محمدرضا جبلی ۹۴۳۱۰۳۵

تمرین اول برنامه نویس هوش مصنوعی

نحوه ی مدلسازی :
از یک کلاس search که تمامی حالات مختلف search در آن به صورت تابع قرارداده شده استفاده شده. این توابع به هیچ وجه نباید به نحوه ی پیاده سازی کلاس های problem , state , action ارتباط داشته باشند.
سپس هر مسیله را با استفاده از کلاس های problem , action , state مدل شده است که هر کلاس های action , state از کلاس های پدرشان ارث برده اند که کد دسته بندی شود و کار ها مشخص شود . کلاس problem نیز از یک interface , ایمپلمنت کرده است.

مسئله ۱ : ربات امدادگر

عکس های گرفته شده به ترتیب برای الگوریتم های
uniformCost , dfs_graph , bidirectional , A*
میباشد.
طبق آنچه مشاهده میشود الگوریتم A* کمترین حافظه را مصرف کرده و همه ی الگوریتم ها نیز به جز اول عمق جواب بهینه را پیدا کرده اند.

```java
public class Main {
    public static void main(String[] args) {
        Problem1 problem = new Problem1();
        Search s = new Search();
        s.uniformCost(problem);
    }
}
```

Problems @ Javadoc Declaration Console

```
<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/java (Khordad 19, 1397 AP, 10:4
5 5
4
3 2 4 2 3 3 4 3 2 3 2 4 3 3 3 4
numOfObservedNodes: 25
numOfExtendedNodes: 24
max memory: 24
pathStates : ( 1 , 1 ) ( 2 , 1 ) ( 3 , 1 ) ( 4 , 1 ) ( 5 , 1 ) ( 5 , 2 ) ( 5 , 3 ) ( 5 , 4 ) ( 5 , 5 )
pathActions : r r r r d d d d
pathCost : 8.00.0
```

```java
public class Main {
    public static void main(String[] args) {
        Problem1 problem = new Problem1();
        Search s = new Search();
        s.dfs_graph(problem);
    }
}
```

Problems @ Javadoc Declaration Console

```
<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/java (Khordad 19, 1397 AP, 10:44:29 PM)
5 5
4
3 2 4 2 3 3 4 3 2 3 2 4 3 3 3 4
numOfObservedNodes: 25
numOfExtendedNodes: 19
max memory: 24
pathStates : ( 1 , 1 ) ( 1 , 2 ) ( 1 , 3 ) ( 1 , 4 ) ( 1 , 5 ) ( 2 , 5 ) ( 3 , 5 ) ( 3 , 4 ) ( 4 , 4 ) ( 5 , 4 ) ( 5 , 5 )
pathActions : d d d d r r u r r d
pathCost : 10.00.0
```

```java
public class Main {
    public static void main(String[] args) {
        Problem1 problem = new Problem1();
        Search s = new Search();
        s.bidirectional(problem);
    }
}
```

Problems Javadoc Declaration Console

<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/java (Khordad 19, 1397 AP, 10:45

```
5 5
4
3 2 4 2 3 3 4 3 2 3 2 4 3 3 3 4
numOfObservedNodes: 26
numOfExtendedNodes: 18
max memory: 25
pathStates : ( 1 , 1 ) ( 2 , 1 ) ( 3 , 1 ) ( 4 , 1 ) ( 4 , 2 ) ( 4 , 3 ) ( 4 , 4 ) ( 4 , 5 ) ( 5 , 5 )
pathActions : r r r d d d d r
pathCost : 8
```

```java
public class Main {
    public static void main(String[] args) {
        Problem1 problem = new Problem1();
        Search s = new Search();
        s.A_Star(problem);
    }
}
```

Problems Javadoc Declaration Console

<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/java (Khordad 19, 1397 AP, 10:45

```
5 5
4
3 2 4 2 3 3 4 3 2 3 2 4 3 3 3 4
numOfObservedNodes: 19
numOfExtendedNodes: 14
max memory: 18
pathStates : ( 1 , 1 ) ( 1 , 2 ) ( 1 , 3 ) ( 1 , 4 ) ( 2 , 4 ) ( 3 , 4 ) ( 4 , 4 ) ( 4 , 5 ) ( 5 , 5 )
pathActions : d d d r r r d r
pathCost : 8.00.0
```

مسئله ۲ : پازل ۸ تایی

```java
public class Main {
    public static void main(String[] args) {
        Problem2 problem = new Problem2();
        Search s = new Search();
        s.uniformCost(problem);
    }
}
```

Problems @ Javadoc Declaration Console

<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/jav

```
3 4 1
6 0 2
7 8 5
numOfObservedNodes: 508
numOfExtendedNodes: 314
max memory: 507
pathStates :
3 4 1
6 0 2
7 8 5

3 0 1
6 4 2
7 8 5

3 1 0
6 4 2
7 8 5

3 1 2
6 4 0
7 8 5

3 1 2
6 4 5
7 8 0

3 1 2
6 4 5
7 0 8

3 1 2
6 4 5
0 7 8

3 1 2
0 4 5
6 7 8

0 1 2
3 4 5
6 7 8

pathActions : u r d d l l u u
pathCost : 8.00.0
```

```java
public class Main {
    public static void main(String[] args) {
        Problem2 problem = new Problem2();
        Search s = new Search();
        s.dfs_graph(problem);
    }
}
```

