Gebze Technical University

Computer Engineering

CSE424 – 2019

POJECT PAPER

Serkan Sorman
151044057

Problem Özeti

Magic square, her kutusunda farklı bir sayı olan ve tüm satır, sütun ve köşegenin toplamı magic number olarak tanımlanan constant değere eşit olan kare matristir. 3×3'lük bir sihirli kare, 1'den 9'a kadar olan tüm rakamları, 4×4'lük bir kare ise 1'den 16'ya kadar olan sayıları içerir. **Magic number** değeri **M(n) = n(n^2 + 1)/2** formülü ile hesaplanır. Yani 3×3'lük bir kare için, M(3)=3(3^2+1)/2=15 olur.

4	9	2
3	5	7
8	1	6

Brute force yöntemi kullanarak boyutu verilen matrise ait magic square lardan birinin bulunmasının time complexitysi O((n^2)!) dir. 4x4 bir matris için bu değer 16! dir. 4x4 ve üzeri tüm matrislerde brute force yöntemi ile magic square bulunması çok uzun süreler alır.

Genetik Algoritması

1) Algoritma Yapıları

Choromosome: Her kromozom bir çözüm matrisini tek boyutlu array olarak tutar.

Örneğin: [1, 2, 3, 4, 5, 6, 7, 8, 9] 1 2 3 matrisini ifade eder.

4 5 6

7 8 9

Ayrıca her kromozom bir fitness değerine sahiptir.

1) Fitness değeri hesaplanırken öncelikle magic number değeri hesaplanır. (M)

2) Her kolon,satır ve köşegenin toplam değeri elde edilir. (S)

3) Tüm kolon,satır ve köşegenlerin N = |M-S| değeri hesaplanır ve toplanır.
Elde edilen değer fitness değeridir. Fitness değeri 0 olan
çözüm magic squaredir.

2) Parametreler

Size: Magic Square ın boyutu

Population size: Populasyon içinde bulunacak chromosome sayısıdır.

Max iteration: Populasyondan üretilecek nesil sayısını belirtir.

Elite size: Populasyonda bulunan en iyi kromozomlardan kaçının diğer nesle direkt olarak aktarılacağını belirtir. Populasyon boyutundan küçük olmalıdır. Aksi halde bir sonraki nesilde yeni çözümler üretilemez.

Elite death period: Populasyondaki en iyi çözüme sahip kromozomların kaç nesil sonra yok edileceğini belirtir.

Mutation probability: Her nesilde populasyondaki kromozomlar üzerinde yüzde kaç ihtimalle mutasyon gerçekleştirileceğini belirtir.

3) Yöntemler

A) Selection

Tournament selection yöntemi kullanılmıştır. Populasyondan random olarak seçilen iki kromozomdan fitness değeri küçük olan alınarak mating poola eklenir. Oluşturulan mating poolun boyutu populasyon boyutunun yarısıdır.

B) Crossover

One point crossover yöntemi seçilmiştir. Fakat crossover işlemi gerçekleştirildikten sonra oluşan childların çoğunluğu unique değerlere sahip olmadığı görülmüştür. Bu sebeple elemanları unique olmayan çözümler üretilmesini büyük oranda engellemeye çalışan bir algoritma kullanıldıktan sonra One point crossover gerçekleştirilmiştir. Kullanılan algoritma TSP çözümünde kullanılan crossover algoritmasıdır. Buradan kullanılan algoritmaya erişebilirsiniz.

C)Mutation

Mutation probability parametresindeki orana göre kromozomun sahip olduğu çözüm arrayinde random olarak seçilen 2 nokta swap edilir.

4) Deneyler ve sonuçlarının yorumlanması

1) 3x3 Matris üzerinde yapılan deneylerde brute-force algoritmasına göre farklı bir performans görülmemiştir. 9! Değeri bilgisayarlar için büyük bir değer olmadığından Genetik algoritmasının performansının bu değere göre belirlenmesi doğru olmaz. 3x3 Matrisler aşağıda belirtilen parametrlerle test edildiğinde çözüme ulaşma süresinin 300 – 600 ms aralığında olduğu görülmüştür.

```
GeneticAlgorithm magicSquareFinder = new GeneticAlgorithm( size: 3, maxIteration: 5000,
        populationSize: 10, eliteSize: 5, eliteDeathPeriod: 1000, mutationProbability: 0.5);
```

```
Generation number: 89
4   9   2
3   5   7
8   1   6
Search is completed
Total time: 274ms
```

