Hill-Climbing Examples

1] Given the following table of direct distances to Dallas

Austin	Charlotte	San Francisco	Los Angeles	New York	Chicago	Seattle	Santa Fe	Bakersville	Boston
182	929	1230	1100	1368	800	1670	560	1282	1551

and the road graph of U.S.A. map (in miles):

```
Los Angeles ---        San Francisco  ::: 383
Los Angeles ---        Austin         ::: 1377
Los Angeles ---        Bakersville    ::: 153
San Francisco ---      Bakersville    ::: 283
San Francisco ---      Seattle        ::: 807
Seattle ---            Santa Fe       ::: 1463
Seattle ---            Chicago        :::  2064
Bakersville ---        Santa Fe       ::: 864
Austin ---             Dallas         ::: 195
Santa Fe ---           Dallas         ::: 640
Boston ---             Austin         ::: 1963
Dallas ---             New York       ::: 1548
Austin ---             Charlotte      ::: 1200
Charlotte --           New York       ::: 634
New York ---           Boston         ::: 225
Boston ---             Chicago        ::: 983
Chicago --             Santa Fe       ::: 1272
Boston ---             San Francisco  ::: 3095
```

Use hill-climbing to find a path from Seattle to Dallas.

SOLUTION 1: - Select an *objective function* Obj(node).

One possibility is Obj(node)=1/h(node) if node$\neq$ Dallas and 1 if node=Dallas.

STEP 1: Initial node=Seattle

Successors(Seattle) = {San Francisco, Santa Fe, Chicago}

h(San Francisco)=1230 Obj(San Francisco)=1/1230

h(Santa Fe)=560 Obj(Santa Fe)=1/560

h(Chicago)=800 Obj(Chicago)=1/800

NB - the FRINGE and the CLOSED LIST apply only to general search algorithms. Hill Climbing is a local search algorithm. Use the list NEIGHBORS and list the nodes in descending order of the value of their Objective function

NEIGHBORS = { (Santa Fe, 1/560), (Chicago, 1/800), (San Francisco, 1/1230) }

We Select the next node =Santa Fe ===> PATH={Seattle $\rightarrow$ Santa Fe} PATH COST=1463

Given the following table of direct distances to Dallas

Austin	Charlotte	San Francisco	Los Angeles	New York	Chicago	Seattle	Santa Fe	Bakersville	Boston
182	929	1230	1100	1368	800	1670	560	1282	1551

and the road graph of U.S.A. map (in miles):

```
Los Angeles ---        San Francisco  ::: 383
Los Angeles ---        Austin         ::: 1377
Los Angeles ---        Bakersville    ::: 153
San Francisco ---      Bakersville    ::: 283
San Francisco ---      Seattle        ::: 807
Seattle ---            Santa Fe       ::: 1463
Seattle ---            Chicago        :::  2064
Bakersville ---        Santa Fe       ::: 864
Austin ---             Dallas         ::: 195
Santa Fe ---           Dallas         ::: 640
Boston ---             Austin         ::: 1963
Dallas ---             New York       ::: 1548
Austin ---             Charlotte      ::: 1200
Charlotte --           New York       ::: 634
New York ---           Boston         ::: 225
Boston ---             Chicago        ::: 983
Chicago --             Santa Fe       ::: 1272
Boston ---             San Francisco  ::: 3095
```

STEP 2: Current node=Santa Fe

Successors(Santa Fe) = {Bakersville, Dallas, Chicago, Seattle}

h(Chicago)=800 Obj(Chicago)=1/800

h(Dallas)=0 Obj(Dallas)=1

h(Bakersville)=1282 Obj(Bakersville)=1/1282

h(Seattle)=1670 Obj(Seattle)=1/1670

NEIGHBORS = { (Dallas, 1), (Chicago, 1/800), (Bakersville, 1/1282), (Seattle, 1/1670) }

We Select Dallas ===> GOAL NODE <==== We have a solution*******

PATH={Seattle → Santa Fe →Dallas} PATH COST=1463+640=2103

Another Solution:

