

AI 프로그래밍 HW08 실습 수업

2021.11.18

코딩환경 준비해주세요

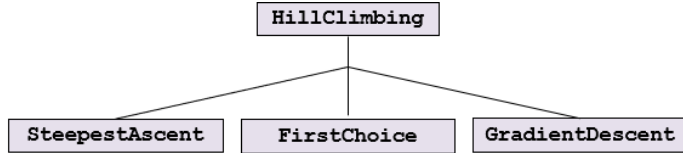

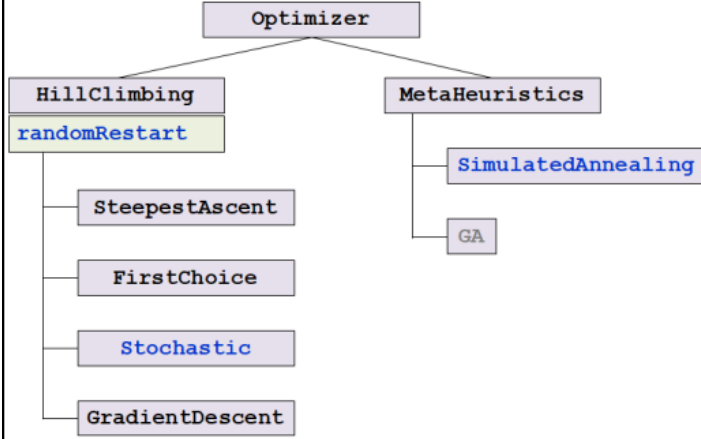
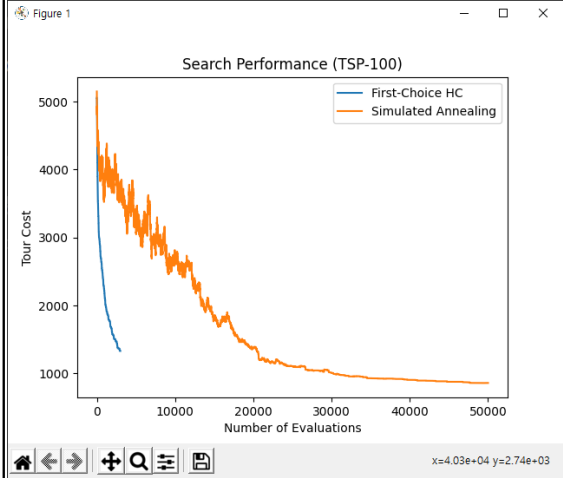
contents

- 1. HW08 과제 목표
- 2. today 실습
- 3. main.py
- 4. plot.py
- 5. 제출사항
- 6. 출석체크

1. HW08 과제 목표

- 1) read setup information file
- new user interface exp.txt, changes Main program
- 2) add algorithm
- changes Class Hierarchy
- add random-restart, stochastic, simulated annealing
- 3) plotting the progress

1. HW08 과제 목표

	1) read setup information file	2) Add algorithm	3) plotting
HW07	<pre>Select the problem type: 1. Numerical Optimization 2. TSP Enter the number: 1 Enter the file name of a function: problem/Griewank.txt Select the search algorithm: 1. Steepest-Ascent 2. First-Choice 3. Gradient Descent Enter the number: 3</pre>	 <pre> graph TD HillClimbing --> SteepestAscent HillClimbing --> FirstChoice HillClimbing --> GradientDescent </pre>	
 HW08	<pre>Enter the file name of experimental setting: exp.txt</pre> <p>exp.txt - Windows 메모장</p> <p>파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)</p> <pre># # Select the problem type: # 1. Numerical Optimization # 2. TSP Enter the number (pType) : 1 # Enter the name of the file : problem/Convex.txt # Enter the name of the file : problem/Griewank.txt # Enter the name of the file : problem/Ackley.txt # Enter the name of the file : problem/tsp30.txt # Enter the name of the file : problem/tsp50.txt # Enter the name of the file : problem/tsp100.txt</pre>	 <pre> graph TD Optimizer --> HillClimbing Optimizer --> MetaHeuristics HillClimbing --> randomRestart HillClimbing --> SteepestAscent HillClimbing --> FirstChoice HillClimbing --> Stochastic HillClimbing --> GradientDescent MetaHeuristics --> SimulatedAnnealing MetaHeuristics --> GA </pre>	 <p>Figure 1: Search Performance (TSP-100)</p> <p>The plot shows Tour Cost (Y-axis, 0 to 5000) versus Number of Evaluations (X-axis, 0 to 50000). Two algorithms are compared: First-Choice HC (blue line) and Simulated Annealing (orange line). First-Choice HC shows a rapid decrease in tour cost, reaching a minimum of approximately 1500 within the first 1000 evaluations. Simulated Annealing shows a more gradual decrease, starting at a high cost of over 5000 and reaching a minimum of approximately 1000 after 50,000 evaluations.</p>