Problems @ Javadoc Declaration Console

<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/jav

```
3 4 1
6 0 2
7 8 5
numOfObservedNodes: 18241
numOfExtendedNodes: 10308
max memory: 18240

pathActions : u l d d r u u l d d r u u l d d r u u l d d r u u l d r u l d d r u u l d d r u u r d d l u u l
pathCost : 10034.00.0
```

```java
public class Main {
    public static void main(String[] args) {
        Problem2 problem = new Problem2();
        Search s = new Search();
        s.bidirectional(problem);
    }
}
```

Problems @ Javadoc Declaration Console

<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jd

```
3 4 1
6 0 2
7 8 5
numOfObservedNodes: 72
numOfExtendedNodes: 39
max memory: 71
pathStates :
3 4 1
6 0 2
7 8 5

3 0 1
6 4 2
7 8 5

3 1 0
6 4 2
7 8 5

3 1 2
6 4 0
7 8 5

3 1 2
6 4 5
7 8 0

3 1 2
6 4 5
7 0 8

3 1 2
6 4 5
0 7 8

3 1 2
0 4 5
6 7 8

0 1 2
3 4 5
6 7 8

pathActions : u r d d l l u u
pathCost : 8
```

```java
public class Main {
    public static void main(String[] args) {
        Problem2 problem = new Problem2();
        Search s = new Search();
        s.A_Star(problem);
    }
}
```

Problems @ Javadoc Declaration Console

<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Conte

```
3 4 1
6 0 2
7 8 5
numOfObservedNodes: 315
numOfExtendedNodes: 189
max memory: 314
pathStates :
3 4 1
6 0 2
7 8 5

3 0 1
6 4 2
7 8 5

3 1 0
6 4 2
7 8 5

3 1 2
6 4 0
7 8 5

3 1 2
6 4 5
7 8 0

3 1 2
6 4 5
7 0 8

3 1 2
6 4 5
0 7 8

3 1 2
0 4 5
6 7 8

0 1 2
3 4 5
6 7 8

pathActions : u r d d l l u u
pathCost : 8.00.0
```

عکس های گرفته شده به ترتیب برای الگوریتم های

uniformCost , dfs_graph , bidirectional , A*

میباشد.

طبق آنچه مشاهده میشود الگوریتم bidirectional کمترین حافظه را مصرف کرده و در طرف دیگر نیز میزان حافظه ی dfs گرافی به شدت بالا است. و همه ی الگوریتم ها نیز به جز اول عمق جواب بهینه را پیدا کرده اند. که جواب الگوریتم dfs به شدت غیر بهینه است.

مسیله سوم : مکعب روبیک

عکس های گرفته شده به ترتیب برای الگوریتم های

افزایش تدریجی عمق dfs , عمق محدود bfs , dfs

میباشد.

همه ی الگوریتم ها جواب بهینه داده اند منتها مشاهده میشود که الگوریتم اول عمق افزایش تدریج عمق به دلیل اینکه در هر مرحله حافظه ی تخصیص داده شده را رها میکند کمترین حافظه را اختصاص داده است.

```
1  public class Main {
2      public static void main(String[] args) {
3          Problem3 problem = new Problem3();
4          Search s = new Search();
5          s.bfs(problem);
6      }
7  }
```

Problems @ Javadoc Declaration Console

```
<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/java (Khordad 19, 1397 AP, 11:16:2
y b y b g y g y w g w g b w b w r r r r o o o o o
numOfObservedNodes: 6
numOfExtendedNodes: 1
max memory: 5
pathStates : (y b y b g y g y w g w g b w b w r r r r o o o o o ) (y y y y g g g g w w w w b b b b r r r r o o o o o )
pathActions : R
pathCost : 1.00.0
```

```java
public class Main {
    public static void main(String[] args) {
        Problem3 problem = new Problem3();
        Search s = new Search();
        s.dfs_limitedDepth_graph(problem , 14);
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/java (Khordad 19, 1397 AP, 11:17:
y b y b g y g y w g w g b w b w r r r r o o o o
numOfObservedNodes: 6
numOfExtendedNodes: 1
max memory: 5
pathStates : (y b y b g y g y w g w g b w b w r r r r o o o o ) (y y y y g g g g w w w w b b b b r r r r o o o o )
pathActions : R
pathCost : 1.00.0
```

```java
public class Main {
    public static void main(String[] args) {
        Problem3 problem = new Problem3();
        Search s = new Search();
        s.dfs_depthIncremental_graph(problem , 14);
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Main [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/java (Khordad 19, 1397 AP, 11:17:3
y b y b g y g y w g w g b w b w r r r r o o o o
numOfObservedNodes: 2
numOfExtendedNodes: 1
max memory: 1
pathStates : (y b y b g y g y w g w g b w b w r r r r o o o o ) (y y y y g g g g w w w w b b b b r r r r o o o o )
pathActions : R
pathCost : 1.00.0
```

جمع بندی:

الگوریتم bidirectional سریع ترین الگوریتم بود به همراه A*

همچنین A* کمترین حافظ را مصرف کرده(به همراه اول عمق درختی) و اول عمق درختی نیز بیشترین مصرف حافظه را دارد همچنین سرعتش نیز بسیار کم است.