Given the following table of direct distances to Dallas

Austin	Charlotte	San Francisco	Los Angeles	New York	Chicago	Seattle	Santa Fe	Bakersville	Boston
182	929	1230	1100	1368	800	1670	560	1282	1551

and the road graph of U.S.A. map (in miles):

```
Los Angeles ---        San Francisco  ::: 383
Los Angeles ---        Austin         ::: 1377
Los Angeles ---        Bakersville    ::: 153
San Francisco ---      Bakersville    ::: 283
San Francisco ---      Seattle        ::: 807
Seattle ---            Santa Fe       ::: 1463
Seattle ---            Chicago        :::  2064
Bakersville ---        Santa Fe       ::: 864
Austin ---             Dallas         ::: 195
Santa Fe ---           Dallas         ::: 640
Boston ---             Austin         ::: 1963
Dallas ---             New York       ::: 1548
Austin ---             Charlotte      ::: 1200
Charlotte --           New York       ::: 634
New York ---           Boston         ::: 225
Boston ---             Chicago        ::: 983
Chicago --             Santa Fe       ::: 1272
Boston ---             San Francisco  ::: 3095
```

Select the Objective function as Obj(node)=10000-g(node$\leftarrow$father(node)) if node$\neq$ Dallas and Obj(Dallas)=10000.

<u>STEP 1</u>: Initial node=Seattle

Successors(Seattle) = {San Francisco, Santa Fe, Chicago}

Obj(San Francisco)=10000-807=9193

Obj(Santa Fe)=10000-1463=8537

Obj(Chicago)=10000-2064=7936

NEIGHBORS = { (San Francisco,9193), (Santa Fe, 8537), (Chicago, 7936) }

We select the next node =San Francisco

Given the following table of direct distances to Dallas

Austin	Charlotte	San Francisco	Los Angeles	New York	Chicago	Seattle	Santa Fe	Bakersville	Boston
182	929	1230	1100	1368	800	1670	560	1282	1551

and the road graph of U.S.A. map (in miles):

```
Los Angeles ---        San Francisco  ::: 383
Los Angeles ---        Austin         ::: 1377
Los Angeles ---        Bakersville    ::: 153
San Francisco ---      Bakersville    ::: 283
San Francisco ---      Seattle        ::: 807
Seattle ---            Santa Fe       ::: 1463
Seattle ---            Chicago        :::  2064
Bakersville ---        Santa Fe       ::: 864
Austin ---             Dallas         ::: 195
Santa Fe ---           Dallas         ::: 640
Boston ---             Austin         ::: 1963
Dallas ---             New York       ::: 1548
Austin ---             Charlotte      ::: 1200
Charlotte --           New York       ::: 634
New York ---           Boston         ::: 225
Boston ---             Chicago        ::: 983
Chicago --             Santa Fe       ::: 1272
Boston ---             San Francisco  ::: 3095
```

STEP 2: Current node=San Francisco

PATH={Seattle $\rightarrow$ San Francisco} PATH COST=807

Successors(San Francisco) = {Los Angeles, Bakersville, Seattle, Boston}

Obj(Los Angeles)=10000-383=9617

Obj(Bakersville)=10000-283=9717

Obj(Seattle)=10000-807=9193

Obj(Boston)=10000-3095=6905

NEIGHBORS = { (Bakersville,9717), (Los Angeles 9617), (Seattle, 9193), (Boston, 6905) }

We select the next node =Bakersville

Given the following table of direct distances to Dallas

Austin	Charlotte	San Francisco	Los Angeles	New York	Chicago	Seattle	Santa Fe	Bakersville	Boston
182	929	1230	1100	1368	800	1670	560	1282	1551

and the road graph of U.S.A. map (in miles):

```
Los Angeles ---       San Francisco  ::: 383
Los Angeles ---       Austin         ::: 1377
Los Angeles ---       Bakersville    ::: 153
San Francisco ---     Bakersville    ::: 283
San Francisco ---     Seattle        ::: 807
Seattle ---           Santa Fe       ::: 1463
Seattle ---           Chicago        :::  2064
Bakersville ---       Santa Fe       ::: 864
Austin ---            Dallas         ::: 195
Santa Fe ---          Dallas         ::: 640
Boston ---            Austin         ::: 1963
Dallas ---            New York       ::: 1548
Austin ---            Charlotte      ::: 1200
Charlotte --          New York       ::: 634
New York ---          Boston         ::: 225
Boston ---            Chicago        ::: 983
Chicago --            Santa Fe       ::: 1272
Boston ---            San Francisco  ::: 3095
```

