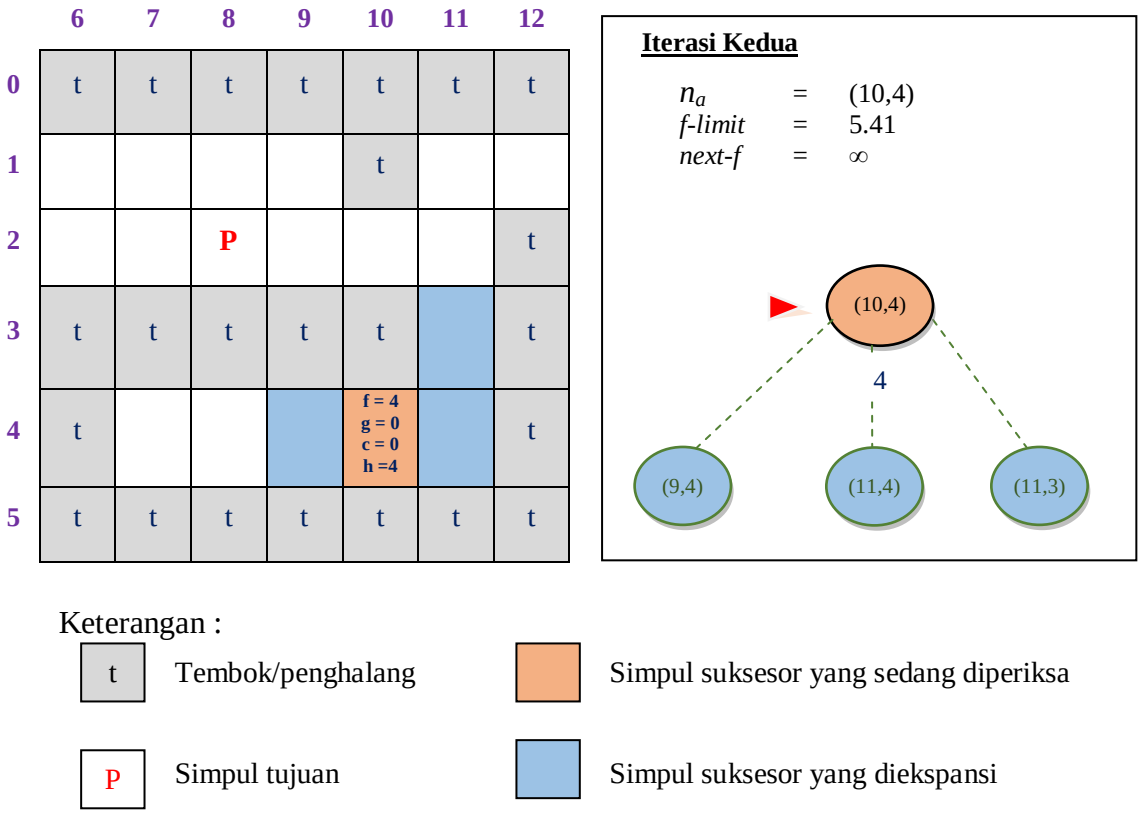
Simpul suksesor yang sudah diperiksa

P — Simpul tujuan

Simpul suksesor yang sedang diperiksa

Gambar 3.11 Langkah 11 pencarian IDA* pada *game* labirin

12. Set nilai *next-f* menjadi nilai *f-limit* baru dan nilai *next-f* menjadi ∞.
    f-limit = **5.41** ; *next-f* = ∞
13. Pada iterasi pertama, semua simpul sudah dibangkitkan dan diperiksa, namun belum menemukan solusi. Oleh karena itu, pencarian dimulai dari status awal lagi dengan nilai *f-limit* sekarang yaitu **5.41**. Lakukan pencarian seperti iterasi pertama. Ekspansi simpul (10,4) dengan lakukan perulangan untuk setiap suksesor n_i dari simpul (10,4). Simpul (10,4) mempunyai 3 simpul suksesor yaitu simpul (9,4), (11,4), dan (11,3). Maka lakukan perulangan sebanyak 3 kali. Lihat Gambar 3.12.



Keterangan :

t — Tembok/penghalang

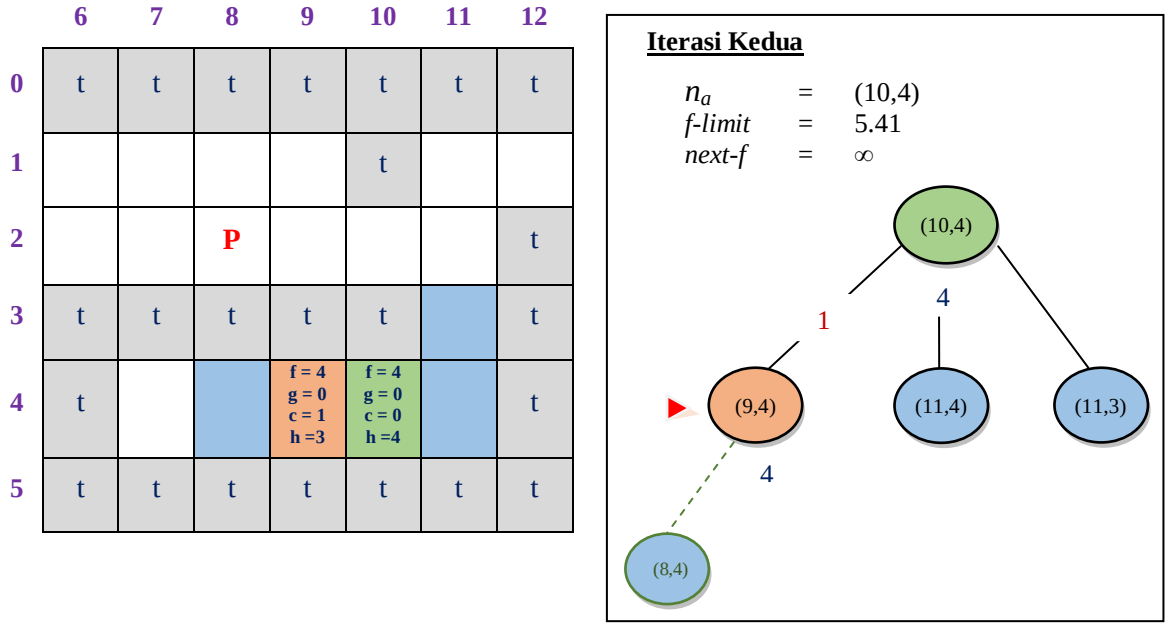
P — Simpul tujuan

Simpul suksesor yang sedang diperiksa

Simpul suksesor yang diekspansi

Gambar 3.12 Langkah 12-13 pencarian IDA* pada *game* labirin

14. Hitung nilai $f(n)$ suksesor pertama, diperoleh nilai $f(9,4) \leq$ *f-limit* yaitu **4**, oleh karena itu lakukan perulangan untuk mengekspansi simpul suksesor dari simpul (9,4). Lihat Gambar 3.13.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

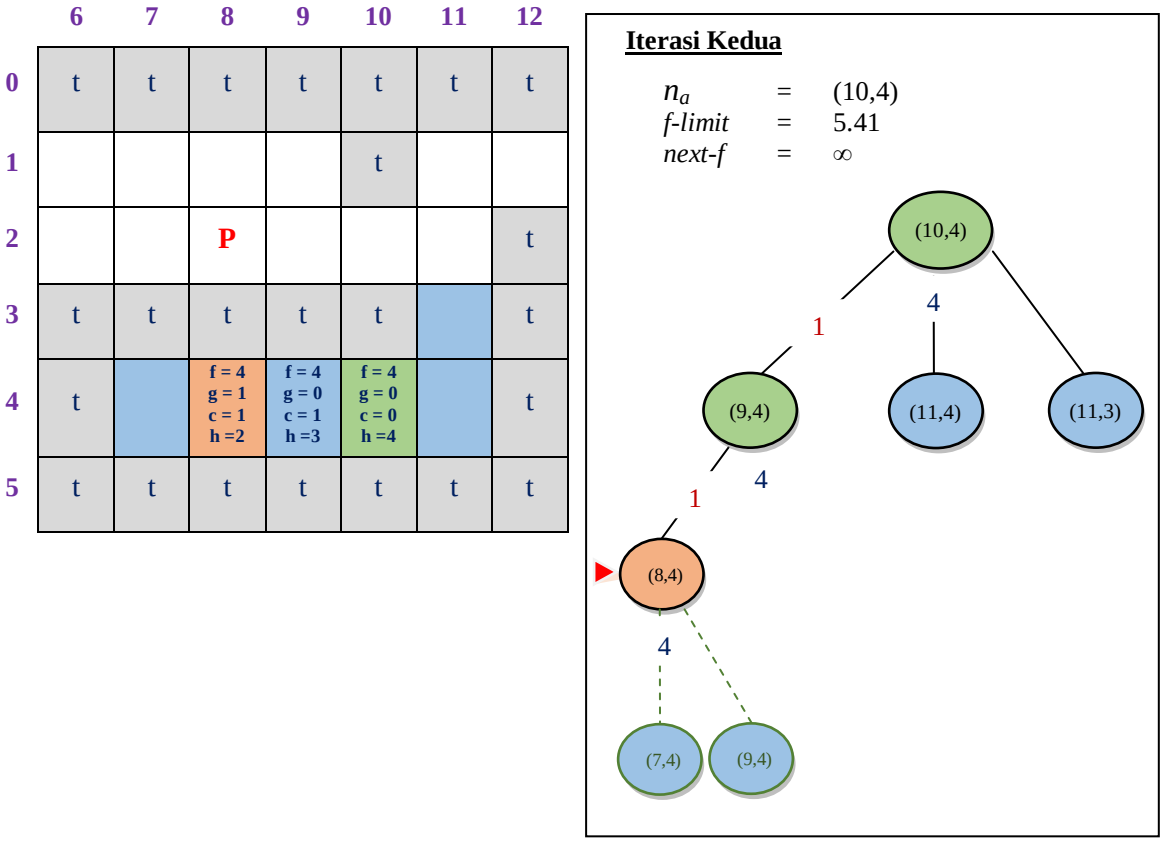
P — Simpul tujuan

Simpul suksesor yang diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.13 Langkah 14 pencarian IDA* pada *game* labirin

15. Diperoleh simpul (8,4) sebagai suksesor dari simpul (9,4) dengan nilai *f*(8,4) = **4**. Karena nilai *f*(8,4) masih lebih kecil sama dengan nilai *f-limit*, maka ekspansi simpul (8,4). Simpul (8,4) mempunyai 2 suksesor yaitu simpul (7,4) dan (9,4). Lihat Gambar 3.14.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

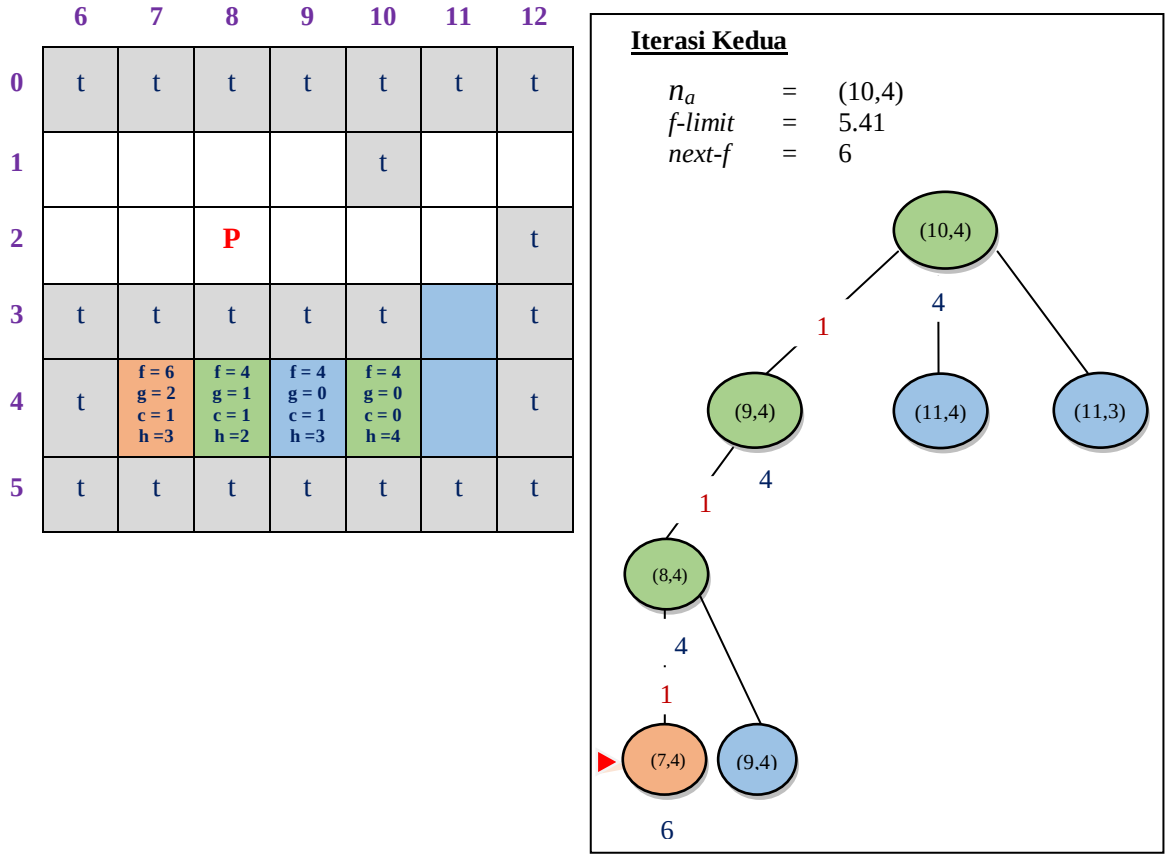
P — Simpul tujuan

Simpul suksesor yang diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.14 Langkah 15 pencarian IDA* pada *game* labirin

16. Periksa simpul suksesor pertama dari simpul (8,4) yaitu simpul (7,4). Diperoleh nilai $f(7,4)$ = **6**. Namun karena simpul (7,4) bukan simpul tujuan dan nilai $f(7,4) > f\text{-}limit$, maka simpul (7,4) tidak diekspansi tetapi jadikan nilai $f(7,4)$ menjadi nilai *next-f*. Lihat Gambar 3.15.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

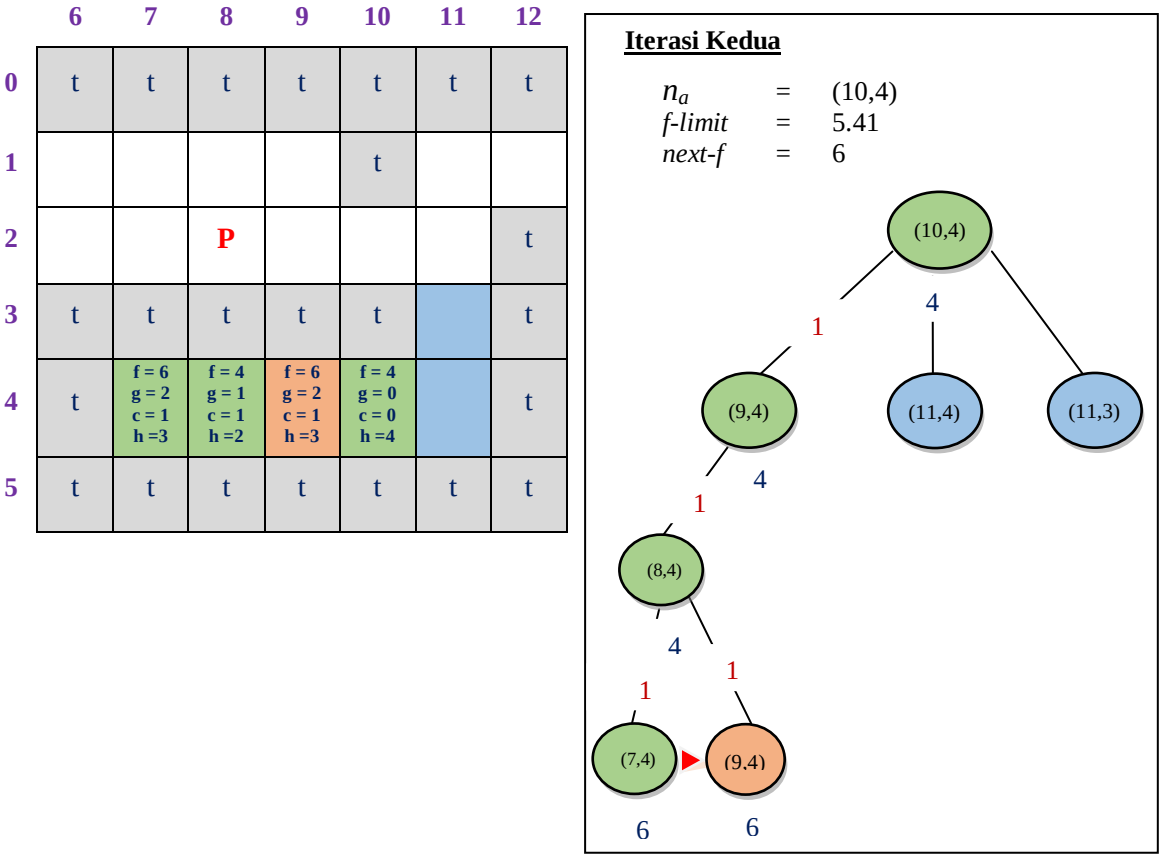
P — Simpul tujuan

Simpul suksesor yang diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.15 Langkah 16 pencarian IDA* pada *game* labirin