2. today 실습

- read setup information file
- new user interface exp.txt, changes Main program
- main.py
 - 1) createProblem()
 - 2) createOptimizer()
 - 3) conductExperiment()
- plot.py
 - 1) steepest (numeric) 예시
 - 2) first choice, steepest ascent (TSP) 예시

3. main.py – createProblem() 5min

- createProblem() = 문제 타입 읽어오기 (numeric, TSP)
- setVariables(parameter)

```
def createProblem(parameters): ###  
    # Create a problem instance (a class object) 'p' of the type as  
    # specified by 'pType', set the class variables, and return 'p'.  
    pType = parameters['pType']  
    if pType == 1:  
        p = Numeric()  
    p.setVariables(parameters)  
    return p
```

3. main.py – createProblem()

- setVariables(**parameter**)
- class Problem에서 parameter의 파일이름 설정
- class Numeric에서 파일이름 열고 읽어오기

```
class Problem(Setup):
    def __init__(self):
        Setup.__init__(self)
        self._solution = []
        self._value = 0
        self._numEval = 0

    def setVariables(self, parameters):
        # pass
        self._pFileName = parameters['pFileName']
```

```
class Numeric(Problem):
    def __init__(self):
        Problem.__init__(self)
        self._expression = ''
        self._domain = [] # domain as a list

    def setVariables(self, parameters):
        ## Read in a function and its domain from file
        ## Then, set the relevant class variables
        # fileName = input("Enter the file name of the function and domain: ")
        # infile = open(fileName, 'r')
        Problem.setVariables(self, parameters)
        infile = open(self._pFileName, 'r')
        self._expression = infile.readline() # assign expression
        varNames = [] # Variable names
```

3. main.py – createOptimizer() 5min

- createOptimizer() = 서치 알고리즘 읽어오기
- (first-choice, steepest-ascent, gradient-descent...)
- setVariables(parameter)

```
def createOptimizer(parameters): ###
    # Create an optimizer instance (a class object) 'alg' of the type
    # as specified by 'aType', set the class variables, and return 'alg'.
    optimizers = { 1: 'SteepestAscent()' }
    aType = parameters['aType']
    alg = eval(optimizers[aType]) # Create object of target algorithm
    alg.setVariables(parameters)
    return alg
```


3. main.py – createOptimizer()

- setVariables(**parameter**)
- class HillClimbing에서 pType 설정

```
class HillClimbing(Setup):  
    def __init__(self):  
        Setup.__init__(self)  
        self._pType = 0 # Problem type  
        self._limitStuck = 100 # Max evaluations with no improvement  
  
    def setVariables(self, parameters):  
        self._pType = parameters['pType']
```

3. main.py – conductExperiment()

- class HillClimbing에서
- **getAType()** 새롭게 정의하고
- `__init__`에서 초기값 설정

```
class HillClimbing(Setup):
    def __init__(self):
        Setup.__init__(self)
        self._pType = 0          # Problem type
        self.limitStuck = 100    # Max evaluations with n
        self._aType = 0

    def setVariables(self, parameters):
        self._pType = parameters['pType']

    def displaySetting(self):
        print()
        if self._pType == 1:
            print("Mutation step size:", self._delta)

    def run(self):
        pass

    def getAType(self):
        return self._aType
```

3. main.py – conductExperiment() 5min

- conductExperiment()에서
- 필요한대로
- `getSolution()`
- `getValue()`
- `getNumEval()` 등의 함수들
- class Problem에서 정의

```
class Problem(Setup):
    def __init__(self):
        Setup.__init__(self)
        self._solution = []
        self._value = 0
        self._numEval = 0

    def setVariables(self, parameters):
        # pass
        self._pFileName = parameters['pFileName']

    def getSolution(self):
        return self._solution

    def getValue(self):
        return self._value

    def getNumEval(self):
        return self._numEval
```