STEP 3: Current node=Bakersville

PATH={Seattle → San Francisco→ Bakersville } PATH COST=1090

Successors(Bakersville)={Los Angeles, San Francisco, Santa Fe}

Obj(Los Angeles)=10000-153=9847

Obj(San Francisco)=10000-283=9717

Obj(Santa Fe)=10000-864=9136

NEIGHBORS = {(Los Angeles, 9847), (San Francisco,9717), (Santa Fe,9136)}

We select the next node =Los Angeles

Given the following table of direct distances to Dallas

Austin	Charlotte	San Francisco	Los Angeles	New York	Chicago	Seattle	Santa Fe	Bakersville	Boston
182	929	1230	1100	1368	800	1670	560	1282	1551

and the road graph of U.S.A. map (in miles):

```
Los Angeles ---        San Francisco   ::: 383
Los Angeles ---        Austin          ::: 1377
Los Angeles ---        Bakersville     ::: 153
San Francisco ---      Bakersville     ::: 283
San Francisco ---      Seattle         ::: 807
Seattle ---            Santa Fe        ::: 1463
Seattle ---            Chicago         :::  2064
Bakersville ---        Santa Fe        ::: 864
Austin ---             Dallas          ::: 195
Santa Fe ---           Dallas          ::: 640
Boston ---             Austin          ::: 1963
Dallas ---             New York        ::: 1548
Austin ---             Charlotte       ::: 1200
Charlotte --           New York        ::: 634
New York ---           Boston          ::: 225
Boston ---             Chicago         ::: 983
Chicago --             Santa Fe        ::: 1272
Boston ---             San Francisco   ::: 3095
```

STEP 4: Current node=Los Angeles

PATH={Seattle → San Francisco→ Bakersville→ Los Angeles } PATH COST=1243

Successors(Los Angeles)={San Francisco, Austin, Bakersville}

Obj(San Francisco)= 10000-283=9717

Obj(Austin)=10000-1377=8623

Obj(Bakersville)=10000-283=9717

NEIGHBORS = {(Bakersville,9717), (San Francisco,9717) , (Austin,8623)}

We Select Bakersville ===>BIG PROBLEM!!! Creates a cycle!! {Bakersville → Los Angeles→ Bakersville] is an infinite loop

If we select San Francisco ====? Big PROBLEM too! Creates a cycle as well: {San Francisco → Bakersville → Los Angeles → San Francisco}

What should we do???

Random selection from the other nodes of the NEIGHBORS list: next node: Austin!

Given the following table of direct distances to Dallas

Austin	Charlotte	San Francisco	Los Angeles	New York	Chicago	Seattle	Santa Fe	Bakersville	Boston
182	929	1230	1100	1368	800	1670	560	1282	1551

and the road graph of U.S.A. map (in miles):

```
Los Angeles ---        San Francisco  ::: 383
Los Angeles ---        Austin         ::: 1377
Los Angeles ---        Bakersville    ::: 153
San Francisco ---      Bakersville    ::: 283
San Francisco ---      Seattle        ::: 807
Seattle ---            Santa Fe       ::: 1463
Seattle ---            Chicago        :::  2064
Bakersville ---        Santa Fe       ::: 864
Austin ---             Dallas         ::: 195
Santa Fe ---           Dallas         ::: 640
Boston ---             Austin         ::: 1963
Dallas ---             New York       ::: 1548
Austin ---             Charlotte      ::: 1200
Charlotte --           New York       ::: 634
New York ---           Boston         ::: 225
Boston ---             Chicago        ::: 983
Chicago --             Santa Fe       ::: 1272
Boston ---             San Francisco  ::: 3095
```

STEP 4: Current node=Austin

PATH={Seattle → San Francisco→ Bakersville→ Los Angeles → Austin} PATH COST=2620

Successors (Austin)= {Los Angeles, Dallas}

Obj (Los Angeles) =10000-153=9847

Obj (Dallas)=10000

NEIGHBORS = {(Dallas, 10000), (Los Angeles, 9847)}

We select the next node =Dallas

STEP 5: Reached the goal=Dallas

PATH= {Seattle → San Francisco→ Bakersville→ Los Angeles → Austin→ Dallas}

PATH COST=2815

COMMENT: Hill-climbing may yield different solutions, depending on the objective function that is selected.