17. Periksa simpul suksesor kedua dari simpul (8,4) yaitu simpul (9,4). Diperoleh nilai *f*(9,4) = **6**. Karena nilai *f*(9,4) lebih besar dari nilai *f-limit*, maka simpul (9,4) juga tidak diekspansi dan nilai *next-f* tetap karena nilai *f*(9,4) tidak lebih kecil dari nilai *next-f*. Lihat Gambar 3.16.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

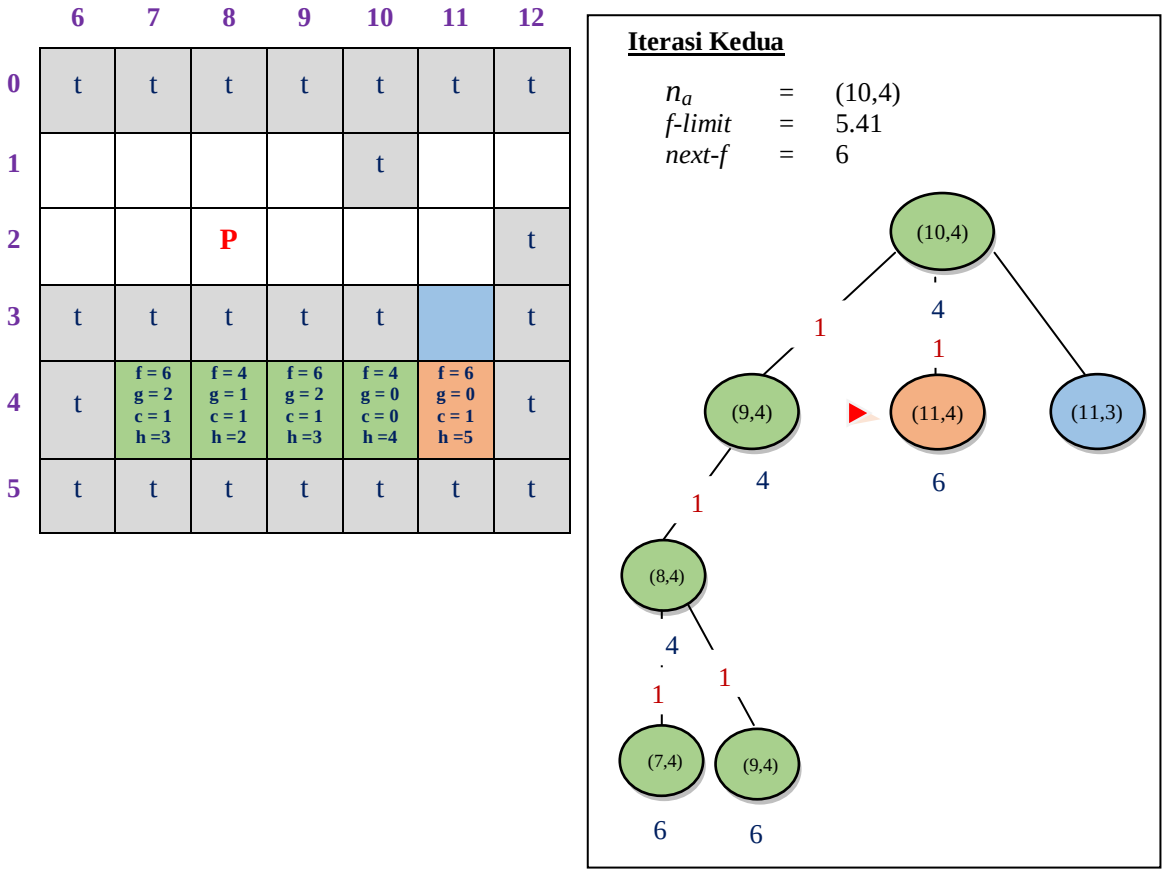
P — Simpul tujuan

Simpul suksesor yang diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.16 Langkah 17 pencarian IDA* pada *game* labirin

18. Karena simpul tujuan belum ditemukan, maka pencarian dilanjutkan ke simpul suksesor berikutnya dari simpul (10,4) yaitu simpul (11,4) dan diperoleh dengan nilai *f*(11,4) = **6**. Karena nilai *f*(11,4) lebih besar dari nilai *f-limit*, maka simpul (11,4) juga tidak diekspansi dan nilai *next-f* tetap karena nilai *f*(11,4) tidak lebih kecil dari nilai *next-f*. Lihat Gambar 3.17.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

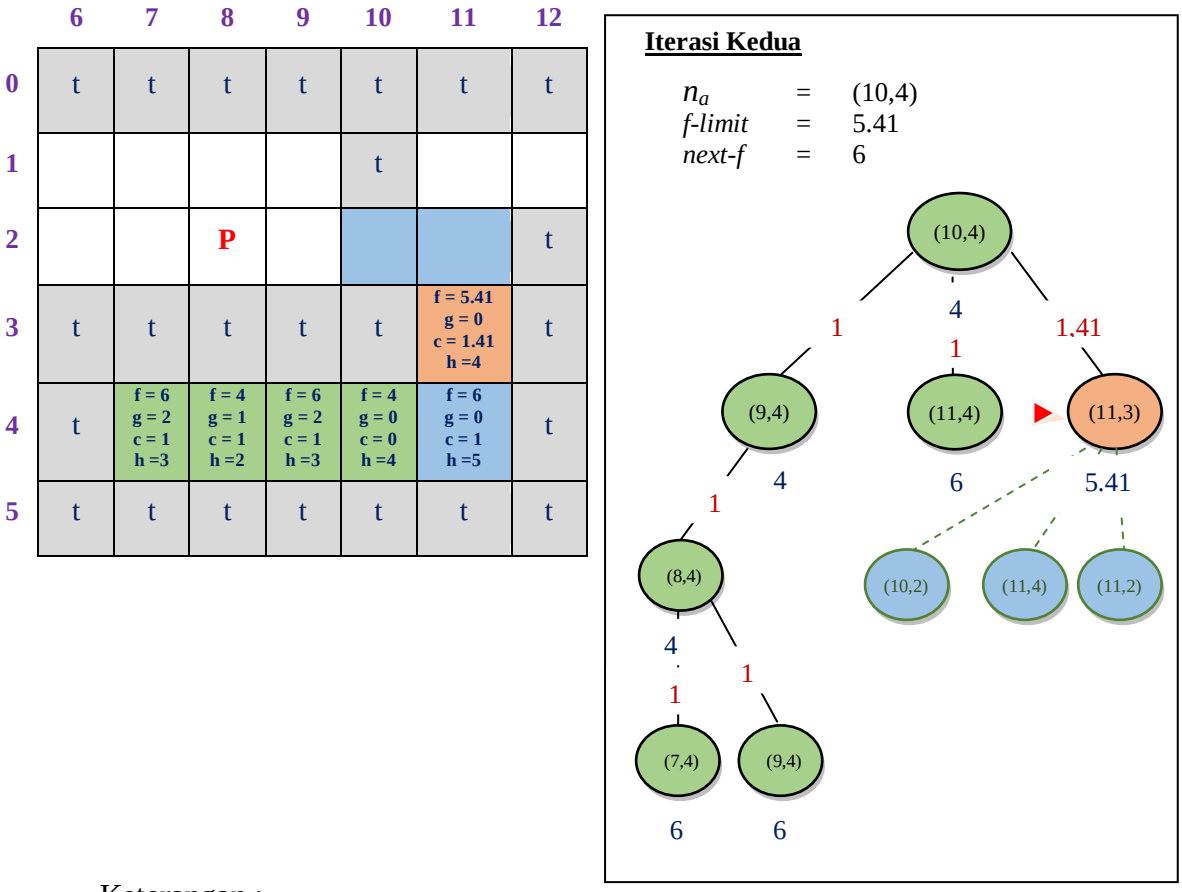
P — Simpul tujuan

Simpul suksesor yang diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.17 Langkah 18 pencarian IDA* pada *game* labirin

19. Karena simpul tujuan masih belum ditemukan, maka pencarian dilanjutkan ke simpul suksesor (11,3). Pada suksesor ke-3 dari simpul (10,4) diperoleh nilai $f(11,3) \leq$ *f-limit* yaitu **5.41**, maka ekspansi simpul suksessor dari simpul (11,3) untuk mencari suksesor selanjutnya. Simpul (11,3) memiliki 3 suksesor yaitu simpul (10,2), simpul (11,4) dan simpul (11,2). Lihat Gambar 3.18.



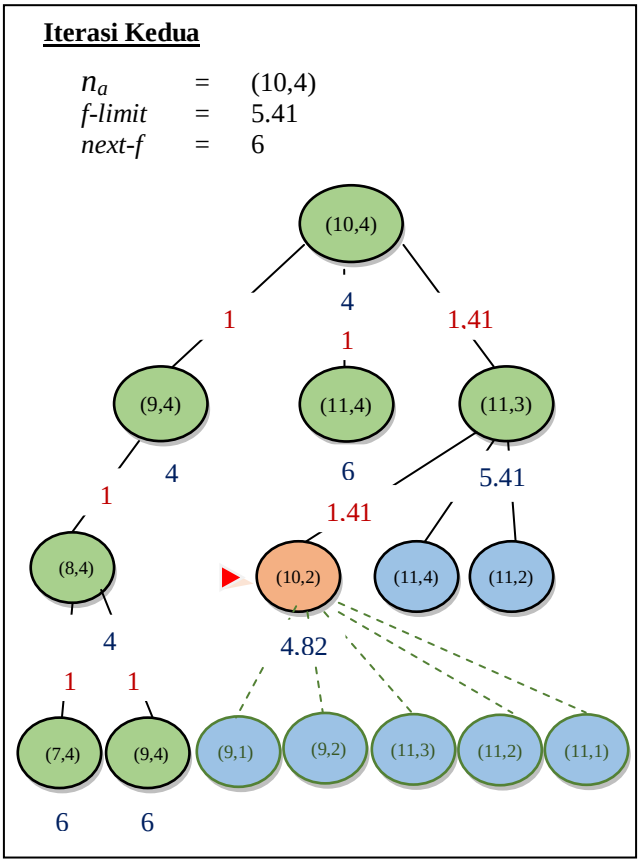
Keterangan :

t	Tembok/penghalang
	Simpul suksesor yang sudah diperiksa
P	Simpul tujuan
	Simpul suksesor yang diekspansi
	Simpul suksesor yang sedang diperiksa

Gambar 3.18 Langkah 19 pencarian IDA* pada *game* labirin

20. Hitung nilai $f(n_i)$ suksesor pertama dari simpul (11,3) yaitu simpul (10,2). Diperoleh nilai $f(10,2)$ = 4.82 yang artinya masih lebih dari *f-limit*, maka ekspansi simpul (10,2) dengan melakukan perulangan perhitungan $f(n_i)$ untuk setiap suksesor n_i dari simpul (10,2). Simpul (10,2) memiliki 5 simpul suksesor yaitu simpul (9,1), simpul (9,2), simpul (11,3), simpul (11,2) dan simpul (11,1). Lihat Gambar 3.19.

	6	7	8	9	10	11	12
0	t	t	t	t	t	t	t
1					t		
2			P		f = 4.82 g = 1.41 c = 1.41 h =2		t
3	t	t	t	t	t	f = 5.41 g = 0 c = 1.41 h =4	t
4	t	f = 6 g = 2 c = 1 h =3	f = 4 g = 1 c = 1 h =2	f = 6 g = 2 c = 1 h =3	f = 4 g = 0 c = 0 h =4	f = 6 g = 0 c = 1 h =5	t
5	t	t	t	t	t	t	t

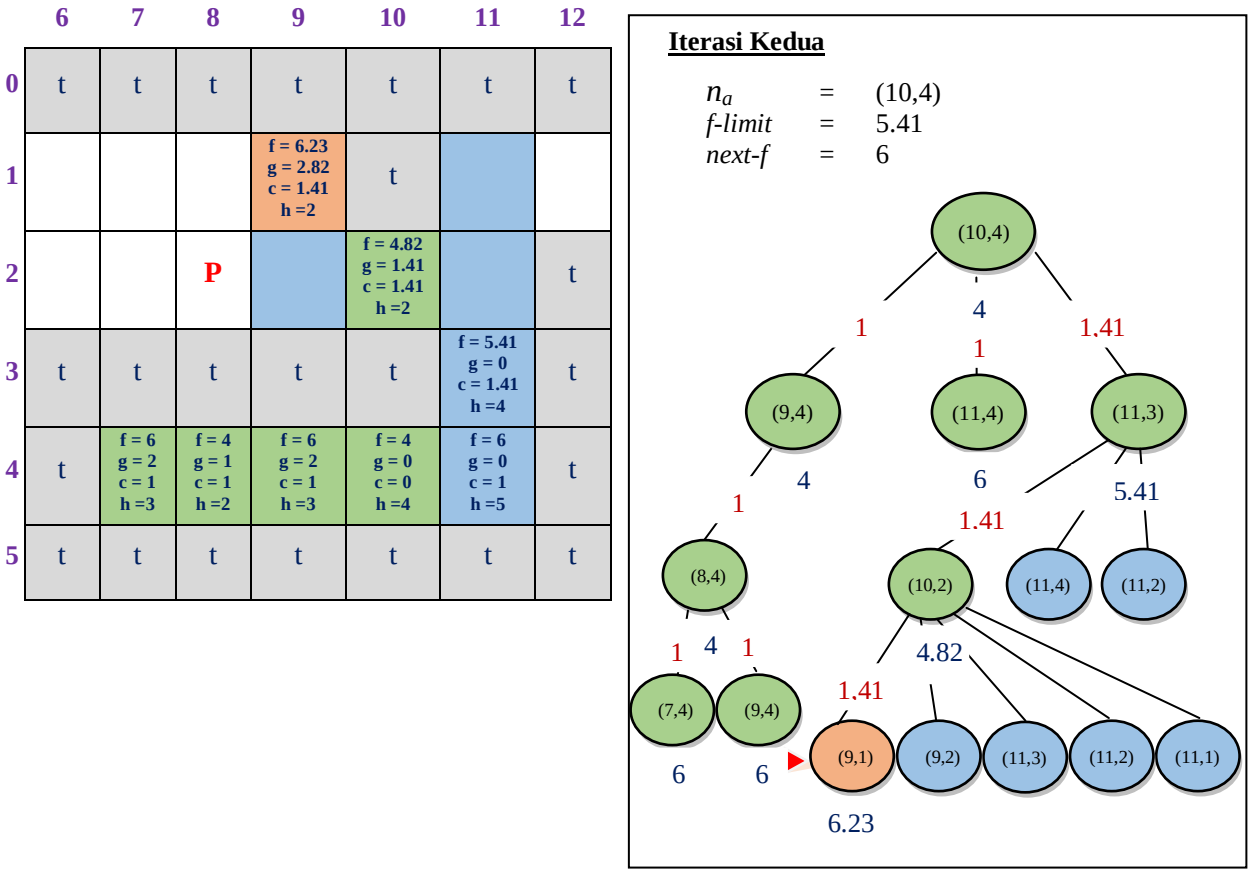


Keterangan :

t	Tembok/penghalang		Simpul suksesor yang sudah diperiksa
P	Simpul tujuan		Simpul suksesor yang diekspansi
	Simpul suksesor yang sedang diperiksa		

Gambar 3.19 Langkah 20 pencarian IDA* pada *game* labirin

21. Hitung nilai $f(n_i)$ suksesor pertama dari simpul (10,2). Simpul (9,1) adalah simpul suksesor pertama yang diperiksa dan hitung nilai $f(9,1)$. Diperoleh nilai $f(9,1)$ = 6.23. Namun karena nilai $f(9,1)$ lebih besar dari nilai *f-limit*, maka simpul (9,1) tidak diekspansi dan nilai *next-f* tetap karena nilai $f(9,1)$ tidak lebih kecil dari nilai *next-f*. Lihat Gambar 3.20.

