KELOMPOK 6

FINAL PROJECT

PENYUSUN

Randi Palguna Artayasa
5025231020

I Gusti Ngurah Arya Sudewa
5025231030

Kadek Fajar Pramartha Yasodana
5025231185

STUDI KASUS

Sudoku

Sudoku adalah permainan mengisikan angka-angka dari 1 sampai n ke dalam cell sedemikian hingga tidak ada angka yang sama dalam satu baris, satu kolom, dan satu sub kotak.

Adapun permasalah yang kita ambil disini adalah permainan sudoku 4x4, dan 9x9

Sudoku

```
[8] 0 4 6  0 5 0  8 2 3   001101111 010111110 011001110
[7] 0 0 8  0 9 6  7 0 4   010001010 111101000 100111000
[6] 0 0 0  0 0 4  0 0 0   011011000 000001000 010101000

[5] 3 5 0  0 7 2  0 4 0   110111011 001011110 000001001
[4] 0 6 0  0 0 1  0 0 0   101010010 000100001 101000011
[3] 0 0 7  0 0 9  0 0 1   100001000 101000001 001110100

[2] 0 9 0  4 0 5  0 8 7   011101000 111011000 011111010
[1] 8 0 0  0 0 0  4 0 2   100111100 010001010 110011010
[0] 7 3 4  9 2 8  5 0 6   011000100 111111110 111001100

Fitness : 38
Penalty : 0
```

```
[3] 0 0  0 0   0010 0000 0110
[2] 0 0  3 2   0100 0110 0000

[1] 4 3  0 0   0100 1100 0000
[0] 0 0  0 0   1000 0000 1100

Fitness : 4
Penalty : 0
```

METODE YANG DIGUNAKAN

Metode

Untuk metode yang akan digunakan merupakan metode berupa algoritma local search antara lain:

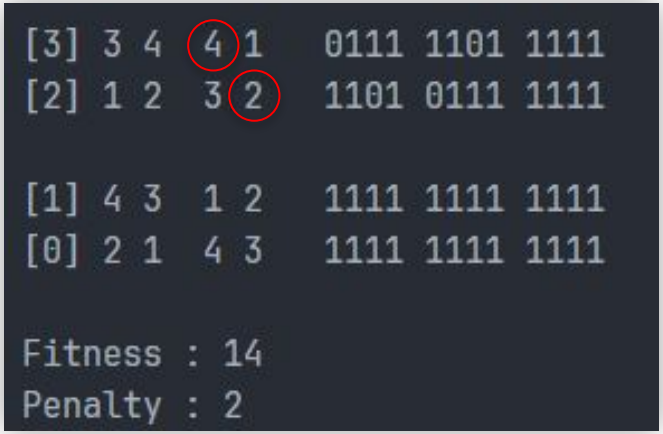
- Hill Climbing
- Simulated Annealing
- Genetic

Untuk semua implementasi kode kami menggunakan c++ akan disimpan pada
GitHub

Evaluation

Untuk mendapatkan Evaluation E(x) / Fitness function, disini kami menghitung cell yang dimana cell tersebut jika saat ditambahkan pada sudoku, jika tidak mengganggu cell yang lainnya akan menambahkan nilai sebesar 1, sedangkan jika mengganggu tidak akan menambahkan nilai. Memulainya evaluasi dari kiri bawah, menuju kanan

Evaluation



Bisa dilihat dari testcase disamping, terdapat dua cell yang mengganggu cell lainnya. Sehingga total nilai yang didapatkan adalah 14.

ITS
Institut
Teknologi
Sepuluh Nopember

ITS
BERGERAK

HILL CLIMBING

INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA

Penjelasan

Hill Climbing merupakan algoritma local search yang bertujuan untuk menemukan solusi optimal dengan melakukan iterasi terhadap solusi yang sudah ada dan memperbaikinya secara bertahap. Dalam konteks Sudoku, Hill Climbing dapat digunakan untuk menyelesaikan puzzle dengan cara membuat perubahan kecil pada grid hingga mencapai solusi yang valid.

Alur Algoritma

1. Mulai dengan solusi awal, isi cell kosong dengan angka acak.
2. Definisikan fungsi evaluasi yang menghitung jumlah konflik pada grid.
3. Periksa apakah solusi saat ini sudah optimal (fitness = 0). Jika ya, selesai.
4. Pilih cell dengan konflik dalam grid.
5. Ganti angka di titik tersebut dengan angka lain yang memungkinkan, satu per satu.
6. Hitung fungsi evaluasi pada perubahan yang dilakukan.
7. Jika perubahan mengurangi konflik, simpan perubahan tersebut.
8. Ulangi langkah 4 hingga 7 sampai menemukan solusi optimal atau mencapai batas iterasi.