Simulated Annealing Example

> **function** SIMULATED-ANNEALING(*problem*, *schedule*) **returns** a solution state
> **inputs**: *problem*, a problem
> *schedule*, a mapping from time to "temperature"
> **local variables**: *current*, a node
> *next*, a node
> T, a "temperature" controlling prob. of downward steps
>
> *current* $\leftarrow$ MAKE-NODE(INITIAL-STATE[*problem*])
> **for** $t \leftarrow$ **1 to** ∞ **do**
> $T \leftarrow$ *schedule*[t]
> **if** $T = 0$ **then return** *current*
> *next* $\leftarrow$ a randomly selected successor of *current*
> $\Delta E \leftarrow$ VALUE[*next*] – VALUE[*current*]
> **if** $\Delta E > 0$ **then** *current* $\leftarrow$ *next*
> **else** *current* $\leftarrow$ *next* only with probability $e^{\Delta E/T}$

A/ Let us consider the same problem:

Given the following table of direct distances to Dallas

Austin	Charlotte	San Francisco	Los Angeles	New York	Chicago	Seattle	Santa Fe	Bakersville	Boston
182	929	1230	1100	1368	800	1670	560	1282	1551

and the road graph of U.S.A. map (in miles):

```
Los Angeles ---        San Francisco  ::: 383
Los Angeles ---        Austin         ::: 1377
Los Angeles ---        Bakersville    ::: 153
San Francisco ---      Bakersville    ::: 283
San Francisco ---      Seattle        ::: 807
Seattle ---            Santa Fe       ::: 1463
Seattle ---            Chicago        :::  2064
Bakersville ---        Santa Fe       ::: 864
Austin ---             Dallas         ::: 195
Santa Fe ---           Dallas         ::: 640
Boston ---             Austin         ::: 1963
Dallas ---             New York       ::: 1548
Austin ---             Charlotte      ::: 1200
Charlotte --           New York       ::: 634
New York ---           Boston         ::: 225
Boston ---             Chicago        ::: 983
Chicago --             Santa Fe       ::: 1272
Boston ---             San Francisco  ::: 3095
```

B/ We need do define a *schedule*:

T_0	T_1	T_2	T_3	T_4	T_5	T_5	T_6	T_7	T_8
1000	950	900	850	800	750	700	650	600	550

C/ We need to define an energy function *E(node)= 2000- h(node)*

<u>STEP 1</u>: Initial node=Seattle; T_0=1000

Successors (Seattle) = {San Francisco, Santa Fe, Chicago}

- Random select next node from Successors: Chicago!

E(Chicago) = 2000-800=1200

E(Seattle) = 2000-1670=330

ΔE=870>0, then next node is Chicago!

<u>STEP 2</u>: current node=Chicago; T_1=950

Successors (Chicago) = {Seattle, Boston, Santa Fe}

- Random select next node from Successors: Boston!

E(Boston) = 2000-1551 = 449

E(Chicago) = 2000-800=1200

ΔE=449-1200=-751 <0, then compute the probability $e^{-\frac{751}{950}}$, i.e. $e^{-0.79} = 0.4538$

This means that with probability 0.4538 the next city shall be Boston, otherwise we stay in Chicago.

Suppose Boston is not selected.

<u>STEP 3</u>: current node=Chicago; T_2=900

Successors (Chicago) = {Seattle, Boston, Santa Fe}

- Random select next node from Successors: Seattle!

E(Seattle) = 2000-1670 = 330

E(Chicago) = 2000-800=1200

ΔE=330-1200=-870 <0, then compute the probability $e^{-\frac{870}{900}}$, i.e. $e^{-0.966} = 0.3829$

This means that with probability 0.3829 the next city shall be Seattle, otherwise we stay in Chicago.

Suppose Seattle is not selected.

<u>STEP 4</u>: current node=Chicago; T_3=850

Successors (Chicago) = {Seattle, Boston, Santa Fe}

- Random select next node from Successors: Santa Fe!

E(Santa Fe) = 2000-560 = 1440

E(Chicago) = 2000-800=1200

ΔE=1440-1200=240>0, then next node is Santa Fe!

<u>STEP 5</u>: current node=Santa Fe; T_4=800

Successors (Santa Fe) = {Seattle, Chicago, Dallas}

- Random select next node from Successors: Dallas!

E(Dallas) = 2000

E (Santa Fe) = 2000-560 = 1440

ΔE=2000-1440=-560 > 0, then next node is Dallas!

<u>STEP 6</u>: current node=Dallas; T_5=750

Dallas has the largest value of the function E – we stop!