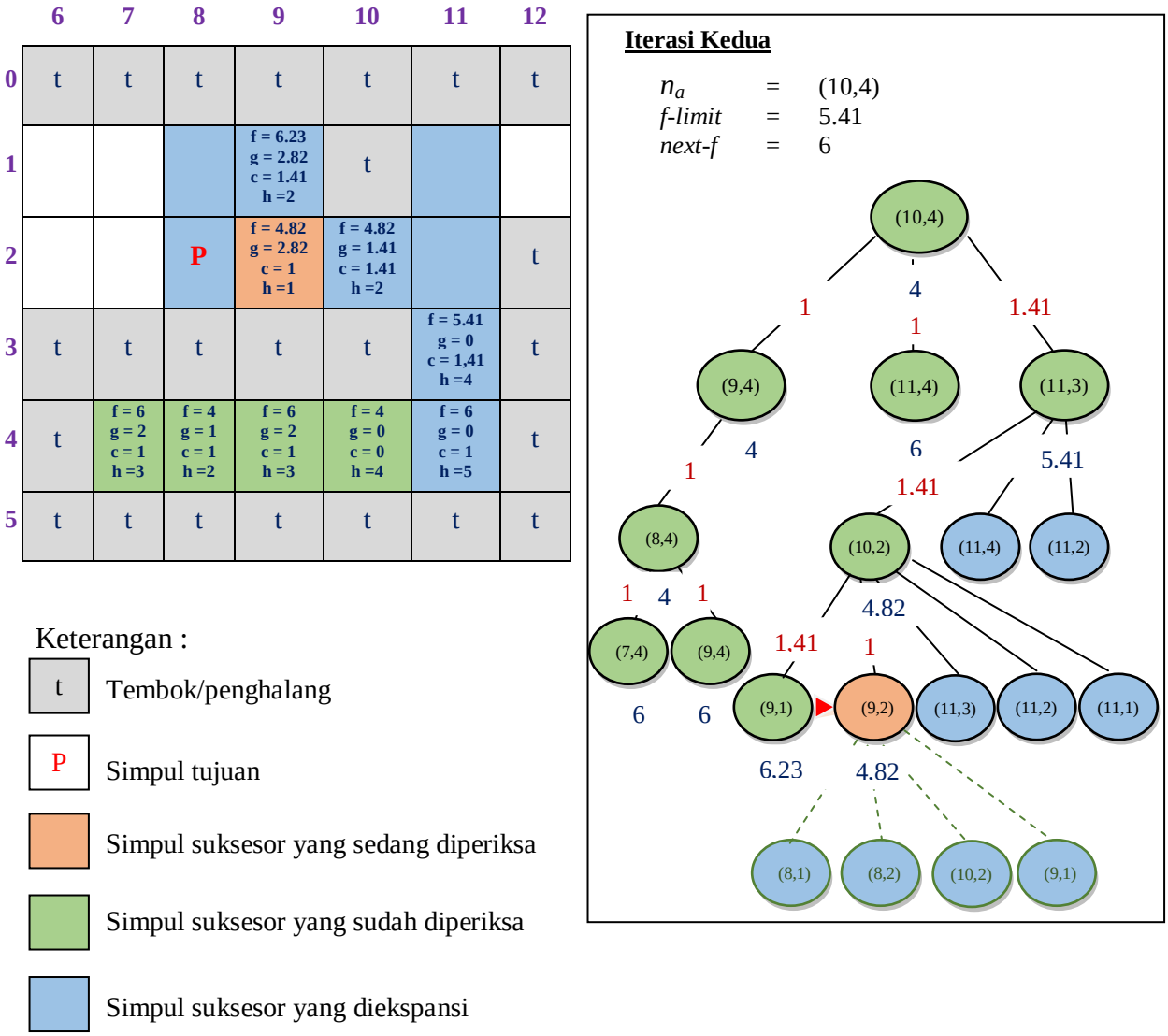


Keterangan :

t — Tembok/penghalang	Simpul suksesor yang sudah diperiksa
P — Simpul tujuan	Simpul suksesor yang diekspansi
Simpul suksesor yang sedang diperiksa	

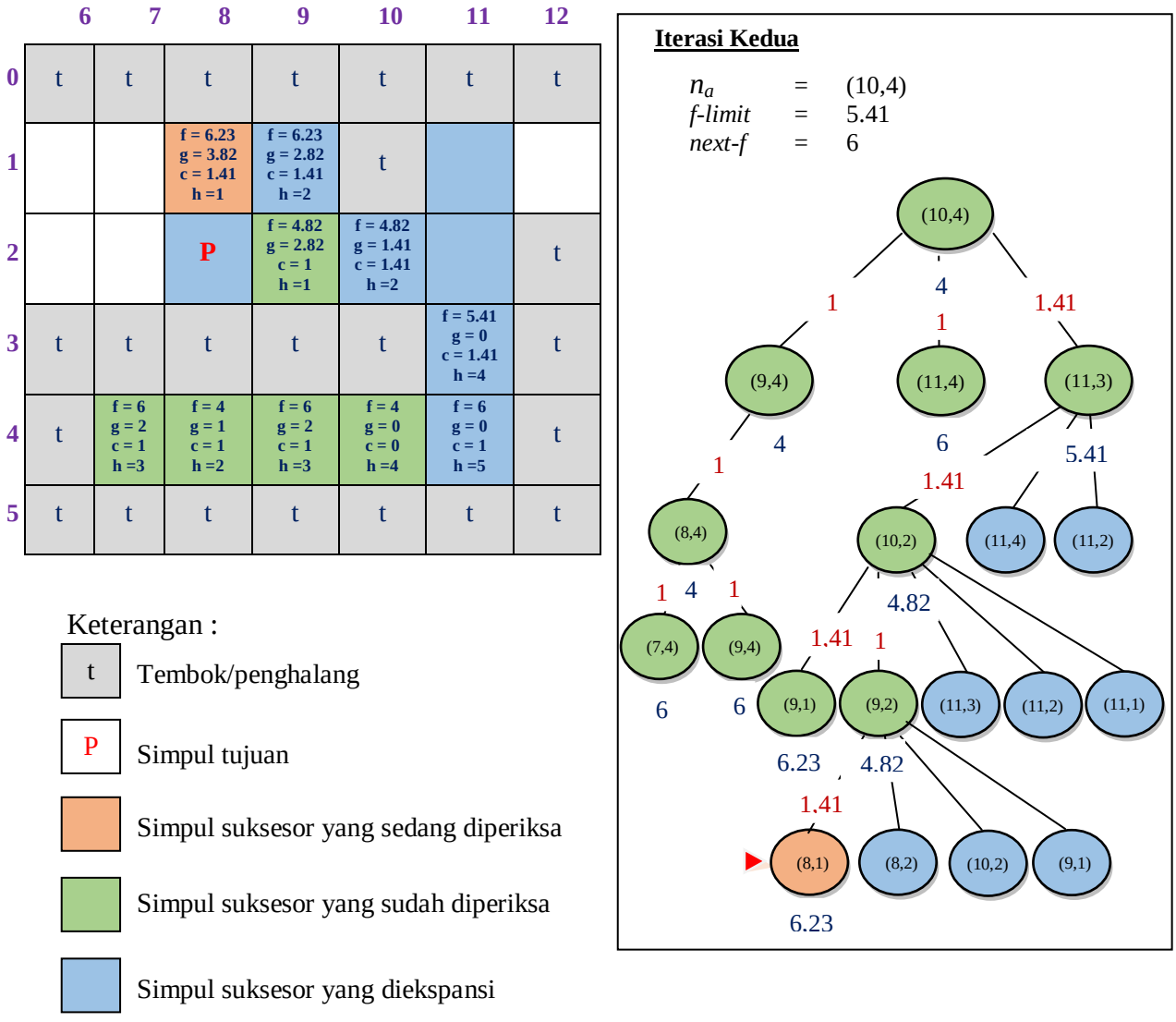
Gambar 3.20 Langkah 21 pencarian IDA* pada *game* labirin

22. Namun karena simpul (9,1) memiliki nilai *f*(9,1) lebih besar dari nilai *f-limit*, maka dilakukan ke suksesor berikutnya yaitu simpul (9,2). Ternyata simpul (9,2) memiliki nilai *f*(9,2) = 4.82 yang nilainya masih lebih kecil sama dengan *f-limit*, maka ekspansi simpul (9,2) dengan melakukan perulangan perhitungan $f(n_i)$ untuk setiap suksesor n_i dari simpul (9,2). Simpul (9,2) memiliki 4 simpul suksesor yaitu simpul (8,1), simpul (8,2), simpul (10,2) dan simpul (9,1). Lihat Gambar 3.21.



Gambar 3.21 Langkah 22 pencarian IDA* pada *game* labirin

23. Hitung nilai $f(n_i)$ suksesor pertama dari simpul (9,2). Simpul (8,1) adalah simpul yang pertama kali diperiksa dan diperoleh nilai $f(8,1) = 6.23$. Namun karena nilai $f(8,1)$ lebih besar dari nilai *f-limit*, maka simpul (8,1) tidak diekspansi dan nilai *next-f* tetap karena nilai $f(8,1)$ tidak lebih kecil dari nilai *next-f*. Lihat Gambar 3.22.



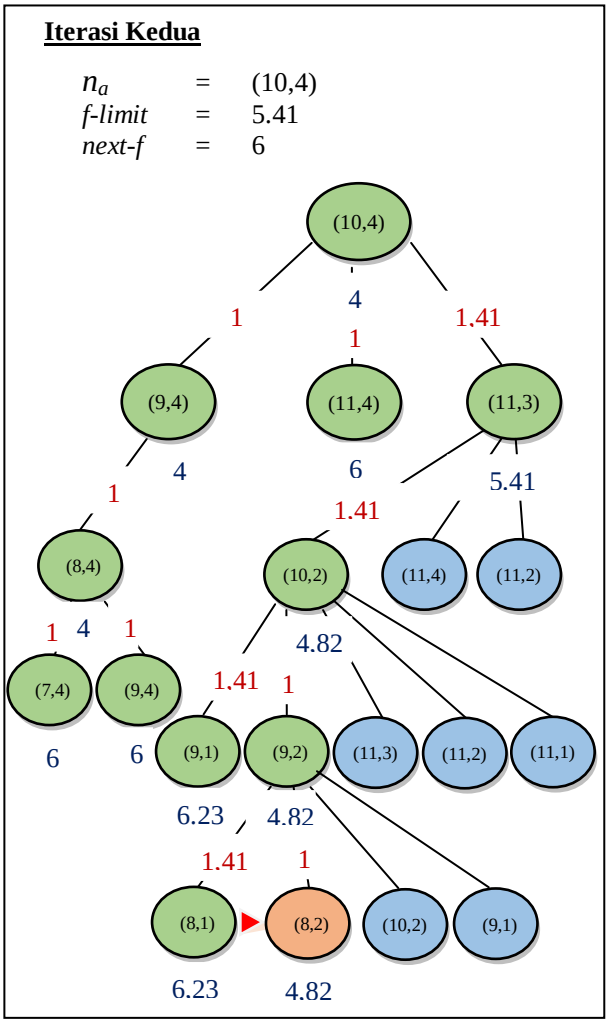
Gambar 3.22 Langkah 23 pencarian IDA* pada *game* labirin

24. Namun karena simpul (8,1) memiliki nilai *f*(8,1) lebih besar dari nilai *f-limit*, maka pencarian dilakukan ke suksesor berikutnya yaitu simpul (8,2). Ternyata simpul (8,2) merupakan simpul tujuan dan memiliki nilai *f*(8,2) = 4.82 yang artinya memiliki nilai yang lebih kecil sama dengan *f-limit*. Maka pecarian berhenti dengan menemukan simpul (8,2) sebagai simpul tujuan. Lihat Gambar 3.23.

	6	7	8	9	10	11	12
0	t	t	t	t	t	t	t
1			f = 6.23 g = 3.82 c = 1.41 h =1	f = 6.23 g = 2.82 c = 1.41 h =2	t		
2			f = 4.82 g = 3,82 c = 1 h =0	f = 4.82 g = 2.82 c = 1 h =1	f = 4.82 g = 1.41 c = 1.41 h =2		t
3	t	t	t	t	t	f = 5.41 g = 0 c = 1.41 h =4	t
4	t	f = 6 g = 2 c = 1 h =3	f = 4 g = 1 c = 1 h =2	f = 6 g = 2 c = 1 h =3	f = 4 g = 0 c = 0 h =4	f = 6 g = 0 c = 1 h =5	t
5	t	t	t	t	t	t	t

Keterangan :

- t — Tembok/penghalang
- P — Simpul tujuan
- Simpul suksesor yang sedang diperiksa
- Simpul suksesor yang sudah diperiksa
- Simpul suksesor yang diekspansi



Gambar 3.23 Langkah 24 pencarian IDA* pada *game* labirin

Pencarian IDA* pada kasus *game* labirin ini memeriksa simpul yang diekspansi sebanyak 19 simpul, dengan 7 simpul pada iterasi pertama dan pada iterasi kedua berjumlah 12 simpul. Karakteristik IDA* akan menghapus satu jalur yang dipilihnya dan menghapus jalur yang tidak dipilihnya. Sehingga hasil akhir pencarian jalur dari simpul awal (E) menuju simpul tujuan (P) dapat dilihat pada Gambar 3.24.

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	t	t	t	t	t	t	t	t	t	t	t	t	t
1	t				t						t		
2	t								P				t
3	t			t	t	t	t	t	t	t	t		t
4	t						t				E		t
5	t		t	t			t	t	t	t	t	t	t

Keterangan :

t — Tembok/penghalang

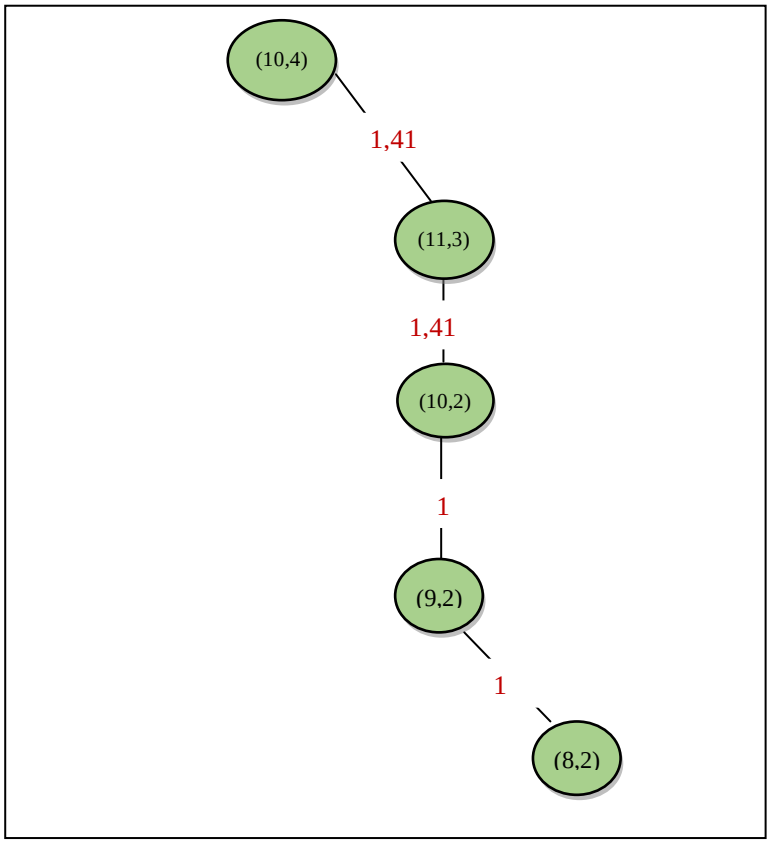
Simpul yang dipilih

E — Simpul awal

P — Simpul tujuan

Gambar 3.24 Hasil pencarian IDA* pada *game* labirin

Berdasarkan hasil keseluruhan pencarian jalur dari simpul awal (E) menuju simpul tujuan (P), maka diperoleh jalur terpendek yaitu **(10,4) – (11,3) – (10,2) – (9,2) – (8,2)** dan dengan biaya minimum yaitu **4.82**. Adapun hasil akhir pencarian dalam bentuk pohon pencarian IDA* pada *game* labirin dapat dilihat pada Gambar 3.25.