DEMO HILL CLIMBING

SIMULATED ANNEALING

Penjelasan

Simulated Annealing adalah algoritma optimasi yang dapat menyelesaikan Sudoku dengan memulai dari solusi acak dan mengevaluasi pelanggaran aturan. Kemudian membuat solusi tetangga dengan menukarkan pasangan cell yang dapat ditukar.

Algoritma akan menerima solusi baru dengan suatu probabilitas tergantung suhu yang menurun seiring waktu. Proses ini diulang hingga menemukan solusi valid atau mencapai batas iterasi, memungkinkan eksplorasi ruang solusi yang efisien.

Alur Algoritma

1. Inisialisasi: Mulai dengan solusi acak dan hitung nilai fitness (jumlah pelanggaran aturan).
2. Tentukan Suhu Awal: Lakukan beberapa iterasi untuk menghitung suhu awal berdasarkan variasi fitness.
3. Set Current Fitness: Tetapkan fitness saat ini dari solusi awal.
4. Generate Solusi Tetangga: Buat solusi baru dengan melakukan perubahan kecil pada grid.
5. Hitung Fitness Tetangga: Dapatkan nilai fitness untuk solusi tetangga.
6. Evaluasi: Jika fitness tetangga lebih baik, terima solusi tersebut. Jika lebih buruk, terima dengan probabilitas berdasarkan suhu.
7. Turunkan Suhu: Kalikan suhu dengan faktor cooling multiplier setelah setiap iterasi.
8. Ulangi: Kembali ke langkah 4 hingga mencapai jumlah iterasi maksimum atau solusi optimal ditemukan.


ITS
Institut
Teknologi
Sepuluh Nopember
ITS
BERGERAK
DEMO SIMULATED
ANNEALING

GENETIC

Penjelasan

Algoritma genetika adalah metode optimasi dan pencarian yang terinspirasi oleh proses seleksi alam dalam biologi. Algoritma ini digunakan untuk menemukan solusi optimal atau mendekati optimal untuk berbagai masalah kompleks. Genetic algorithm bekerja secara melakukan

- Selection : Pemilihan parent yang akan di crossover.
- Crossover : Perkawinan antara 2 parent yang menghasilkan 2 child yang memiliki gen antara 2 parent
- Mutation : Sebuah perbedaan genetic yang dilakukan secara random

Penjelasan

Selain itu kami juga menambahkan sebuah pengecekan berupa plateu state. Jika terdeteksi plateu state, maka semua sample akan dilakukan randomisasi tetapi tidak besar. Dengan harapan menghasilkan sebuah state yang baru yang bisa menuju global maximum

Alur Algoritma

- Buatlah sebuah populasi dengan setiap sample memiliki state yang random.
- Mulailah Iterasi
- Pertama cek apakah kita sudah berada dalam plateu state, jika iya maka lakukan randomisasi
- Setelah itu kita akan memilih sample yang akan dijadikan parent, dengan melakukan tournament selection. Tournament selection kali ini mengambil 3 sample secara random dan membandingkan antara mereka yang memiliki fitness terbaik. Tetapi, walaupun sebuah sample tidak memiliki fitness lebih baik diantaranya, mereka masih bisa dipilih dengan sebuah random chance

Alur Algoritma

- Lakukan tournament selection hingga jumlah parent yang dipilih sudah setengah dari populasi awal
- Lakukan crossover antara 2 parent dengan menggunakan bitmasking crossover sehingga menghasilkan 2 child.
- 2 child jika dipilih dengan probability untuk dimutasi, maka child tersebut statenya akan dilakukan randomisasi dengan swap mutation
- Bandingkan fitness value dari child yang dihasilkan, dan jika lebih baik dari current fitness, kita set yang terbaik adalah child tersebut. Dan jika fitness sudah global maximum, kita keluar dari looping. Jika fitness masih belum global maximum kita ulangi looping lagi.


DEMO GENETIC

ANALISIS

Hill Climbing

```
4x4 Grid
Hill climbing is running
Fitness : 7
Fitness : 12
Fitness : 14
Fitness : 16
Finished in : 3166 iterations

[3] 3 2  1 4   1111 1111 1111
[2] 1 4  3 2   1111 1111 1111

[1] 4 3  2 1   1111 1111 1111
[0] 2 1  4 3   1111 1111 1111

Fitness : 16
Penalty : 0
Frame time : 16.8618
```

Untuk algoritma ini, kami mendapatkan kecepatan runtime selama ~**16.8618 milisecond** dengan ~**3166 iterasi** untuk sudoku dengan size 4x4.