2) 4x4 matris üzerinde düşük population boyutları ile aşağıdaki parametreler kullanılarak yapılan testlerde çözüme ulaşma sürelerinin 500 – 1500 ms aralığında değiştiği görülmüştür.

```
GeneticAlgorithm magicSquareFinder = new GeneticAlgorithm( size: 4, maxIteration: 20000,
        populationSize: 10, eliteSize: 8, eliteDeathPeriod: 2000, mutationProbability: 0.5);
```

```
Generation number: 3278
7   1   14  12
10  16  3   5
4   6   9   15
13  11  8   2
Search is completed
Total time: 768ms
```

3) 4x4 matris üzerinde daha yüksek population boyutları ile aşağıdaki parametreler kullanılarak yapılan testlerde çözüme ulaşma sürelerinin 5000 – 9000 ms aralığında değiştiği görülmüştür.

```
GeneticAlgorithm magicSquareFinder = new GeneticAlgorithm( size: 4, maxIteration: 20000,
        populationSize: 200, eliteSize: 180, eliteDeathPeriod: 0, mutationProbability: 0.5);
```

```
Generation number: 2234
15  8   9   2
10  1   16  7
4   11  6   13
5   14  3   12
Search is completed
Total time: 7308ms
```

4) 5x5 matris üzerinde düşük population boyutları ile aşağıdaki parametreler kullanılarak yapılan testlerde çözüme ulaşma sürelerinin 11– 40 saniye aralığında değiştiği görülmüştür.

```
GeneticAlgorithm magicSquareFinder = new GeneticAlgorithm( size: 5, maxIteration: 1000000,
        populationSize: 10, eliteSize: 8, eliteDeathPeriod: 2000, mutationProbability: 0.5);
```

```
Generation number: 198170
6   18  22  15  4
8   10  16  19  12
24  21  17  2   1
13  11  7   9   25
14  5   3   20  23
Search is completed
Total time: 11305ms
```

5) 5x5 matris üzerinde daha yüksek populasyon boyutları ile aşağıdaki parametreler kullanılarak yapılan testlerde çözüme ulaşma sürelerinin 24 – 90 saniye aralığında değiştiği görülmüştür.

```
GeneticAlgorithm magicSquareFinder = new GeneticAlgorithm( size: 5, maxIteration: 1000000,
        populationSize: 200, eliteSize: 180, eliteDeathPeriod: 2000, mutationProbability: 0.5);
```

```
Generation number: 5444
24  14  6   2   19
5   13  17  23  7
15  11  4   25  10
20  18  16  3   8
1   9   22  12  21
Search is completed
Total time: 24629ms
```

6) 6x6 matris üzerinde daha yüksek populasyon boyutları ile aşağıdaki parametreler kullanılarak yapılan testlerde çözüme ulaşma sürelerinin 114 – 200 saniye aralığında değiştiği görülmüştür.

```
GeneticAlgorithm magicSquareFinder = new GeneticAlgorithm( size: 6, maxIteration: 2000000,
        populationSize: 20, eliteSize: 10, eliteDeathPeriod: 60000, mutationProbability: 0.5);
```

```
Generation number: 291719
35  33  4   3   12  24
20  18  25  15  2   31
19  14  6   23  17  32
29  8   34  13  22  5
7   27  16  21  30  10
1   11  26  36  28  9
Search is completed
Total time: 114013ms
```

Yapılan testlerde populasyon boyutları azaldıkça daha kısa sürede çözüme ulaşıldığı görülmüştür. Populasyon boyutu düşük örneklerde, fazla populasyon sayısına sahip örneklere göre aynı süre içerisinde daha fazla nesil üretme imkanı oluşmuştur.

Tabu Search

1) Algoritma Yapıları

Solution: Genetik algoritmasında bulunan chromosome yapısı ile aynı görevi üstlenir.

TabuList: Başlangıçta belirlenen boyuta göre oluşturulan bir solution listesidir. Daha önceden elde edilen çözümler bu listeye eklenir. Böylece bir hafıza yapısı oluşturulmuş olur. Neighborhood oluşturulurken tabu listesinde bulunan solutionlar eklenmez. Tabu listesinin boyutu maksimum boyuta ulaştığında listenin ilk elemanı silinir ve yeni eleman listenin sonuna eklenir.