Gambar 3.25 Pohon Hasil Akhir Pencarian IDA* pada *game* labirin

3.4.2 Analisis Algoritma IDA* SNC

Analisis untuk algoritma IDA* SNC seperti algoritma IDA. Cara kerja IDA* SNC menggunakan parameter tambahan berupa inputan awal yaitu *M* dan *p*. Pada contoh kasus *game* labirin diasumsikan nilai inputan awal $M = 2$ dan $p = 0{,}6$. Pada IDA* SNC simpul yang akan diekspansi akan di-*cache* dan di-*update* sebagai nilai heuristik yang baru, maka nilai $f(n_i)$ juga semakin bertambah sehingga tidak mengalami ekspansi yang berulang pada iterasi berikutnya. Nilai heuristik yang baru diperoleh dari nilai $c(n,n_i) + h(n_i)$ terkecil simpul suksesor.

Untuk penjelasan langkah-langkah IDA* SNC akan menggunakan contoh kasus pada *game* labirin. Cara perhitungan $f(n_i)$, *next-f*, dan *f-limit* sama caranya dengan pencarian IDA*. Berikut adalah langkah-langkah perhitungan dalam pencarian jalur menuju pemain menggunakan algoritma IDA* SNC.

1. nputkan nilai *M* dan *p*. Nilai *M* adalah banyak memori yang disediakan sebagai jumlah simpul yang dapat disimpan di memori. Sedangkan *p* adalah probabilitas untuk sebuah simpul dalam *cache* setiap kali diekspansi. Diasumsikan nilai inputan awal *M* = 2 dan *p* = 0.6.

   M : **2**

   p : **0.6**

2. Tentukan simpul awal dan simpul tujuan.

   Simpul awal (n_a) = (10,4)

   Simpul tujuan (n_t) = (8,2)

3. Hitung $f(n_a)$ simpul awal.

   g(10,4) = 0

   c(10,4;10,4) = 0

   h(10,4) = (abs|10-8| + abs|4-2|)

   = 2 + 2 = 4

   f(10,4) = g(10,4) + c(10,4;10,4) + h(10,4)

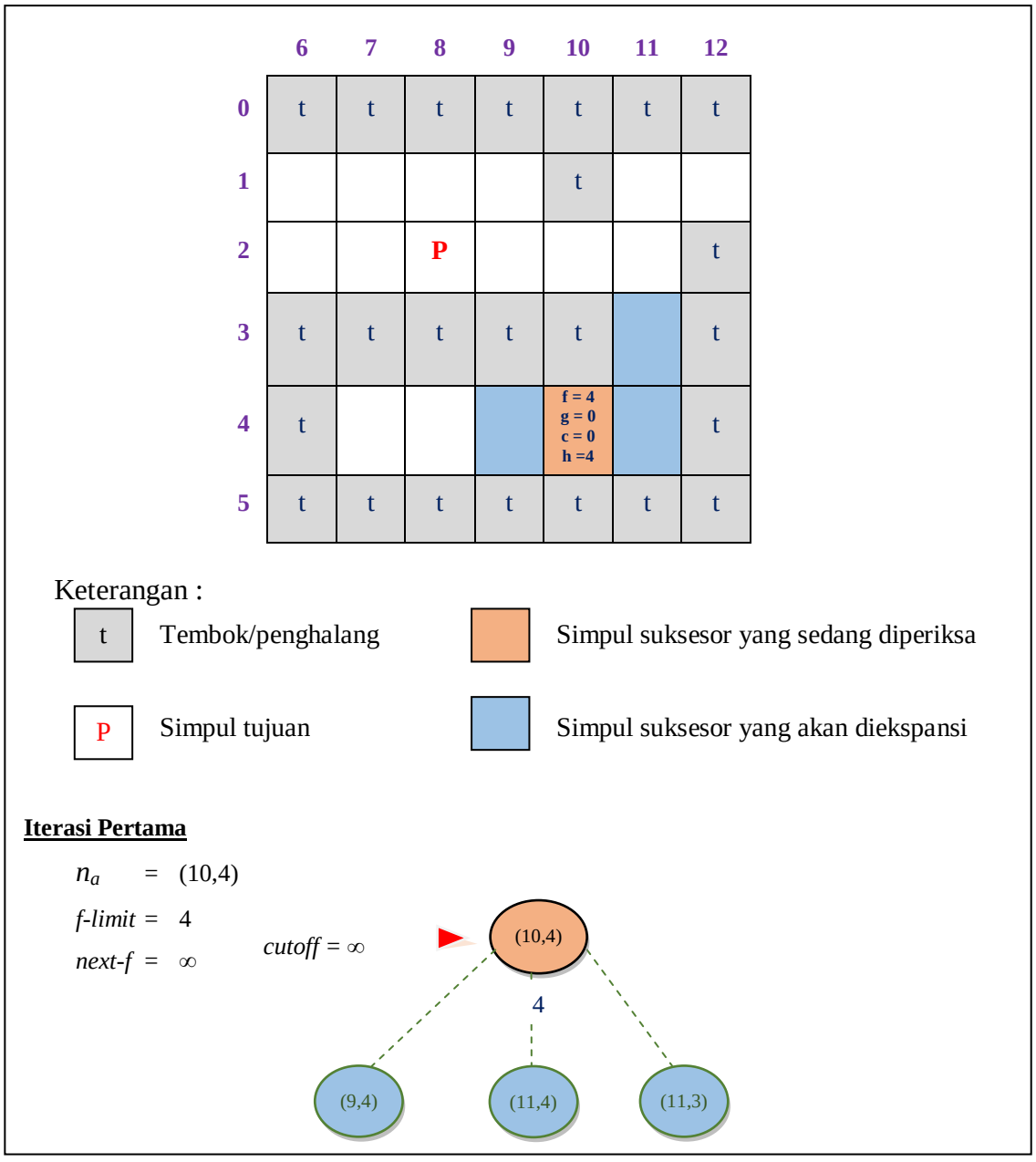
   = 0 + 0 + 4 = **4**

4. Tentukan *f-limit* dari $f(n_a)$ simpul awal.

   f-limit = f(10,4) = **4**

5. Inisialisasi *next-f* = ∞ dan *cutoff* = ∞.

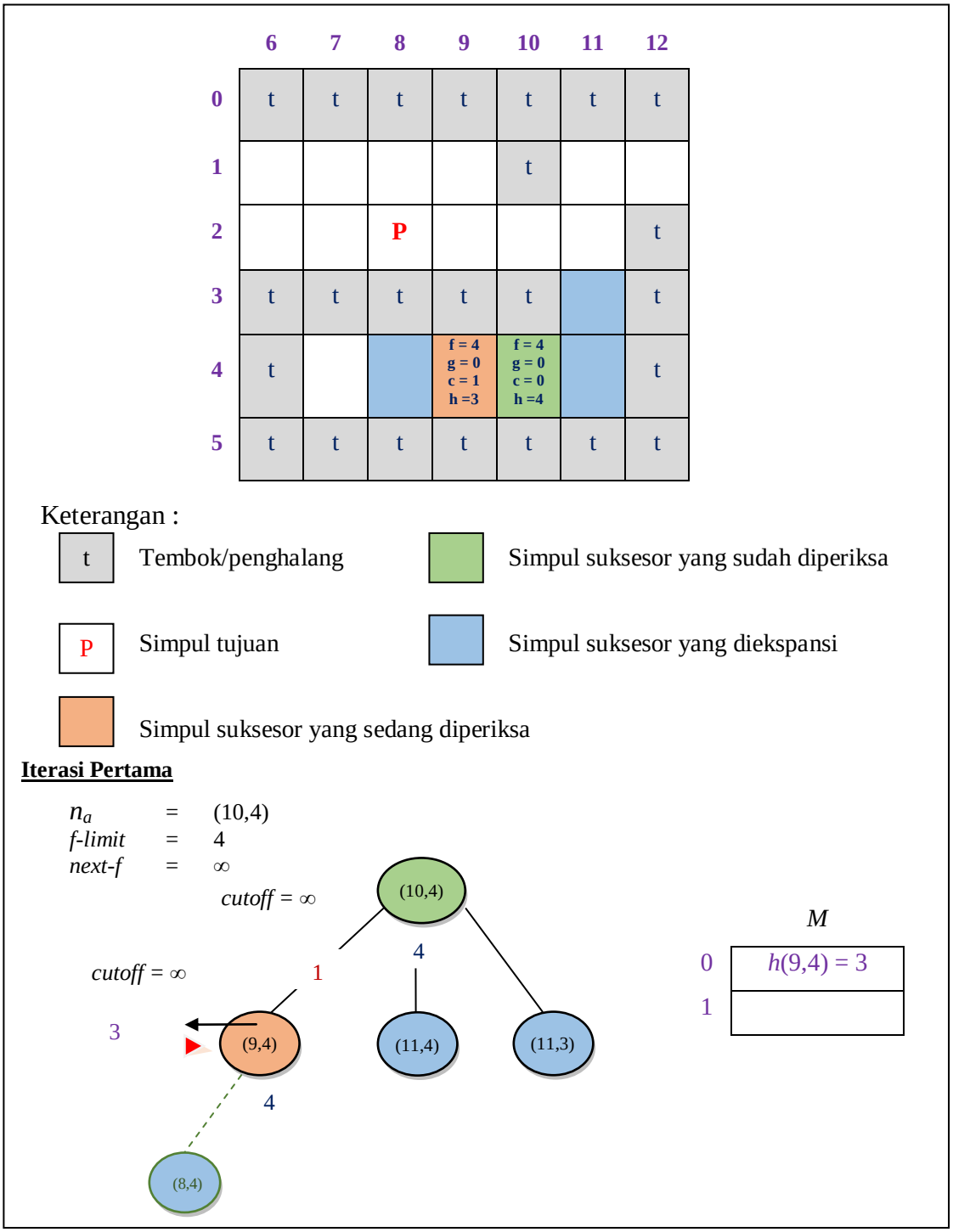
6. Lakukan pengecekan apakah *cache M* masih bisa menyimpan simpul dan nilai *RAND* ≤ *p* ? Misalkan nilai *random* yang didapat yaitu 0.74, maka simpul (10,4) dan nilai heuristiknya tidak bisa disimpan ke dalam *cache M*. Maka langsung ekspansi simpul suksessor dari simpul (10,4) dengan melakukan perulangan untuk setiap suksesor nya. Simpul (10,4) mempunyai 3 simpul suksesor yaitu simpul (9,4), (11,4), dan (11,3). Maka lakukan perulangan sebanyak 3 kali. Lihat Gambar 3.26.



Gambar 3.26 Langkah 1-6 pencarian IDA* SNC pada *game* labirin

7. Inisialisasi *cutoff* = ∞
8. Hitung nilai *f*(*n*) suksesor pertama, diperoleh nilai *f*(9,4) yaitu **4**. Karena nilai *f*(9,4) lebih kecil sama dengan nilai *f-limit*, maka lakukan pengecekan apakah c*ache M* masih bisa menyimpan simpul (9,4) dan nilai *random* ≤ *p* ? Misalkan angka *random* yang didapat adalah 0.41, maka nilai *random* menjadi lebih kecil dari nilai *p*, dan karena nilai *cache M* masih bisa

menyimpan simpul, maka simpan simpul (9,4) dan heuristik *h*(9,4) ke dalam *cache M*. Kemudian ekspansi simpul (9,4). Lihat Gambar 3.27.



Gambar 3.27 Langkah 7-8 pencarian IDA* SNC pada *game* labirin

9. Inisialisasi *cutoff* = ∞
10. Diperoleh simpul (8,4) sebagai suksesor dari simpul (9,4) dengan nilai *f*(8,4) = **4**. Karena nilai *f*(8,4) masih lebih kecil sama dengan nilai *f-limit*, maka lakukan pengecekan apakah *cache M* masih bisa menyimpan simpul dan nilai *RAND* ≤ *p* ? Misalkan nilai *random* yang didapat yaitu 0.221, maka nilai *random* menjadi lebih kecil dari nilai *p*, dan karena nilai *cache M* masih bisa menyimpan simpul, maka simpan simpul (8,4) dan heuristik *h*(8,4) ke dalam *cache M*. Kemudian ekspansi simpul (8,4). Simpul (8,4) mempunyai 2 suksesor yaitu simpul (7,4) dan (9,4). Lihat Gambar 3.28 dan Gambar 3.29.

	6	7	8	9	10	11	12
0	t	t	t	t	t	t	t
1					t		
2			P				t
3	t	t	t	t	t		t
4	t		f = 4 g = 1 c = 1 h =2	f = 4 g = 0 c = 1 h =3	f = 4 g = 0 c = 0 h =4		t
5	t	t	t	t	t	t	t

Keterangan :