Algoritma ini merupakan salah satu algoritma yang cepat dalam menyelesaikan sudoku 4x4

Kami sering memperoleh solusi optimal sebesar **~100%** dengan menjalankan algoritma ini.

Hill Climbing

```
Fitness : 69
Fitness : 71
Fitness : 72
Fitness : 73
Fitness : 75
Fitness : 76
Fitness : 77
Fitness : 78
Fitness : 79
Fitness : 81
Finished in : 647402 iterations

[8] 9 4 6  1 5 7  8 2 3   111111111 111111111 111111111
[7] 1 2 8  3 9 6  7 5 4   111111111 111111111 111111111
[6] 5 7 3  2 8 4  1 6 9   111111111 111111111 111111111

[5] 3 5 1  6 7 2  9 4 8   111111111 111111111 111111111
[4] 4 6 9  8 3 1  2 7 5   111111111 111111111 111111111
[3] 2 8 7  5 4 9  6 3 1   111111111 111111111 111111111

[2] 6 9 2  4 1 5  3 8 7   111111111 111111111 111111111
[1] 8 1 5  7 6 3  4 9 2   111111111 111111111 111111111
[0] 7 3 4  9 2 8  5 1 6   111111111 111111111 111111111

Fitness : 81
Penalty : 0
Frame time : 4473.25
```

Untuk algoritma ini, kami mendapatkan kecepatan runtime selama ~**4473.25 milisecond** dengan ~**647402 iterasi** untuk sudoku dengan size 9x9.

Algoritma ini tidak secepat algoritma Simulated Annealing, tapi lebih cepat daripada algoritma Genetic untuk menyelesaikan permasalahan sudoku 9x9.

Namun, solusi optimal hanya muncul sebesar **~30%** dengan menjalankan algoritma ini.

Simulated Annealing

```
4x4 Grid
Simulated annealing is running
Start Temperature : 1.85742
Fitness : 10 , Temperature : 1.85742
Fitness : 12 , Temperature : 1.83884
Fitness : 13 , Temperature : 1.82045
Fitness : 12 , Temperature : 1.78423
Fitness : 11 , Temperature : 1.69678
Fitness : 13 , Temperature : 1.5657
Fitness : 12 , Temperature : 1.47407
Fitness : 13 , Temperature : 1.41599
Fitness : 13 , Temperature : 1.37393
Fitness : 14 , Temperature : 1.34659
Fitness : 14 , Temperature : 1.28059
Fitness : 13 , Temperature : 1.04741
Fitness : 14 , Temperature : 0.865335
Fitness : 12 , Temperature : 0.556074
Fitness : 14 , Temperature : 0.454817
Fitness : 16 , Temperature : 0.187817
Finished in : 230 iterations

[3] 3 2  1 4   1111 1111 1111
[2] 1 4  3 2   1111 1111 1111

[1] 4 3  2 1   1111 1111 1111
[0] 2 1  4 3   1111 1111 1111

Fitness : 16
Penalty : 0
Frame time : 13.4033
```

Untuk algoritma ini, kami mendapatkan kecepatan runtime selama ~**13 milisecond** dengan ~**230 iterasi** untuk sudoku dengan size 4x4.

Untuk menyelesaikan sudoku 4x4, algoritma ini adalah **salah satu** algoritma **tercepat** yang mendekati kecepatan algoritma genetic.

Kami sering memperoleh solusi optimal sebesar ~100% dengan menjalankan algoritma ini.

Simulated Annealing

```
Fitness : 68 , Temperature : 0.0451437
Fitness : 69 , Temperature : 0.034415
Fitness : 70 , Temperature : 0.00165416
Fitness : 71 , Temperature : 0.00111776
Fitness : 72 , Temperature : 0.000210762
Fitness : 73 , Temperature : 7.2631e-05
Fitness : 74 , Temperature : 1.78036e-06
Fitness : 75 , Temperature : 7.43413e-08
Fitness : 76 , Temperature : 1.50951e-21
Fitness : 77 , Temperature : 2.69679e-28
Fitness : 79 , Temperature : 8.25942e-41
Fitness : 81 , Temperature : 2.42092e-322
Finished in : 82026 iterations

[8] 9 4 6  1 5 7  8 2 3   111111111 111111111 111111111
[7] 1 2 8  3 9 6  7 5 4   111111111 111111111 111111111
[6] 5 7 3  2 8 4  1 6 9   111111111 111111111 111111111

[5] 3 5 1  6 7 2  9 4 8   111111111 111111111 111111111
[4] 4 6 9  8 3 1  2 7 5   111111111 111111111 111111111
[3] 2 8 7  5 4 9  6 3 1   111111111 111111111 111111111

[2] 6 9 2  4 1 5  3 8 7   111111111 111111111 111111111
[1] 8 1 5  7 6 3  4 9 2   111111111 111111111 111111111
[0] 7 3 4  9 2 8  5 1 6   111111111 111111111 111111111

Fitness : 81
Penalty : 0
Frame time : 767.652
```