The path is PATH= {Seattle $\rightarrow$ Chicago $\rightarrow$ Santa Fe$\rightarrow$ Dallas}

PATH COST=3976

Random Walk Search

In simulated annealing:

- If T=∞ (or not temperature schedule!)

$$e^{-\frac{\Delta E}{T}} = e^{-\frac{\Delta E}{\infty}} = 1$$

Therefore, the probability to select the next node is P=1, it means that we always select the randomly chose next node (from the successors of the current node).

Local Beam Search

Local Beam Search keeps track of ***k*** states instead of just one!!!

BEGIN: *k* randomly generated states

AT EACH STEP:

- generate all successors of all *k* states
- select the best *k* successors
- ✓ if any selected successor is GOAL, program ends

Example: Problem 3 from Homework # 1. Let us use local beam search.

K=2

Step 1: Initial Node: Seattle

Successors (Seattle) = {San Francisco (2034), Santa Fe (2023), Chicago (2864)

Select the best 2 successors: San Francisco and Santa Fe

Step 2: current nodes: San Francisco and Santa Fe

Successors (Santa Fe) = { Dallas (0), Chicago (3535), Seattle (4596)}

Successors (San Francisco) = {Bakersville (2372), Seattle (3287), Boston (5433)}

Select the best 2 successors: Dallas and Bakersville

Step 3: current nodes: Dallas and Bakersville

Dallas is the Goal!

The path is PATH= {Seattle $\rightarrow$ Santa Fe$\rightarrow$ Dallas}

PATH COST=2103

Genetic Algorithms

<u>Example:</u> *the coloring map problem*

Background: The problem was proposed in the early 1850's. The idea is to use the minimum number of colors to shade different countries, which are of different color to their adjacent neighbors, in the world map.

The problem is considered difficult since no restriction is imposed on the number of regions in the map sharing the same boundary. Many mathematicians, including Augustus De Morgan, Arthur Kempe and Peter Tait had proven that the problem of any simple planar graph can be colored with a minimum of <u>four colors</u>. In 1976 Appel and Haken presented an optimization technique for solving four-coloring problem. However, the sequential method proposed was not very effective and it required many computing hours to solve a large problem. Takefuji and Lee proposed in 1991 a neural network algorithm to solve the four-coloring problem using a 4 x *n* two-dimensional neural network array where *n* is the number of regions to be colored. In 1992 a solution using Genetic Algorithms was proposed by Gwee, Lin and Ho.

We have available the following four colors: {RED, YELLOW, BLUE, GREEN} and we want to color the map of the 48 continental U.S. states.

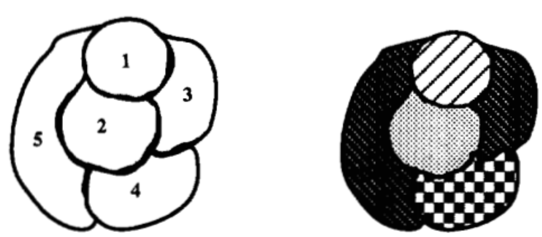
First the map coloring problem can be represented in genetic chromosomal structure:

- Red, yellow, blue and green are represented by R, Y, B and G respectively. Each region can be represented by <u>a gene</u>.
  - A possible solution to the four coloring of the 48-states USA map (excluding Alaska and Hawaii) can be *represented* by <u>**a chromosome**</u> composed of 48 genes with each gene taking on values from R, Y, B or G. A chromosome **S** representing the coloring of an *n*-regions map is then represented as an *n*-tuple as follows:

$$S = < s_1, s_2, \dots, s_n > \text{ for } s_i \in \{R, Y, B, G\}$$

  where s_i is <u>the allele</u> representing the color of region *i* in the map!

  - For example, given the following map in the Figure (a), the chromosome solving the 4-color map problem is <R,Y,B,G,B>, illustrated in the Figure (b).



(a) (b)

In order to find such a chromosome we use a Genetic Algorithm.

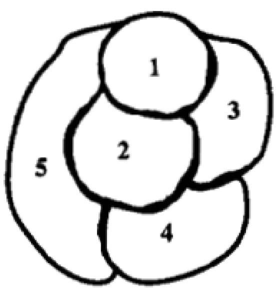
The string length of the chromosome is 5 with each gene denoting a region of the map. The color assignment to each allele, <R,Y,B,G,B> , corresponds to R=RED, Y=YELLOW, B=BLUE and G=GREEN.