t — Tembok/penghalang

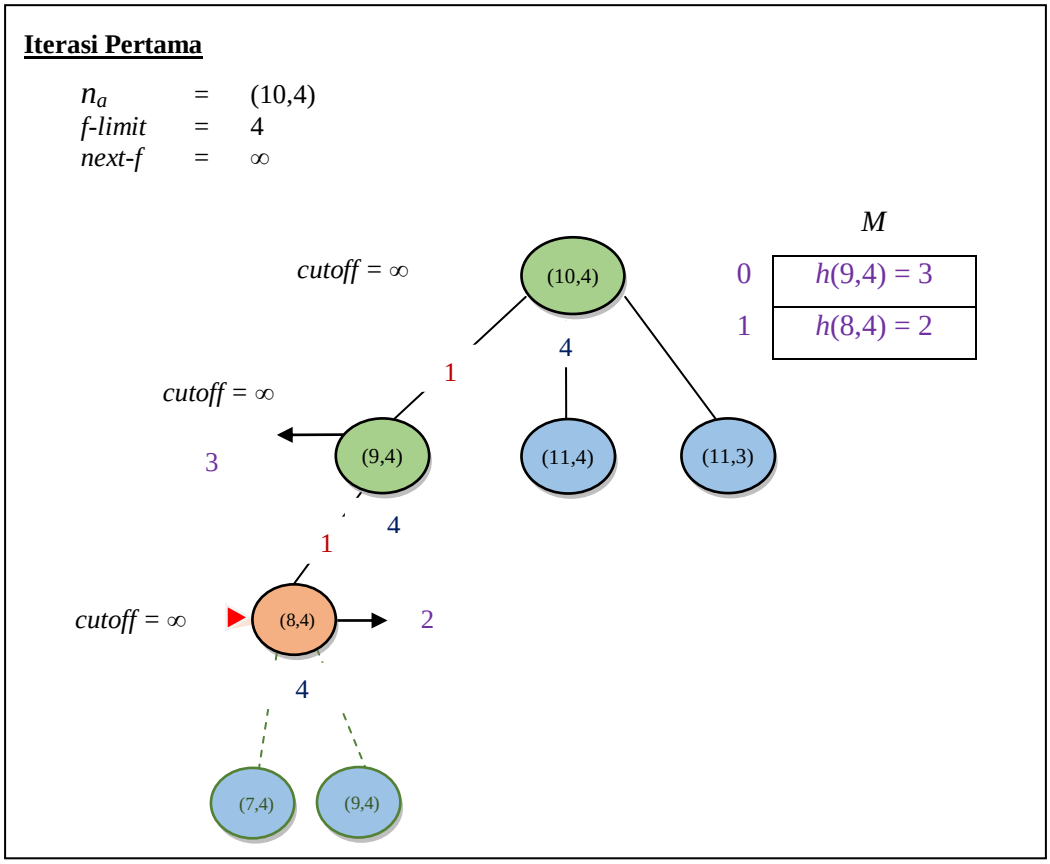
Simpul suksesor yang sudah diperiksa

P — Simpul tujuan

Simpul suksesor yang diekspansi

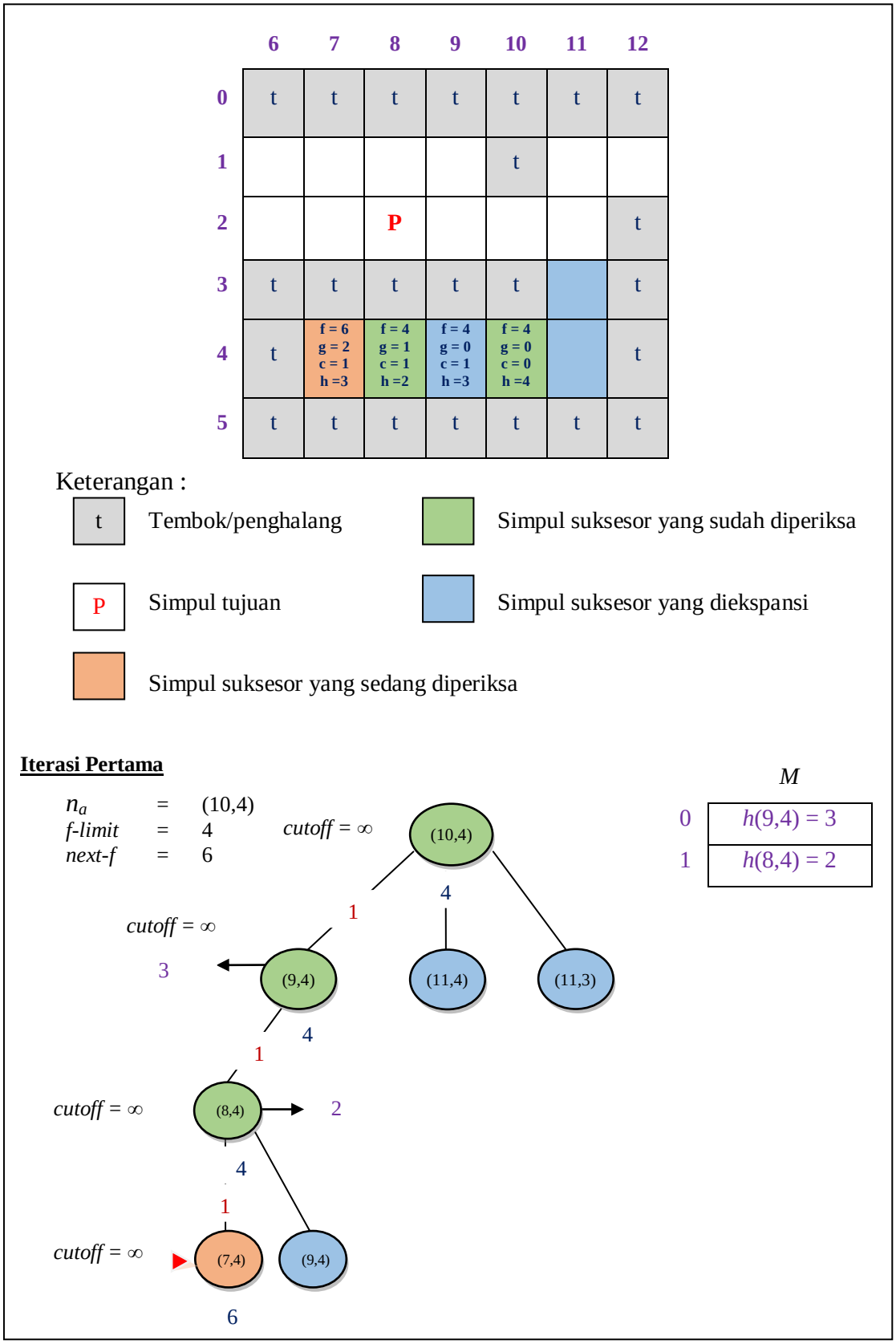
Simpul suksesor yang sedang diperiksa

Gambar 3.28 Langkah 9-10 pencarian IDA* SNC pada *game* labirin



Gambar 3.29 Langkah 9-10 pencarian IDA* SNC pada *game* labirin (2)

11. Inisialisasi *cutoff* = ∞
12. Lakukan pencarian berdasarkan arah pencarian. Diperoleh simpul (7,4) sebagai suksesor pertama dari simpul (8,4) dengan nilai *f*(7,4) = **6**. Namun karena simpul (7,4) bukan simpul tujuan dan nilai *f*(7,4) > *f-limit*, maka simpul (7,4) tidak diekspansi tetapi jadikan nilai *f*(7,4) menjadi nilai *next-f*. Lihat Gambar 3.30.



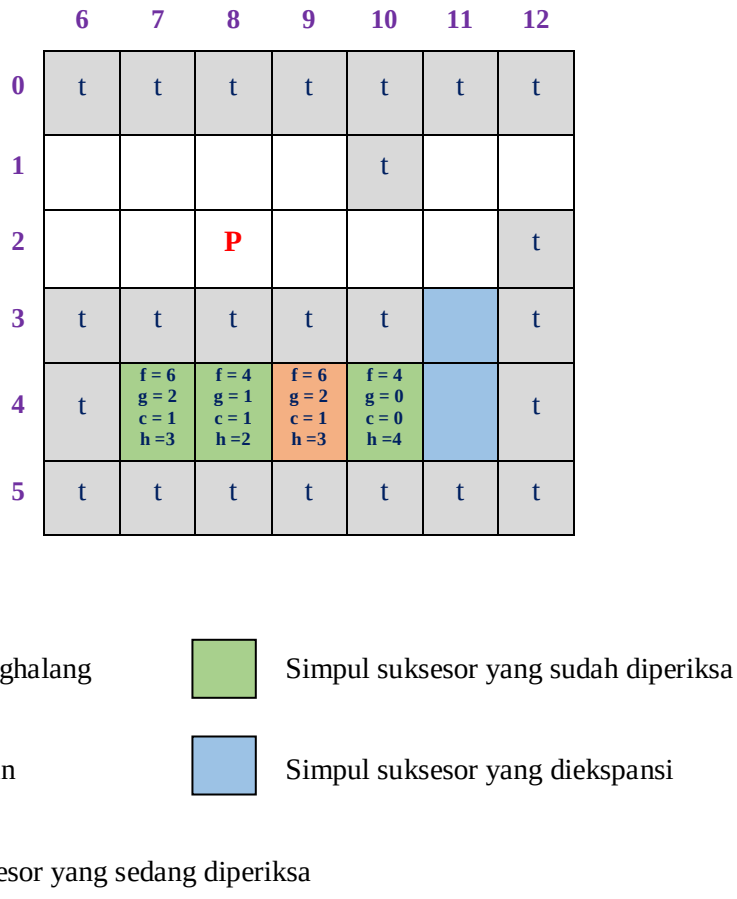
Gambar 3.30 Langkah 11-12 pencarian IDA* SNC pada *game* labirin

13. Lakukan pengecekan untuk suksesor pertama dari simpul (8,4) yaitu simpul (7,4).

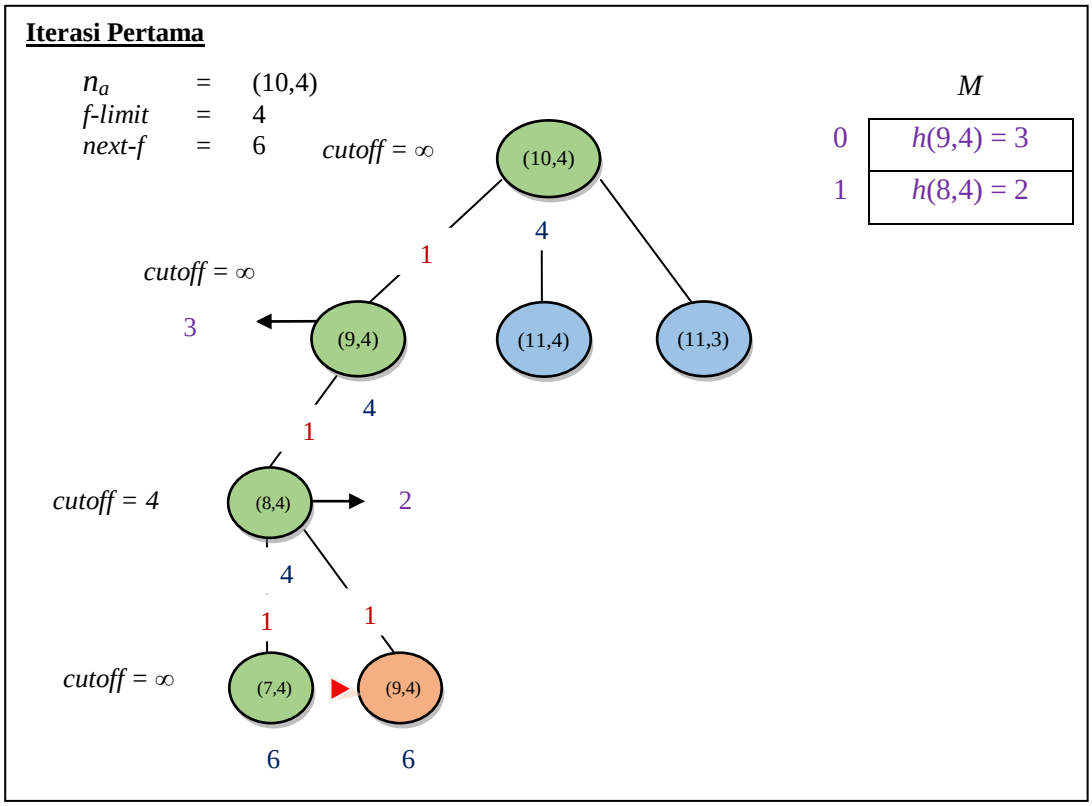
    Apakah nilai $h(7,4) + c(8,4;7,4) < cutoff$?

    Ya, $3 + 1 < \infty$, maka jadikan sebagai nilai *cutoff* baru. *Cutoff* = 4

14. Lakukan pencarian ke suksesor dari simpul (8,4) berikutnya yaitu simpul (9,4). Diperoleh nilai $f(9,4) = \mathbf{6}$. Karena nilai $f(9,4)$ lebih besar dari nilai *f-limit*, maka simpul (9,4) juga tidak diekspansi dan nilai *next-f* tetap karena nilai $f(9,4)$ tidak lebih kecil dari nilai *next-f*. Lihat Gambar 3.31 dan Gambar 32.



Gambar 3.31 Langkah 13-14 pencarian IDA* SNC pada *game* labirin



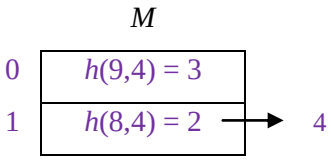
Gambar 3.32 Langkah 13-14 pencarian IDA* SNC pada *game* labirin (2)

15. Lakukan pengecekan untuk suksesor kedua dari simpul (8,4) yaitu simpul (9,4).

    Apakah nilai $h(9,4) + c(8,4;9,4) < cutoff$?

    Ya, 3 + 1 < 4? Tidak, maka nilai *cutoff* tetap. *Cutoff* = 4

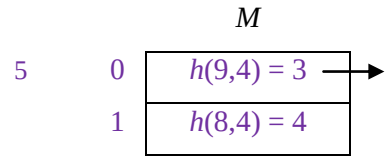
16. *Update* nilai heuristik simpul (8,4) di *cache M* dengan nilai *cutoff*. Perubahan nilai heuristik ini akan berakibat pada nilai $f(n_i)$ iterasi berikutnya juga semakin bertambah.



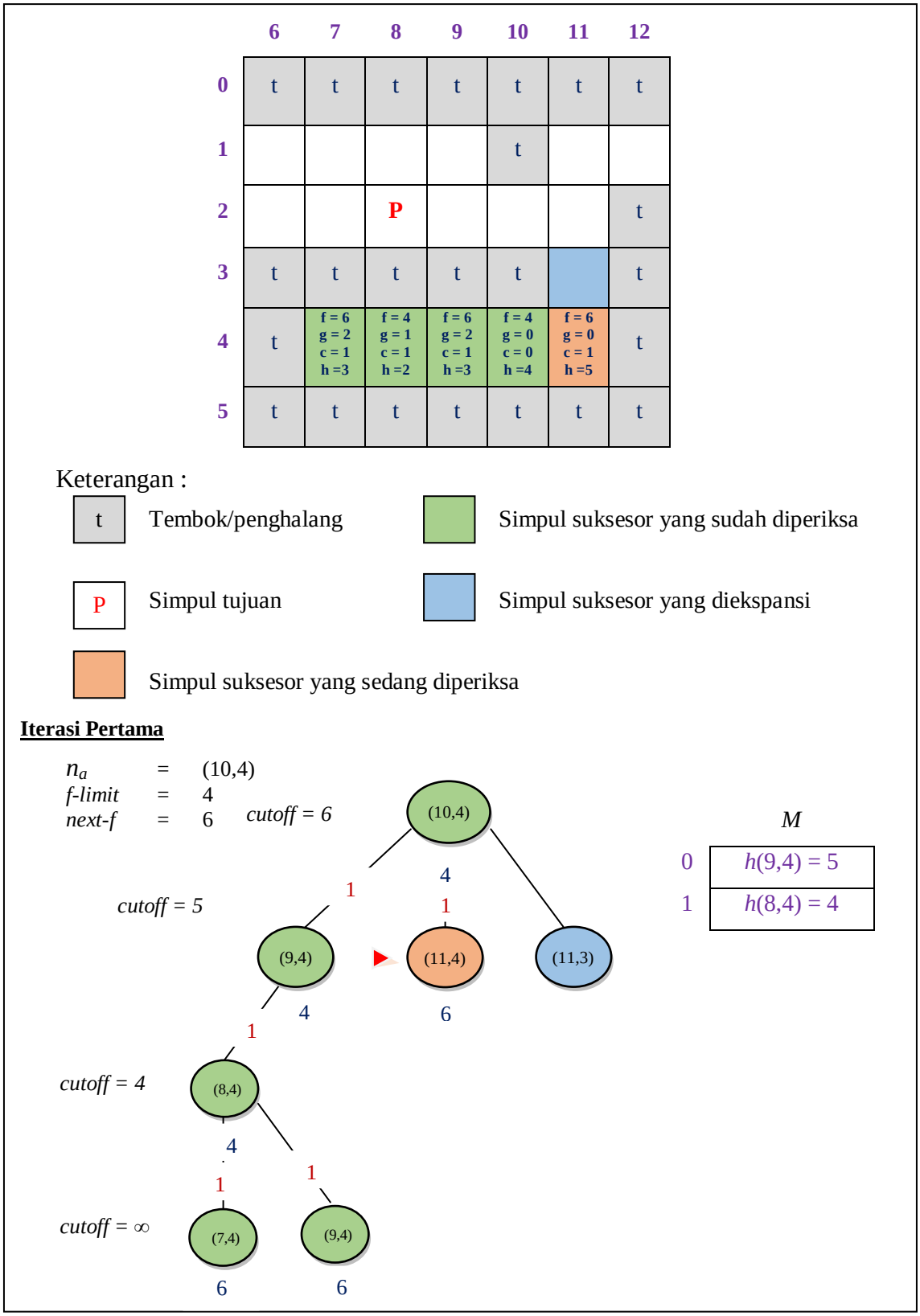
17. Setelah keluar dari perulangan untuk suksesor dari simpul (8,4), sekarang lakukan pengecekan untuk suksesor dari simpul (9,4). Apakah nilai $h(8,4) + c(9,4;8,4) < cutoff$? (nilai heuristik yang ada di *cache M*)

    Ya, 4 + 1 < ∞, maka jadikan sebagai nilai *cutoff* baru. *Cutoff* = 5

18. *Update* nilai heuristik simpul (9,4) di *cache M* dengan nilai *cutoff*. Perubahan nilai heuristik ini akan berakibat pada nilai $f(n_i)$ iterasi berikutnya juga semakin bertambah.



19. Setelah keluar dari perulangan untuk suksesor dari simpul (9,4), sekarang lakukan pengecekan untuk suksesor dari simpul (10,4). Apakah nilai $h(9,4) + c(10,4;9,4) < cutoff$? (nilai heuristik yang ada di *cache M*)
    Ya, $5 + 1 < \infty$, maka jadikan sebagai nilai *cutoff* baru. *Cutoff* = 6
20. Simpul (9,4) adalah suksesor dari simpul (10,4) yang sudah diperiksa, sekarang pencarian dilanjutkan ke simpul suksesor berikutnya dari simpul (10,4) yaitu simpul (11,4) dengan nilai $f(11,4) = \mathbf{6}$. Karena nilai $f(11,4)$ lebih besar dari nilai *f-limit*, maka simpul (11,4) juga tidak diekspansi dan nilai *next-f* tetap karena nilai $f(11,4)$ tidak lebih kecil dari nilai *next-f*. Lihat Gambar 3.33.