Untuk algoritma ini, kami mendapatkan kecepatan runtime selama ~**767.652** milisecond dengan ~**82026 iterasi** untuk sudoku dengan size 9x9.

Algoritma ini adalah algoritma **tercepat** untuk menyelesaikan permasalahan sudoku 9x9.

Kami sering memperoleh solusi optimal sebesar ~90% dengan menjalankan algoritma ini.

Genetic

```
4x4 Grid
Genetic is running
Fitness : 8 , Gen : 1
Fitness : 9 , Gen : 1
Fitness : 12 , Gen : 1
Fitness : 14 , Gen : 1
Fitness : 16 , Gen : 7
Finished in : 7 iterations

[3] 3 2  1 4   1111 1111 1111
[2] 1 4  3 2   1111 1111 1111

[1] 4 3  2 1   1111 1111 1111
[0] 2 1  4 3   1111 1111 1111

Fitness : 16
Penalty : 0
Frame time : 6.748

------------------------------------------------
```

Pada genetic algorithm kami mendapatkan untuk grid 4x4, algoritma ini paling cepat untuk menemukan solusi dengan rata rata iterasi yang paling kecil diantara algoritma lainnya. Untuk seringnya mendapatkan solusi optimal kami mendapatkan ~**100% mendapatkan solusi optimal.** Iterasi ~**7**, dalam ~**6.748ms**

Genetic

```
Fitness : 75 , Gen : 300245
Fitness : 77 , Gen : 300670
Plateud : 74
Fitness : 76 , Gen : 400026
Fitness : 77 , Gen : 401123
Fitness : 78 , Gen : 410870
Plateud : 72
Fitness : 74 , Gen : 500022
Fitness : 75 , Gen : 500028
Fitness : 76 , Gen : 500034
Fitness : 77 , Gen : 500202
Fitness : 78 , Gen : 500393
Fitness : 79 , Gen : 502488
Fitness : 81 , Gen : 512294
Finished in : 512294 iterations

[8] 9 4 6  1 5 7  8 2 3   111111111 111111111 111111111
[7] 1 2 8  3 9 6  7 5 4   111111111 111111111 111111111
[6] 5 7 3  2 8 4  1 6 9   111111111 111111111 111111111

[5] 3 5 1  6 7 2  9 4 8   111111111 111111111 111111111
[4] 4 6 9  8 3 1  2 7 5   111111111 111111111 111111111
[3] 2 8 7  5 4 9  6 3 1   111111111 111111111 111111111

[2] 6 9 2  4 1 5  3 8 7   111111111 111111111 111111111
[1] 8 1 5  7 6 3  4 9 2   111111111 111111111 111111111
[0] 7 3 4  9 2 8  5 1 6   111111111 111111111 111111111

Fitness : 81
Penalty : 0
Frame time : 272860

--------------------------------------------------
```

Untuk grid yang besar menggunakan algoritma genetic performanya sedikit lebih buruk dibandingkan simulated annealing dan lebih baik dibandingkan hill climbing. Untuk seringnya mendapatkan solusi optimal kami mendapatkan ~**75%** **mendapatkan solusi optimal.** Iterasi ~**512294**, dalam ~**272860ms (272.86 detik)**

Referensi:

Lewis, R. Metaheuristics can solve sudoku puzzles. *J Heuristics* **13**, 387–401 (2007). https://doi.org/10.1007/s10732-007-9012-8

Challenging Luck. (2020, Oktober 2). *Simulasi Annealing Dijelaskan Dengan Memecahkan Sudoku* [Video]. YouTube. https://www.youtube.com/watch?v=FyyVbuLZav8&t=1s

RUSSELL, S. J., & NORVIG, P. (2021). Artificial Intelligence: A modern approach Stuart J. Russell and Peter Norvig ; contributing writers: Ming-Wei Chang, Jacob Devlin, Anca Dragan, . Pearson Education.

TERIMA KASIH