- To produce this assignment, we need to define *an objective function g* and its value for each possible chromosome.
- To inform the objective function, we need to consider the adjacency information of each region in a *n*-regions map, which can be expressed by a n x n matrix which is known as the adjacency matrix. In general, an adjacency matrix *M* is given as follows:

$$M = \begin{bmatrix} m_{1,1} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n} \end{bmatrix}$$

where $m_{i,j} = \begin{cases} 1 & if\ region\ i\ and\ j\ are\ adjacent\ on\ the\ map \\ 0 & otherwise \end{cases}$

For the map illustrated in (a), the adjacency matrix is:



i \ j	1	2	3	4	5
1	0	1	1	0	1
2	1	0	1	1	1
3	1	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

Then, given a chromosome *S* (representing the coloring information) and the adjacency matrix *M*, the fitness function value in *S* is:

$fitness(S,M) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} n_{i,j}$ where:

$$n_{i,j} = \begin{cases} m_{i,j} & if\ S_i = Color(region\ i)\ \neq S_j = Color(region\ j) \\ 0 & otherwise \end{cases}$$

```
function Genetic-Algorithm (population, Fitness-Function) returns an individual
  inputs: population: a set of individuals;
          Fitness-Function : a function that measures the fitness of an individual

repeat
   new_population ← empty set
   for i=1 to Size(population) do
         x ← Random-Selection(population , Fitness-Function )
         y ← Random-Selection(population , Fitness-Function )
         child ← Reproduce(x , y )
   if (small random probability) then child ← Mutate (child )
until some individual is fit enough, or enough time has elapsed
return the best individual in population , according to Fitness-Function
```

```
function Reproduce (x , y ) returns an individual
  inputs: x , y : parent individuals

n ← Length(x ); c ← Random number from 1 to n
return Append(SubString(x , 1, c), SubString (y , c+1, n)
```

The search algorithm for a general four-coloring map problem using Genetic Algorithms is given by:

1. ***i := 0; Randomly generate the initial population, Population(0);***

   Example: we randomly generate as *Population(0)* 4 chromosomes: S^1={R,R,Y,B,G};

   $S^2 = \{\mathrm{Y, G, B, R, R}\}$; S^3={G,B,G,Y,B} and S^4={R,G,Y,R,B}

2. ***Evaluate the fitness of every chromosomes in Population(0);***

   Example: we evaluate the $fitness(S^1, M) = \sum_{i=1}^{n}\sum_{j=i+1}^{n} n_{i,j}$, with S^1={R,R,Y,B,G};

i \ j	1	2	3	4	5
1	0	1	1	0	1
2	1	0	1	1	1
3	1	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

$n_{1,2}$=0 because: $S_1^1 = R = S_2^1 = R$, athought $m_{1,2} = 1$

$n_{1,3}$=1 because: $S_1^1 = R \neq S_3^1 = Y$, and $m_{1,3} = 1$

$n_{1,5}$=1 because: $S_1^1 = R \neq S_5^1 = G$, and $m_{1,5} = 1$

$n_{2,3}$=1 because: $S_2^1 = R \neq S_3^1 = Y$, and $m_{2,3} = 1$

$n_{2,4}$=1 because: $S_2^1 = R \neq S_4^1 = B$, and $m_{2,4} = 1$

$n_{2,5}$=1 because: $S_2^1 = R \neq S_5^1 = G$, and $m_{2,5} = 1$

$n_{3,4}$=1 because: $S_3^1 = Y \neq S_4^1 = B$, and $m_{3,4} = 1$

$n_{4,5}$=1 because: $S_4^1 = B \neq S_5^1 = G$, and $m_{4,5} = 1$

Therefore $fitness(S^1, M) = 7$

Similarly, we compute $fitness(S^2, M) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} n_{i,j}$, with S^2={Y,G,B,R,R};

i \ j	1	2	3	4	5
1	0	1	1	0	1
2	1	0	1	1	1
3	1	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