Gambar 3.33 Langkah 15-20 pencarian IDA* SNC pada *game* labirin

21. Lakukan pengecekan apakah nilai $h(11,4) + c(10,4;11,4) < cutoff$? 5+1 < 6? Tidak, maka nilai *cutoff* tetap.
22. Karena simpul tujuan masih belum ditemukan, maka pencarian dilanjutkan ke simpul suksesor (11,3). Pada simpul suksesor (11,3) diperoleh nilai $f(11,3) =$ **5.41**. Namun karena simpul (11,3) bukan simpul tujuan dan nilai $f(11,3) > f\text{-}limit$, maka simpul (11,3) tidak diekspansi tetapi jadikan nilai $f(11,3)$ menjadi nilai *next-f* karena nilai $f(11,3)$ lebih kecil dari nilai *next-f*. Lihat Gambar 3.34 dan Gambar 3.35.
23. Lakukan pengecekan apakah nilai $h(11,3) + c(10,4;11,3) < cutoff$?
    Ya, 4 + 1.41 < 6, maka jadikan sebagai nilai *cutoff* baru. *Cutoff* = 5.41

	6	7	8	9	10	11	12
0	t	t	t	t	t	t	t
1					t		
2			P				t
3	t	t	t	t	t	f = 5.41 g = 0 c = 1.41 h =4	t
4	t	f = 6 g = 2 c = 1 h =3	f = 4 g = 1 c = 1 h =2	f = 6 g = 2 c = 1 h =3	f = 4 g = 0 c = 0 h =4	f = 6 g = 0 c = 1 h =5	t
5	t	t	t	t	t	t	t

Keterangan :

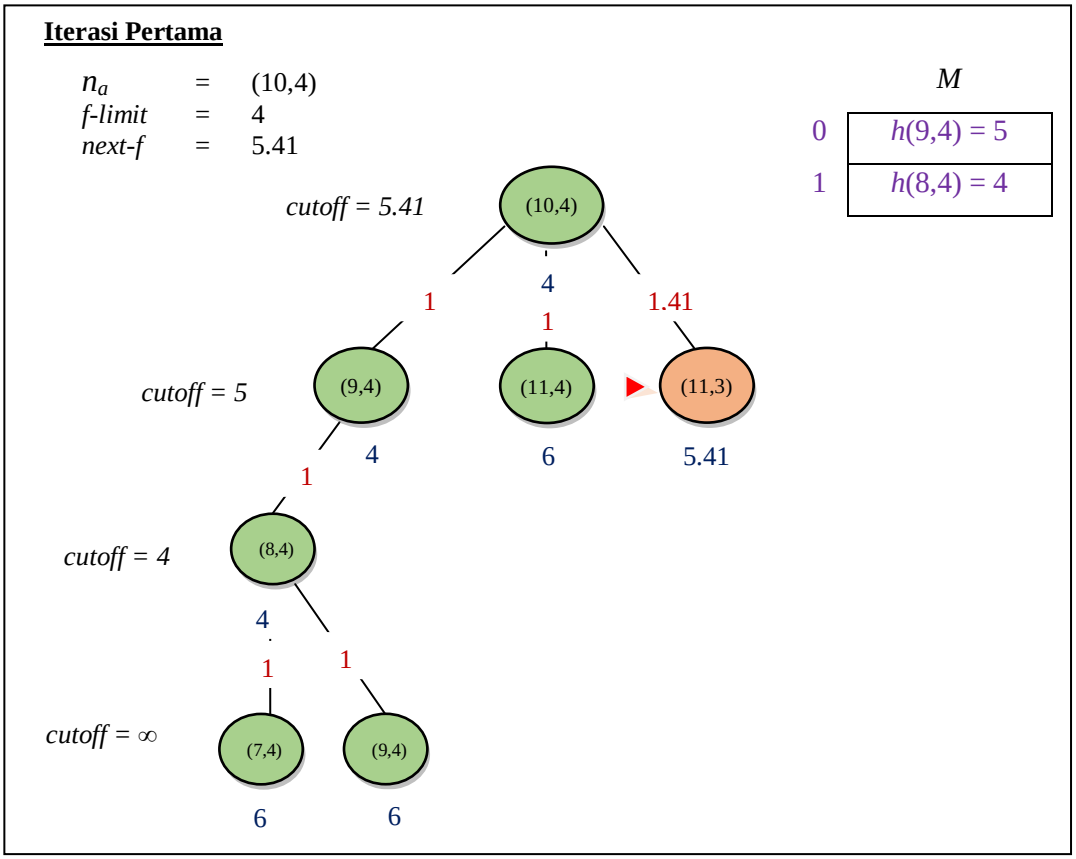
t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

P — Simpul tujuan

Simpul suksesor yang sedang diperiksa

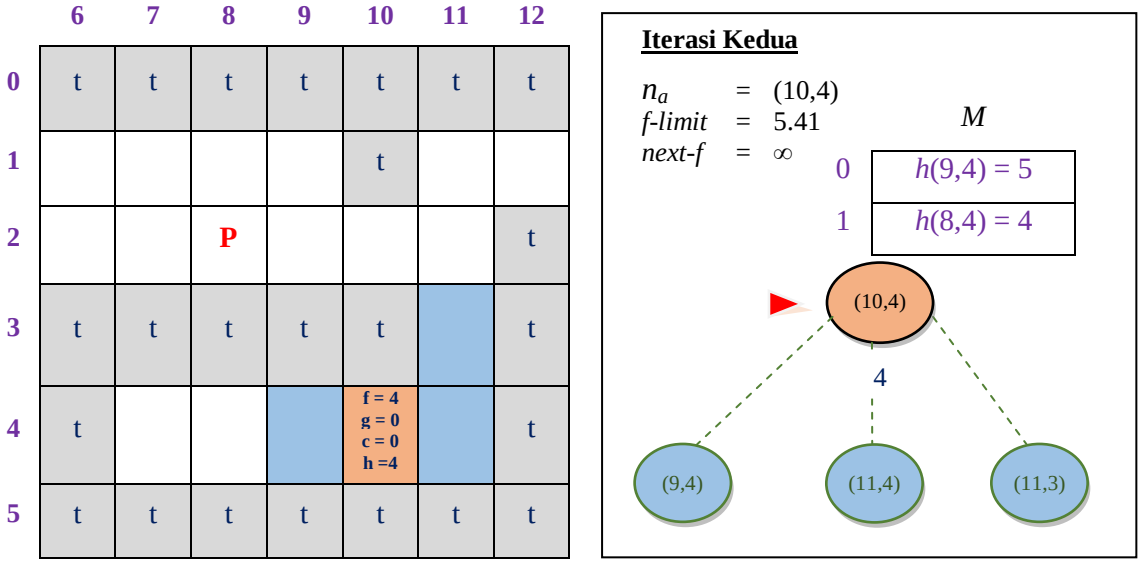
Gambar 3.34 Langkah 21-23 pencarian IDA* SNC pada *game* labirin



Gambar 3.35 Langkah 21-23 pencarian IDA* SNC pada *game* labirin (2)

24. Setelah suksesor dari simpul (10,4) diperiksa dan karena sebelumnya simpul (10,4) tidak di simpan di *cache M*, maka heuristik simpul (10,4) tidak di *update*.
25. Setelah semua suksesor dari simpul (10,4) diperiksa, set nilai *next-f* menjadi nilai *f-limit* baru dan nilai *next-f* menjadi ∞.

    f-limit = **5.41**
26. Pada iterasi pertama, semua simpul sudah diperiksa, namun belum menemukan solusi. Oleh karena itu pencarian dimulai dari status awal lagi dengan nilai *f-limit* sekarang yaitu **5.41**. Ekspansi simpul (10,4) dengan lakukan perulangan untuk setiap suksesor n_i dari simpul (10,4). Simpul (10,4) mempunyai 3 simpul suksesor yaitu simpul (9,4), (11,4), dan (11,3). Maka lakukan perulangan sebanyak 3 kali. Lihat Gambar 3.36.



Keterangan :

t — Tembok/penghalang

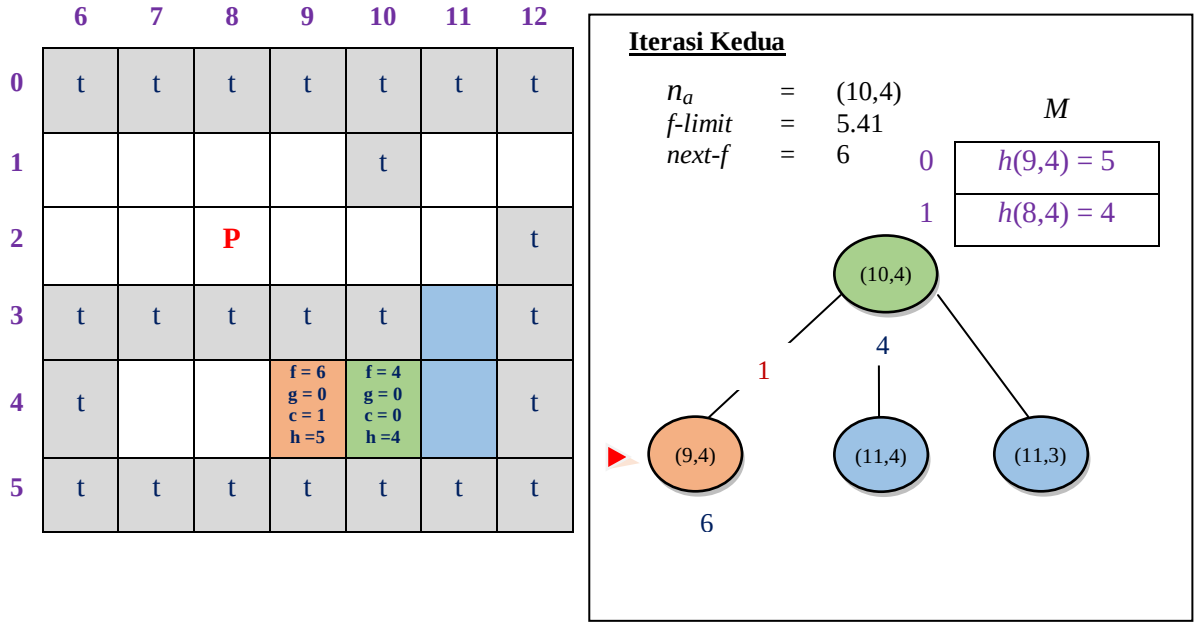
Simpul suksesor yang sedang diperiksa

P — Simpul tujuan

Simpul suksesor yang diekspansi

Gambar 3.36 Langkah 24-26 pencarian IDA* SNC pada *game* labirin

27. Hitung nilai *f*(*n*) suksesor pertama, dengan menggunakan nilai heuristik yang baru yaitu nilai heuristik yang ada di-*cache*, maka diperoleh nilai $f(n_i)$ pada suksesor pertama dari simpul (10,4) yaitu *f*(9,4) = 6. Sehingga pada saat pencarian iterasi kedua dilakukan, simpul (9,4) tidak akan mengekspansi simpul suksesor karena nilai *f*(9,4) menjadi lebih besar dari *f-limit*, dan nilai *next-f* yaitu 6. Lihat Gambar 3.37.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

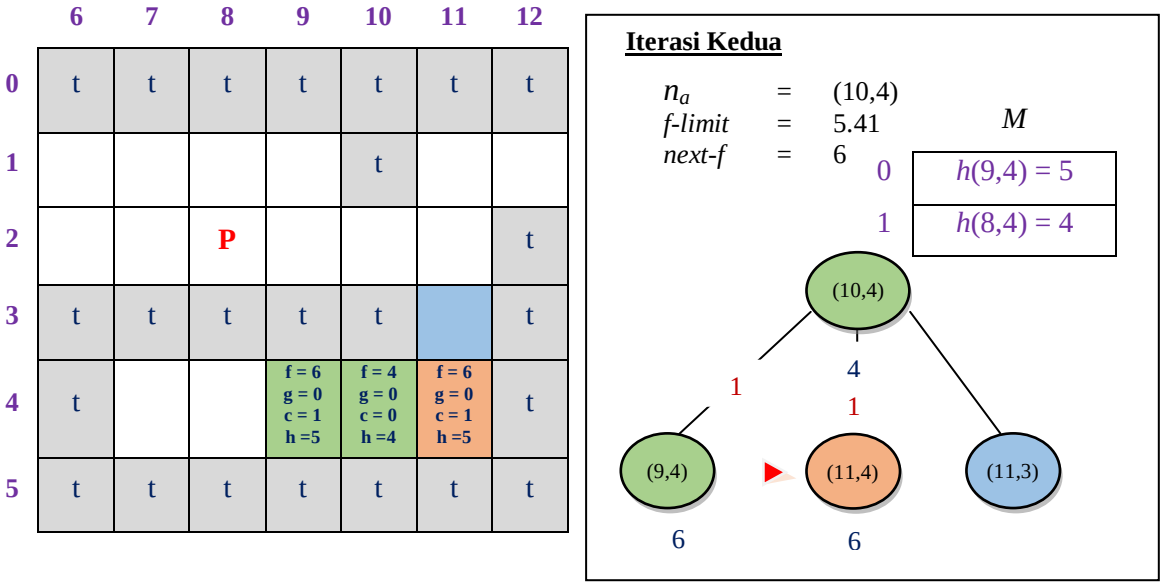
P — Simpul tujuan

Simpul suksesor yang diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.37 Langkah 27 pencarian IDA* SNC pada *game* labirin

28. Karena simpul tujuan belum ditemukan, maka pencarian dilanjutkan ke simpul suksesor berikutnya dari simpul (10,4) yaitu simpul (11,4) dengan nilai $f(11,4)$ = **6**. Lihat Gambar 3.38.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

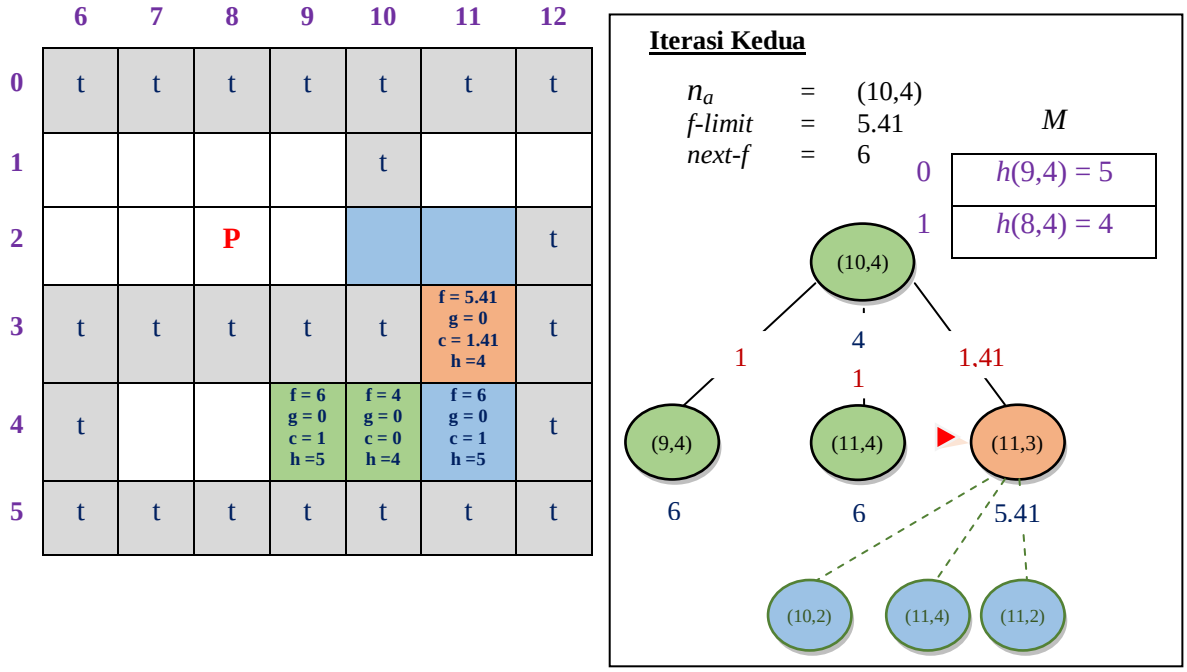
P — Simpul tujuan

Simpul suksesor yang diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.38 Langkah 28 pencarian IDA* SNC pada *game* labirin

29. Karena simpul tujuan masih belum ditemukan, maka pencarian dilanjutkan ke simpul suksesor (11,3). Pada suksesor ke-3 diperoleh nilai $f(11,3) \leq$ *f-limit* yaitu **5.41**, maka langsung ekspansi simpul suksessor dari simpul (11,3) untuk mencari suksesor selanjutnya karena *cache M* tidak dapat menyimpan simpul lagi. Simpul (11,3) memiliki 3 suksesor yaitu simpul (10,2), simpul (11,4) dan simpul (11,2). Lihat Gambar 3.39.