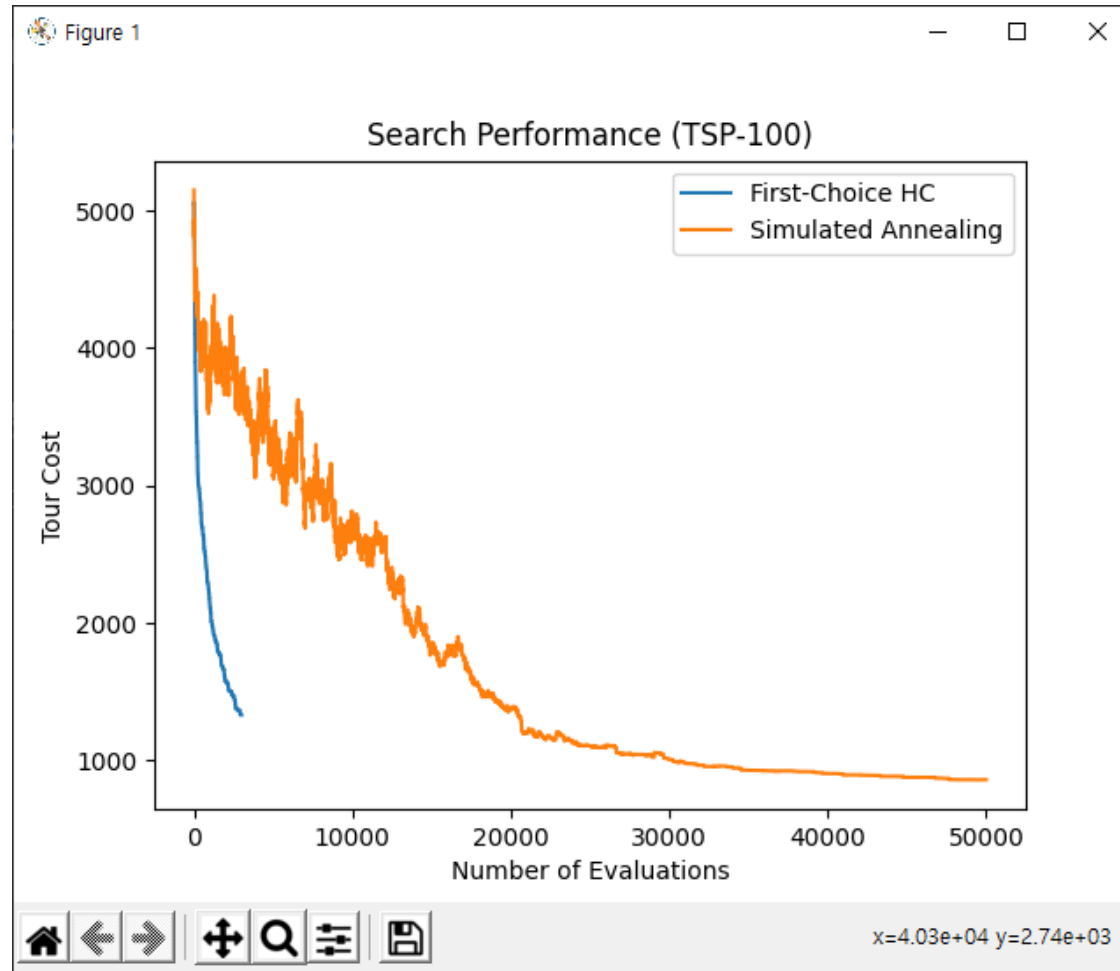
4. plot.py -steepest 예시

- 서칭 과정을
- 그래프로 그리기
- current value를
- 파일에 기록해놓고
- Evaluation 값들을
- Matplotlib을 이용해
- plot

```
class SteepestAscent(HillClimbing):
    def displaySetting(self):
        print()
        print("Search Algorithm: Steepest-Ascent Hill Climbing")
        HillClimbing.displaySetting(self)

    def run(self, p):
        current = p.randomInit() # A current candidate solution
        valueC = p.evaluate(current)
        f=open('steepest.txt', 'w')
        while True:
            neighbors = p.mutants(current)
            successor, valueS = self.bestOf(neighbors, p)
            f.write(str(round(valueC,1))+'\n')
            if valueS >= valueC:
                break
            else:
                current = successor
                valueC = valueS
        f.close()
        p.storeResult(current, valueC)
```

4. plot.py (first-choice, simulated annealing – TSP100 예시)



5. 제출사항

- 소스코드 5개 (main.py, optimizer.py, problem.py, setup.py, plot.py)
- txt 파일 2개 (fisrt.txt, anneal.txt)
- <보고서에 첨부할 스크린샷>
- 새로 추가한 알고리즘 (stochastic, simulated annealing) 결과
터미널 캡처 12개 (각각 numeric 3개, TSP 3개 동작 결과)
- Plot.py로 만든 그래프 2개 (exp.txt의 세팅이
limitStuck=100, limitEval=50000 / limitStuck=1000, limitEval=10000 일때)
- 각각이 의미하는바 설명 (결론 및 토의 내용)