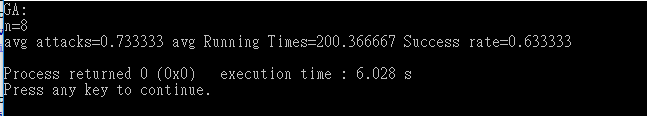
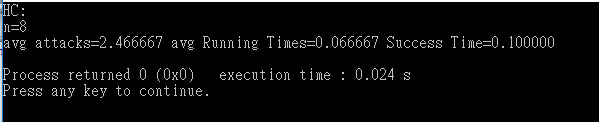
**N=8 RunningTimes=30**

**AvgAttack AvgRnnungTime SR**

**HC: 2.5 0.1 0.1**

**GA: 0.7 200.4 0.6**

****

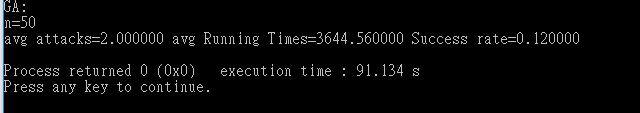
****

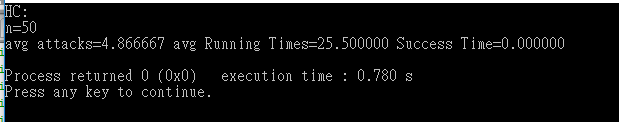
**N=50 RunningTimes=30**

**AvgAttack AvgRnnungTime SR**

**HC: 4.9 25.5 0**

**GA: 2 3644.7 0.12**

****

****

**HC:**

**Condition Number:**

**NSIZE: Size of the chessboard**

**RunTimes: Number of program running times**

### Recursion method

### Neighbor:

### Just one column different from the original state

### GetPoint:

### The more attack the more Point, so less Point is better

### Initialize:

### Randomly make a state while starting

### The solution is the last state

### GA:

**Condition Number:**

**NSIZE: Size of the chessboard**

**RunTimes: Number of program running times**

### SetNum: the population of simple Set

### mutationRate: Base number of divisor

### Runs: How many times of survive loop when running GA one time

### GenerationGap: when there are N children, do survivor function

### Plus: make the fitness state is easily chosen or not

### Sequential method

### Initialize:

### Randomly make a Set of states

### 4-Parts: Selection、Crossover、Mutation、Survivor

### Selection: From PopulationSet choose two states be parents to crossover, with Tournament random

### GetPoint: The more attack the more Point it return, so less return Point is better

### PS: but the point array[ n] = BaseRate - GetPoint( state[n] ), so bigger value in point array is better

### Crossover: With the parents choose above to crossover, randomly choosing one position to cut and randomly from mom or dad to be child’s part of state code

### Mutation: each column randomly mutate when randomNumber is times of mutationRate, so the bigger of the number, the less mutation rate.

### Survivor: include PopulationSet and ChildSet, then sort them(the first state has the biggest point array value)

### Just first SetNum can be survivor

### The solution is the best state which has the highest point of last population

### Discover and conclude:

### HC the faster than GA per round

### If GA Runs is enough the GA avg attack < HC avg attack

**NSIZE and Exe time is proportional**

### Base number

### N=10; SetNum=100; RunTimes=30; mutationRate =100;

### Runs=5000; GenerationGap =2; Plus= 1

### Runs:

### 

### Run and ExeTime is proportional

### Run and (1/avg attack) is proportional

### Run and SR is proportional

### SetNum:

### 

### SetNum and ExeTime is proportional

### SetNum and avg attack do not have obvious relationship in this test

### SetNum and SR do not have obvious relationship in this test

### >>But I Guess if SetNum is big enough can have obvious improve of avg attack and SR

### mutationRate

### 

### mutationRate and ExeTime not have relationship

### mutationRate and avg attack have obvious change in (1,25)🡪(100%~4%)>>have improvement

### mutationRate and SP have obvious change in(1,25)🡪(100%~4%)

### >>have improvement

### GenerationGap

### 

### GenerationGap and ExeTime not have relationship

### >>I Guessed they have relationship, but actually no

### GenerationGap and avg attack not have obvious relationship in this test

### GenerationGap and SR not have obvious relationship in this test

### >>I Guess they have relationship

### Plus

### 

### Plus and ExeTimp is proportional>> effect in Tournament

### Plus and avg attack do not have obvious relationship in this test

### Plus and SR do not have obvious relationship in this test

### >>But I Guess effect