Keterangan :

t — Tembok/penghalang

Simpul suksesor yang sudah diperiksa

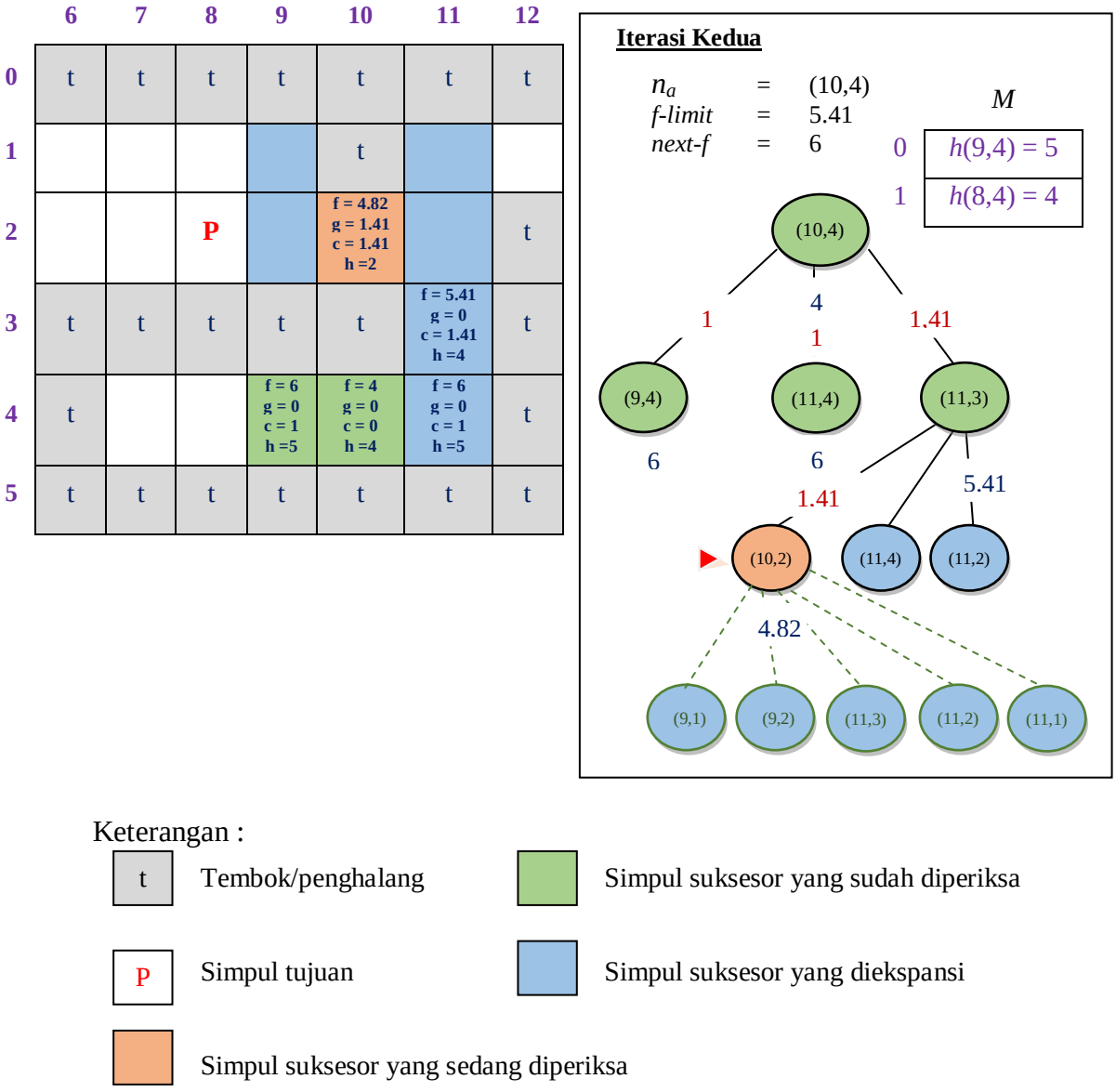
P — Simpul tujuan

Simpul suksesor yang diekspansi

Simpul suksesor yang sedang diperiksa

Gambar 3.39 Langkah 29 pencarian IDA* SNC pada *game* labirin

30. Hitung nilai $f(n_i)$ suksesor pertama dari simpul (11,3) yaitu simpul (10,2). Diperoleh nilai $f(10,2)$ = 4.82 yang artinya masih lebih dari *f-limit*, maka langsung ekspansi simpul (10,2) dengan melakukan perulangan perhitungan $f(n_i)$ untuk setiap suksesor n_i dari simpul (10,2). Simpul (10,2) memiliki 4 simpul suksesor yaitu simpul (9,1), simpul (9,2), simpul (11,2) dan simpul (11,1). Lihat Gambar 3.40.

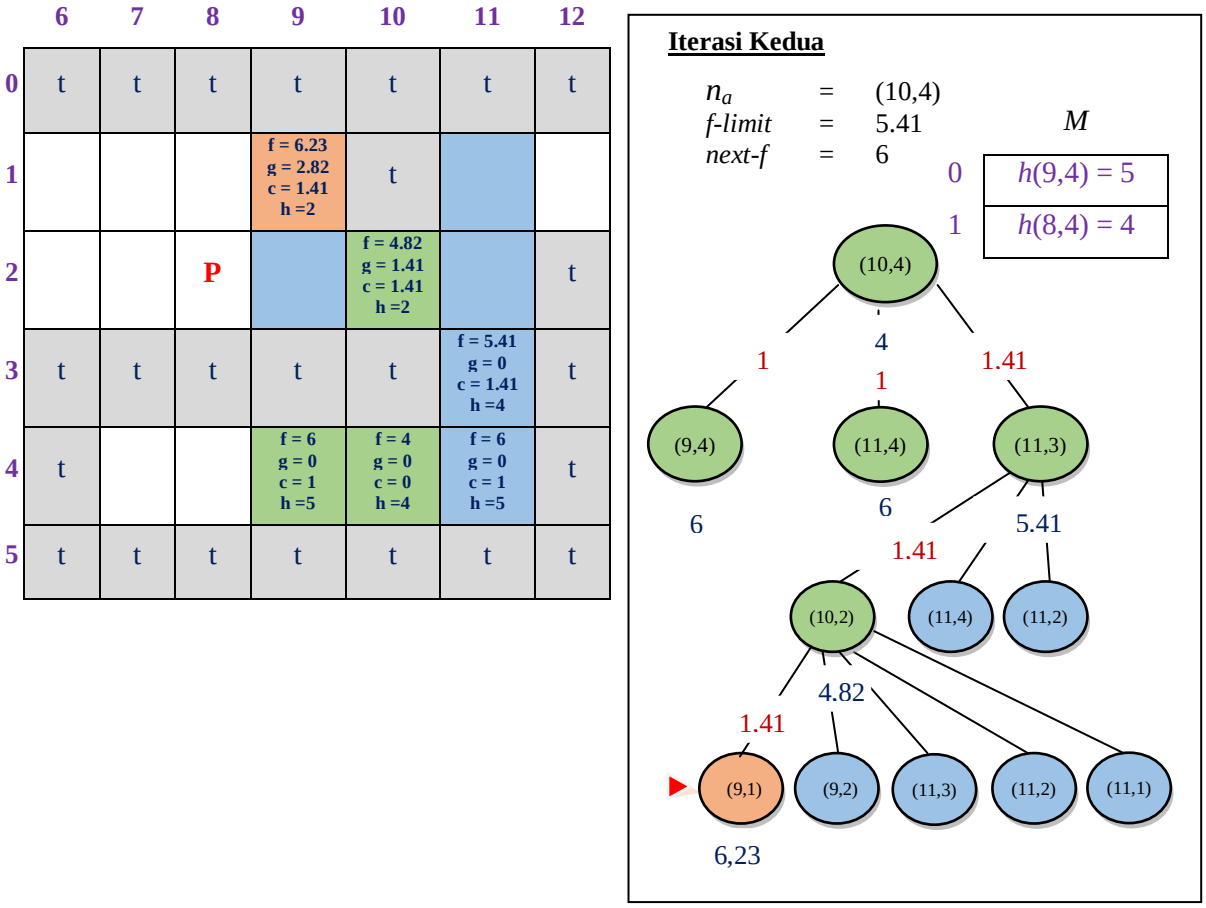


Keterangan :

t	Tembok/penghalang
	Simpul suksesor yang sudah diperiksa
P	Simpul tujuan
	Simpul suksesor yang diekspansi
	Simpul suksesor yang sedang diperiksa

Gambar 3.40 Langkah 30 pencarian IDA* SNC pada *game* labirin

31. Hitung nilai $f(n_i)$ suksesor pertama dari simpul (10,2). Simpul (9,1) adalah simpul yang pertama kali diperiksa dan hitung nilai $f(9,1)$. Diperoleh nilai $f(9,1) = 6{,}23$. Lihat Gambar 3.41.

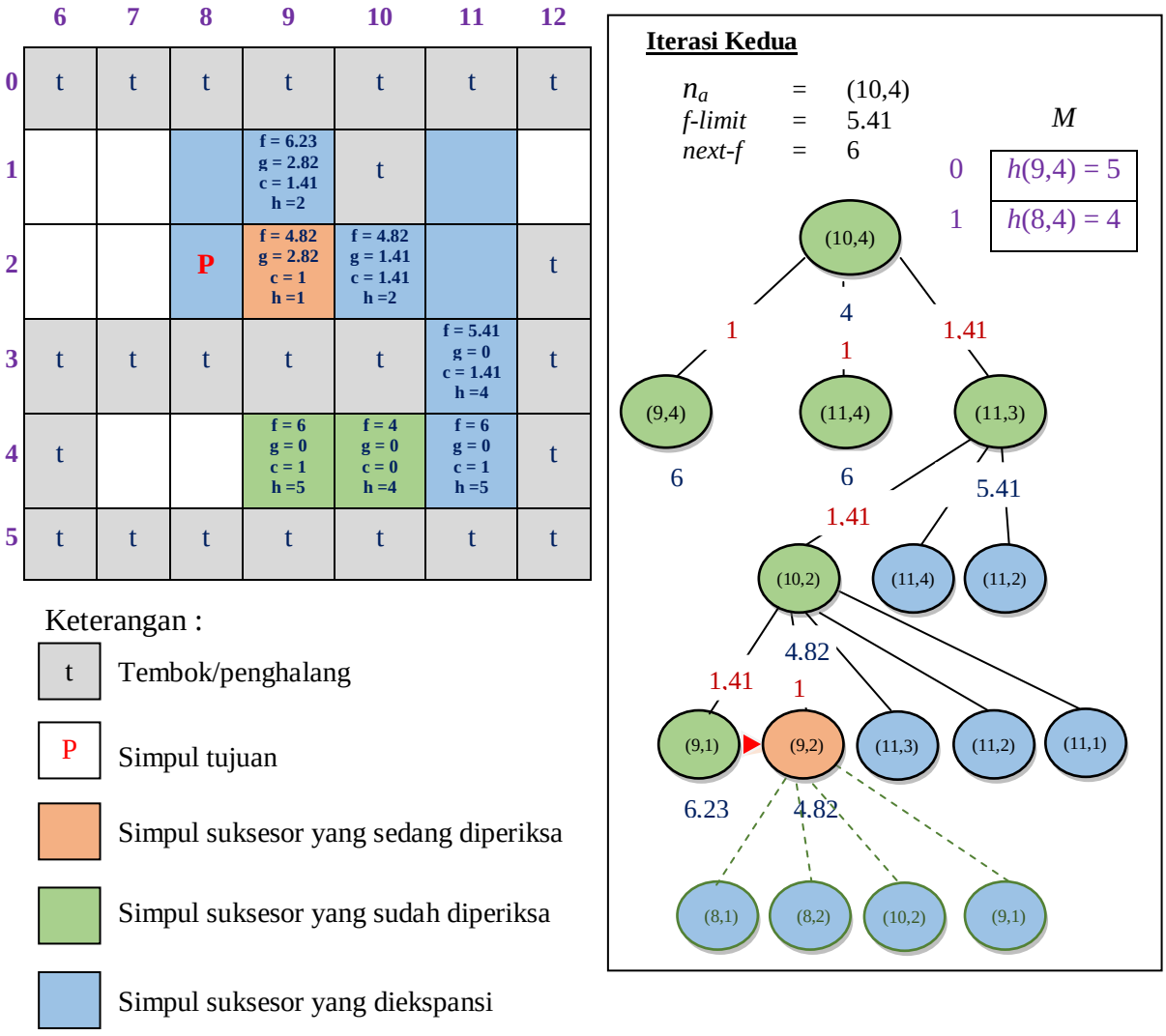


Keterangan :

t	Tembok/penghalang		Simpul suksesor yang sudah diperiksa
P	Simpul tujuan		Simpul suksesor yang diekspansi
	Simpul suksesor yang sedang diperiksa		

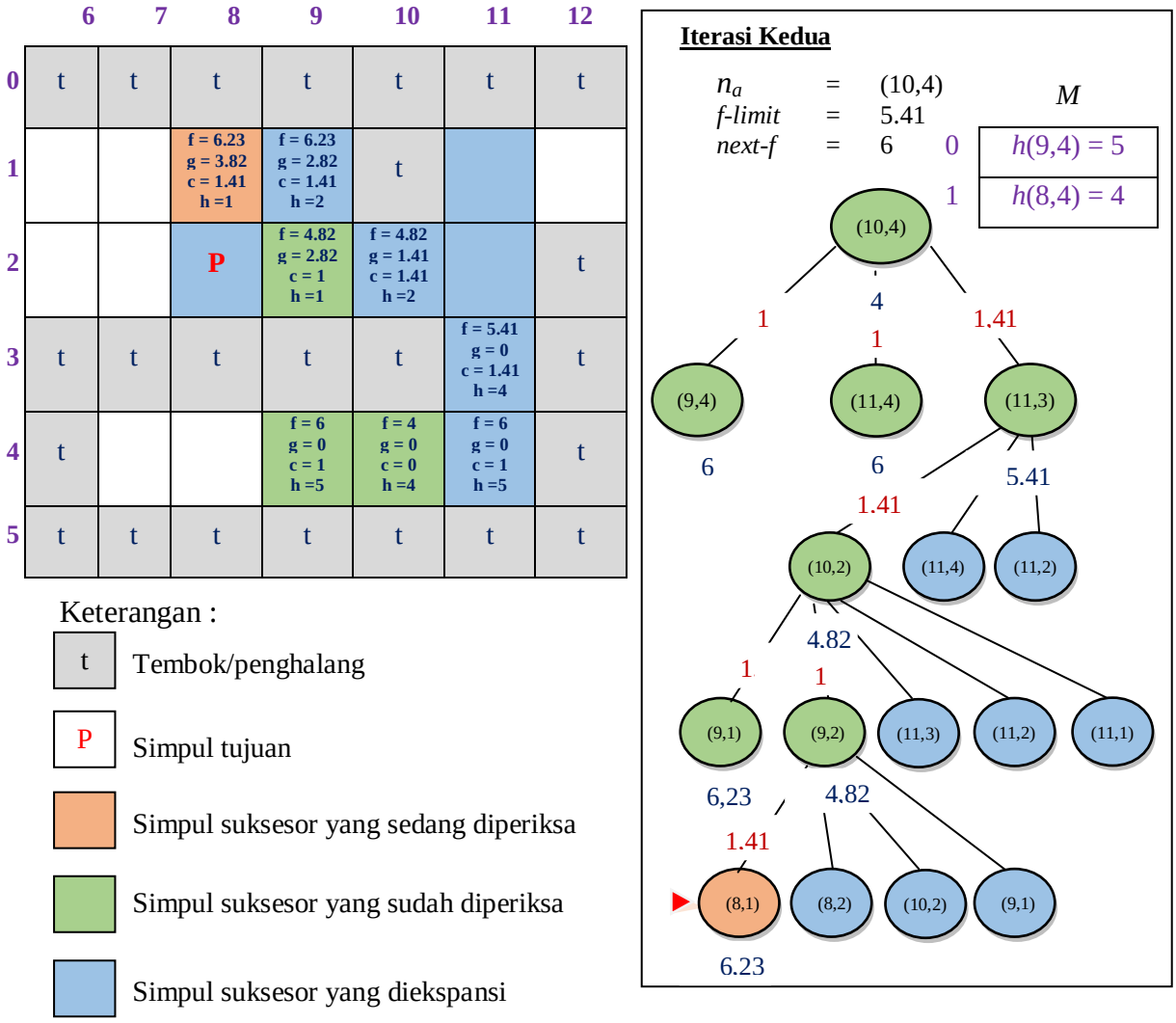
Gambar 3.41 Langkah 31 pencarian IDA* SNC pada *game* labirin

32. Namun karena simpul (9,1) memiliki nilai $f(9,1)$ lebih besar dari nilai *f-limit*, maka dilakukan ke suksesor berikutnya yaitu simpul (9,2). Ternyata simpul (9,2) memiliki nilai $f(9,2) = 4{,}82$ yang nilainya masih lebih kecil sama dengan *f-limit*, maka ekspansi simpul (9,2) dengan melakukan perulangan perhitungan $f(n_i)$ untuk setiap suksesor n_i dari simpul (9,2). Simpul (9,2) memiliki 4 simpul suksesor yaitu simpul (8,1), simpul (8,2), simpul (10,2) dan simpul (9,1). Lihat Gambar 3.42.