$n_{1,2}$=1 because: $S_1^2 = Y \neq S_2^2 = G$, athought $m_{1,2} = 1$

$n_{1,3}$=1 because: $S_1^2 = Y \neq S_3^2 = B$, and $m_{1,3} = 1$

$n_{1,5}$=1 because: $S_1^2 = Y \neq S_5^2 = R$, and $m_{1,5} = 1$

$n_{2,3}$=1 because: $S_2^2 = Y \neq S_3^2 = B$, and $m_{2,3} = 1$

$n_{2,4}$=1 because: $S_2^2 = G \neq S_4^2 = R$, and $m_{2,4} = 1$

$n_{2,5}$=1 because: $S_2^2 = G \neq S_5^2 = R$, and $m_{2,5} = 1$

$n_{3,4}$=1 because: $S_3^2 = B \neq S_4^2 = R$, and $m_{3,4} = 1$

$n_{4,5}$=0 because: $S_4^2 = R = S_5^2 = R$, and $m_{4,5} = 1$

Therefore $fitness(S^2, M) = 7$

Similarly, we compute $fitness(S^3, M) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} n_{i,j}$, with S^3={G,B,G,Y,B};

i \ j	1	2	3	4	5
1	0	1	1	0	1
2	1	0	1	1	1
3	1	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

$n_{1,2}$=1 because: $S_1^3 = G \neq S_2^3 = B$, athought $m_{1,2} = 1$

$n_{1,3}$=0 because: $S_1^3 = G = S_3^3 = G$, and $m_{1,3} = 1$

$n_{1,5}$=1 because: $S_1^3 = G \neq S_5^3 = Y$; and $m_{1,5} = 1$

$n_{2,3}$=1 because: $S_2^3 = B \neq S_3^3 = R$, and $m_{2,3} = 1$

$n_{2,4}$=1 because: $S_2^3 = B \neq S_4^3 = Y$, and $m_{2,4} = 1$

$n_{2,5}$=0 because: $S_2^3 = B = S_5^3 = B$, and $m_{2,5} = 1$

$n_{3,4}$=1 because: $S_3^3 = G \neq S_4^3 = Y$, and $m_{3,4} = 1$

$n_{4,5}$=1 because: $S_4^3 = Y \neq S_5^3 = B$, and $m_{4,5} = 1$

Therefore $fitness(S^3, M) = 6$

Similarly, we compute $fitness(S^4, M) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} n_{i,j}$, with S^4={R,G,Y,R,B}

i \ j	1	2	3	4	5
1	0	1	1	0	1
2	1	0	1	1	1
3	1	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

$n_{1,2}$=1 because: $S_1^4 = R \neq S_2^4 = G$, athought $m_{1,2} = 1$

$n_{1,3}$=1 because: $S_1^4 = R \neq S_3^4 = Y$, and $m_{1,3} = 1$

$n_{1,5}$=1 because: $S_1^4 = R \neq S_5^4 = B$; and $m_{1,5} = 1$

$n_{2,3}$=1 because: $S_2^4 = G \neq S_3^4 = Y$, and $m_{2,3} = 1$

$n_{2,4}$=1 because: $S_2^4 = G \neq S_4^4 = R$, and $m_{2,4} = 1$

$n_{2,5}$=0 because: $S_2^4 = G = S_5^4 = B$, and $m_{2,5} = 1$

$n_{3,4}$=1 because: $S_3^4 = Y \neq S_4^4 = R$, and $m_{3,4} = 1$

$n_{4,5}$=1 because: $S_4^4 = R \neq S_5^4 = B$, and $m_{4,5} = 1$

Therefore $fitness(S^4, M) = 8$

3. repeat {t:=t+l;

3.1. *Reproduce Population(t) using Population(t-1);*

Example: *We had Population(0)={ S^1, S^2, S^3, S^4} and we know that:*

$fitness(S^1, M) = 7$; $fitness(S^2, M) = 7$; $fitness(S^3, M) = 6$; $fitness(S^4, M) = 8$

The production of the next generation is rated by the fitness function: the probability of being selected for reproduction depends on the fitness value. There are 2 pairs of parents: (S^4, S^1) and (S^4, S^2)

3.2. *Crossover M(t) with probability of P_c;*

S^4={R,G,Y,R,B} → S^1={R,G,Y,B,G}
S^1={R,R,Y,B,G} → S^2={R,R,Y,R,B}

S^4={R,G,Y,R,B} → S^3={R,G,Y,R,R}
S^2={Y,G,B,R,R} → S^4={Y,G,B,R,B}

3.3 *Mutate Population(t) with probability of P_m;*

S^1={R,G,Y,B,Y}
S^2={R,B,Y,R,B}

S^3={R,G,Y,R,R}
S^4={Y,G,Y,R,B}

3.4 ***Evaluate the fitness of Population(t);***

Same as we have done in Step 2

}until optimal solution reached