2) Parametreler

Size: Magic Square ın boyutu

Neighborhod size: Best solutiona göre oluşturulan neighborlardan oluşan listenin boyutunu belirtir.

Max iteration: Üretilecek neighborhood sayısını ve local bestin kaç iterasyon daha aranacağını belirtir.

TabuListMaxLength: Tabu listesinin hafızada tutabileceği maksimum solution sayısını belirtir.

MaxIterationWithoutBetterSolution: Daha iyi bir çözüm bulunmadan kaç iterasyon devam edileceğini belirtir.

MaxTries: Tabu search işleminin kaç defa tekrardan başlatılacağını belirtir.

3) Deneyler ve sonuçlarının yorumlanması

1) 4x4 matris üzerinde düşük neighborhood boyutları ile aşağıdaki parametreler kullanılarak yapılan testlerde çözüme ulaşma sürelerinin 250 – 1000 ms aralığında değiştiği görülmüştür.

```
TabuSearch tabuSearch = new TabuSearch( size: 4, maxTries: 10, maxIteration: 5000,
        maxIterationWithoutBetterSolution: 2000, tabuListMaxLength: 50, neighborhoodSize: 10);
```

```
9   3   8   14
6   16  11  1
15  5   2   12
4   10  13  7

Total time: 260ms
```

2) 4x4 matris üzerinde daha yüksek neighborhood ve tabu list boyutları ile aşağıdaki parametreler kullanılarak yapılan testlerde çözüme ulaşma sürelerinin 300ms – 19 saniye aralığında değiştiği görülmüştür.

```
TabuSearch tabuSearch = new TabuSearch( size: 4, maxTries: 10, maxIteration: 5000,
        maxIterationWithoutBetterSolution: 2000, tabuListMaxLength: 200, neighborhoodSize: 200);
```

```
1   6   11  16
12  15  2   5
8   3   14  9
13  10  7   4

Total time: 13833ms
```

```
8   11  2   13
14  12  7   1
3   5   10  16
9   6   15  4

Total time: 313ms
```

2) 5x5 matris üzerinde düşük neighborhood ile aşağıdaki parametreler kullanılarak yapılan testlerde çözüme ulaşma sürelerinin 14 – 100 saniye aralığında değiştiği görülmüştür. Bazı durumlarda belirtilen iterasyon sayılarında çözüme ulaşılamamıştır.

```
TabuSearch tabuSearch = new TabuSearch( size: 5, maxTries: 100, maxIteration: 1000000,
        maxIterationWithoutBetterSolution: 50000, tabuListMaxLength: 10, neighborhoodSize: 10);
```

```
24  16  4   13  8
15  7   17  23  3
20  2   10  11  22
1   19  25  6   14
5   21  9   12  18

Total time: 26172ms
```

2) 5x5 matris üzerinde daha yüksek neighborhood ve tabu list boyutları ile aşağıdaki parametreler kullanılarak yapılan testlerde çözüme ulaşma sürelerinin 18 – 45 saniye aralığında değiştiği görülmüştür. Bazı durumlarda belirtilen iterasyon sayılarında çözüme ulaşılamamıştır.

```
TabuSearch tabuSearch = new TabuSearch( size: 5, maxTries: 10, maxIteration: 100000,
        maxIterationWithoutBetterSolution: 5000, tabuListMaxLength: 50, neighborhoodSize: 200);
```

```
16  24  5   18  2
10  1   22  15  17
3   7   23  20  12
25  14  9   4   13
11  19  6   8   21

Total time: 20158ms
```

TS nin düşük boyutlu matrislerde GA ya göre daha hızlı olduğu görülüyor. Fakat matris boyutu büyüyünce TS nin çözüm bulma zaman aralığı büyük sapmalar gösteriyor. GA için tüm matris boyutlarında düşük populasyon boyutları ile daha iyi performans alındığı görülüyor. TS ise düşük boyutlu matrislerde daha az neighborhood sizeları ile daha kısa sürede çözüme ulaşırken matris boyutu arttıkça daha yüksek neighborhood boyutları daha kısa sürede çözüme ulaşmayı sağlayabiliyor.