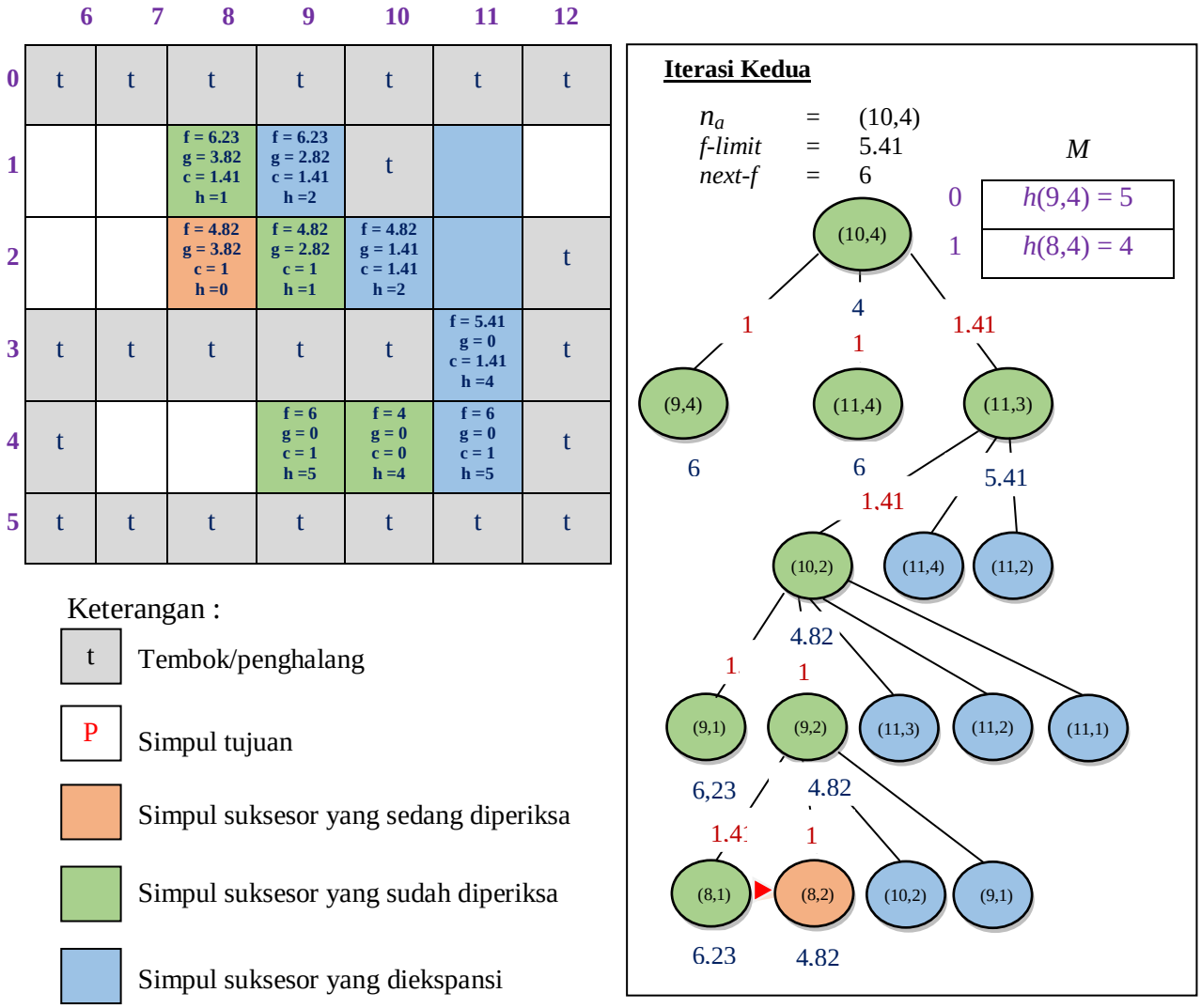
Gambar 3.42 Langkah 32 pencarian IDA* SNC pada *game* labirin

33. Hitung nilai $f(n_i)$ suksesor pertama dari simpul (9,2). Simpul (8,1) adalah simpul yang pertama kali diperiksa dan diperoleh nilai $f(8,1) = 6{,}23$. Lihat Gambar 3.43.



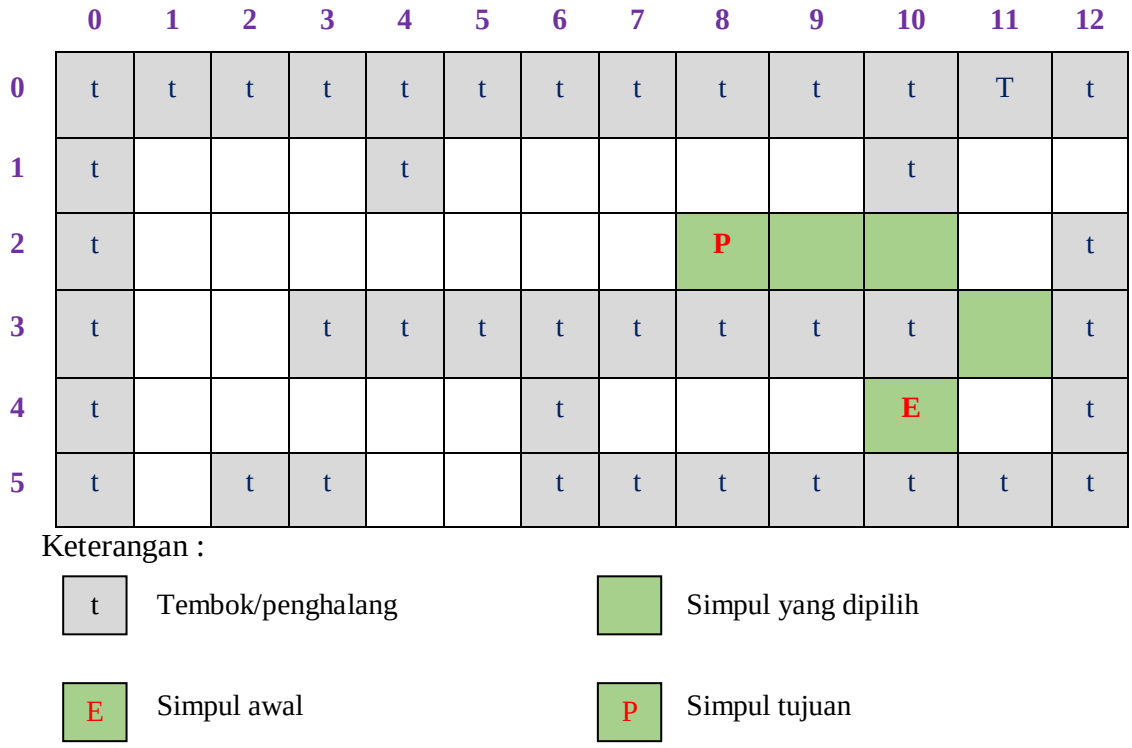
Gambar 3.43 Langkah 33 pencarian IDA* SNC pada *game* labirin

34. Namun karena simpul (8,1) memiliki nilai *f*(8,1) lebih besar dari nilai *f-limit*, maka pencarian dilakukan ke suksesor berikutnya yaitu simpul (8,2). Ternyata simpul (8,2) adalah simpul tujuan dan memiliki nilai *f*(8,2) = 4,82 yang nilainya lebih kecil sama dengan *f-limit*. Maka pecarian berhenti dengan menemukan simpul (8,2) sebagai simpul tujuan. Lihat Gambar 3.44.



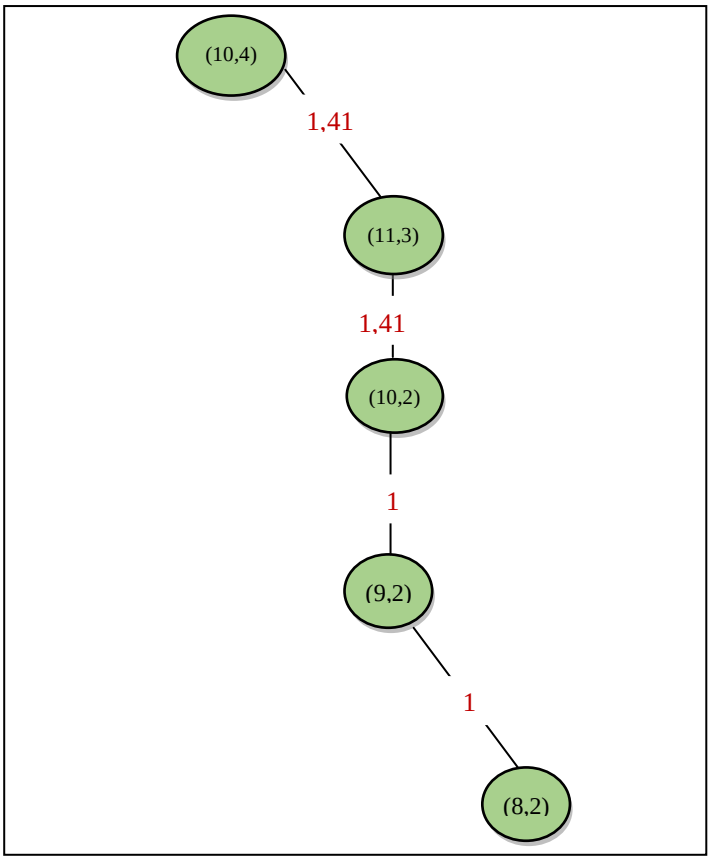
Gambar 3.44 Langkah 34 pencarian IDA* SNC pada *game* labirin

Pencarian IDA* SNC pada kasus *game* labirin ini mengekspansi sebanyak 16 simpul dengan 7 simpul pada iterasi pertama dan pada iterasi kedua berjumlah 9 simpul. Jumlah ekspansi simpul pada pencarian IDA* SNC lebih sedikit dari IDA* yang mengekspansi sebanyak 19 simpul, Hal ini dikarenakan IDA* SNC menyimpan nilai heuristik pada *cache M* sehingga dapat mengurangi jumlah kunjungan kembali simpul yang sama dan menyebabkan waktu pencarian lebih cepat. Hasil akhir pencarian jalur dari simpul awal (E) menuju simpul tujuan (P) dapat dilihat pada Gambar 3.45.



Gambar 3.45 Hasil pencarian IDA* SNC pada *game* labirin

Gambar 3.46 menunjukkan hasil akhir penyelesaian dari algoritma IDA* SNC pada *game* labirin, diperoleh jalur terpendek yang ditemukan adalah **(10,4) – (11,3) – (10,2) – (9,2) – (8,2)** dengan biaya keseluruhan yaitu **4.82**. Pencarian IDA* SNC juga akan menghapus jalur yang tidak dipilihnya.



Gambar 3.46 Pohon Hasil Akhir Pencarian IDA* SNC pada *game* labirin

Kompleksitas Waktu Algoritma IDA* SNC

Kompleksitas waktu algoritma pada IDA* SNC sama dengan kompleksitas waktu pada algoritma IDA* yaitu sebesar $O(b^d)$, dimana b adalah faktor percabangan dan d adalah kedalaman pencarian [8]. Walaupun IDA* dengan SNC dapat mengurangi jumlah revisit (kunjungan kembali) simpul-simpul yang dibangkitkan, namun cara kerja IDA* SNC tetap dilakukan secara iteratif sama seperti IDA*, membangkitkan simpul-simpul yang sama secara berulang-ulang. Berikut ini adalah perhitungan kompleksitas waktu algoritma IDA* SNC :

Tabel 3. 1 Solusi Kedalaman *d* dengan percabangan *b*

Kedalaman / *Depth*	Total *path* pada IDA* SNC
1	db
2	$(d-1)b^2$
. .	. .
d-1	$2b^{d-1}$
d	b^d

Tabel 3.1 adalah pencarian solusi IDA* SNC dengan mengekspansi kedalaman level *d* dengan percabangan *b*. Simpul pada level terbawah diekspansi 1 kali, simpul yang berada diatas node terbawah diekspansi 2 kali dan begitu seterusnya sampai ke simpul akar dari pohon pencarian. Sehingga total keseluruhan *path* yang diekspansi yaitu :

$$b^d + 2b^{d-1} + 3b^{d-2} + \cdots + db$$
$$= b^d(1b^0 + 2b^{-1} + 3b^{-2} + \cdots + db^{1-d}$$
$$= b^d\left(\sum_{i=1}^{d} ib^{1-i}\right)$$
$$= b^d\left(\sum_{i=1}^{d} i(b^{-1})^{i-1}\right)$$
$$= b^d\left(\frac{1}{1-b^{-1}}\right)^2$$
$$= b^d\left(\frac{b}{b-1}\right)^2 \in O(b^d)$$

Jadi, algoritma IDA* SNC memiliki kompleksitas waktu sebesar $O(b^d)$.

3.5 Analisis Kebutuhan Implementasi Algoritma

Analisis kebutuhan implementsi algoritma mecangkup analisis non fungsional yang akan digunakan untuk keperluan implementasi algoritma pada penelitian ini. Analisis kebutuhan non fungsional menggambarkan kebutuhan luar sistem yang diperlukan untuk menjalankan *game* yang dibangun. Adapun kebutuhan non fungsional pada implementasi algoritma IDA* dan IDA* SNC pada *game* labirin meliputi analisis kebutuhan perangkat keras dan analisis kebutuhan perangkat lunak.

3.5.1 Analisis Perangkat Keras

Spesifikasi perangkat keras (*hardware*) yang dibutuhkan untuk menguji pengimplementasian algoritma IDA* dan IDA* SNC pada *game* labirin ini dapat dilihat pada Tabel 3.2.

Tabel 3.2 Kebutuhan Perangkat Keras

Perangkat Keras	Spesifikasi
Processor	*Intel Core*-i5-3230M 2.6 Ghz
Resolusi layar	1024 x 768 pixel
Memori	4 GB
Harddisk	1TB
Keyboard	-
Mouse	-

3.5.2 Analisis Perangkat Lunak

Spesifikasi perangkat lunak (*software*) yang dibutuhkan untuk menguji pengimplementasian algoritma IDA* dan IDA* SNC pada *game* labirin ini dapat dilihat pada Tabel 3.3.

Tabel 3.3 Kebutuhan Perangkat Lunak

Komputer	Spesifikasi perangkat lunak
Sistem Operasi	*Windows 7*
Bahasa Pemograman	*Java JDK 1.7*
IDE	*Eclipse*

3.6 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem dapat berjalan dengan baik sesuai kebutuhan. Analisis yang dilakukan dimodelkan dengan menggunakan UML (*Unified Modeling Language*). Tahapan pemodelan dalam analisis tersebut antara lain pembuatan *use case diagram*, definisi aktor, definisi *use case*, skenario *use case*, *activity diagram*, *sequence diagram*, dan *class diagram*. Kebutuhan fungsional yang dibutuhkan untuk mengimplementasikan algoritma dalam penelitian ini terlampir pada Lampiran A.