AI 프로그래밍
HW09 실습

2021.11.25

코딩 환경 준비하세요

contents

- 1. HW09 전체 목표
- 2. today 실습 내용
- 3. 제출사항
- 4. 출석체크

1. HW09 전체 목표

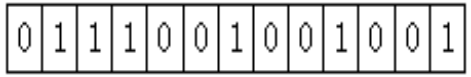
- 1) Genetic Algorithm 구현
- 2) 6개 문제 X 6개 알고리즘 성능비교, discussion

2. Today 실습 내용

Genetic Algorithms 단계	optimizer.py	problem.py	
	class GA(metaheuristics):	class Numeric(Problem):	class Tsp(Problem):
1) Chromosome design 2) initialization	def run(self, p)	def initializePop(self, size) def randBinStr(self)	def initializePop(self, size)
3) Fitness evaluation	def evalAndFindBest(self, pop, p)	def evalInd(self, ind) def decode(self, chromosome) def binaryToDecimal(self, binCode, l, u)	def evalInd(self, ind)
4) Selection	def selectParents(self, pop) def selectTwo(self, pop) def binaryTournament(self, ind1, ind2)		
5) Crossover		def crossover(self, ind1, ind2, uXp) def uXover(self, chrInd1, chrInd2, uXp)	def crossover(self, ind1, ind2, XR) def oXover(self, chrInd1, chrInd2)
6) Mutation		def mutation(self, ind, mrF)	def mutation(self, ind, mR)
		def indToSol(self, ind)	def indToSol(self, ind)

1) Chromosome design 5min
2) Initialization

1) Chromosome design

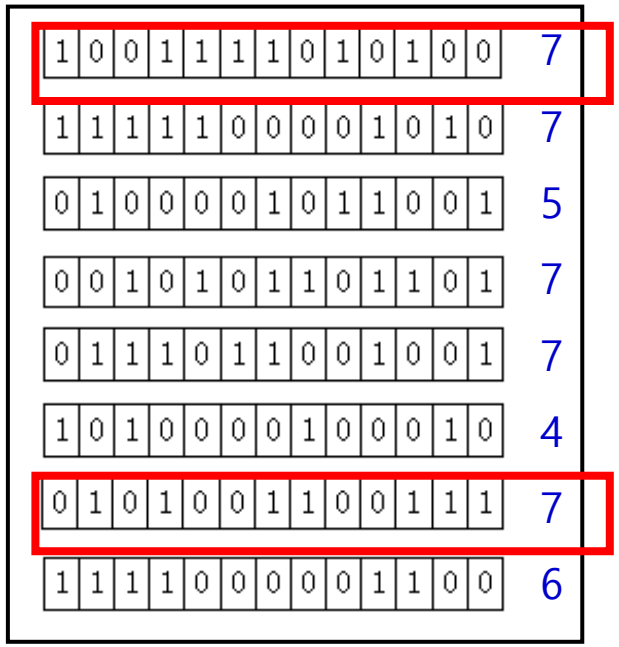



```python
# Make a population of given size
# chromosome = 염색체
def initializePop(self, size):
    pop = []
    for i in range(size):
        chromosome = self.randBinStr()
        pop.append([0, chromosome])
        # 앞에 0은 fitness(=evaluation) 값
    return pop
```

```python
def randBinStr(self):
    k = len(self._domain[0]) * self._resolution
    chromosome = []
    for i in range(k):
        allele = random.randint(0, 1)
        chromosome.append(allele)
    return chromosome
# allele = 대립유전자
```

3) Fitness evaluation

10min



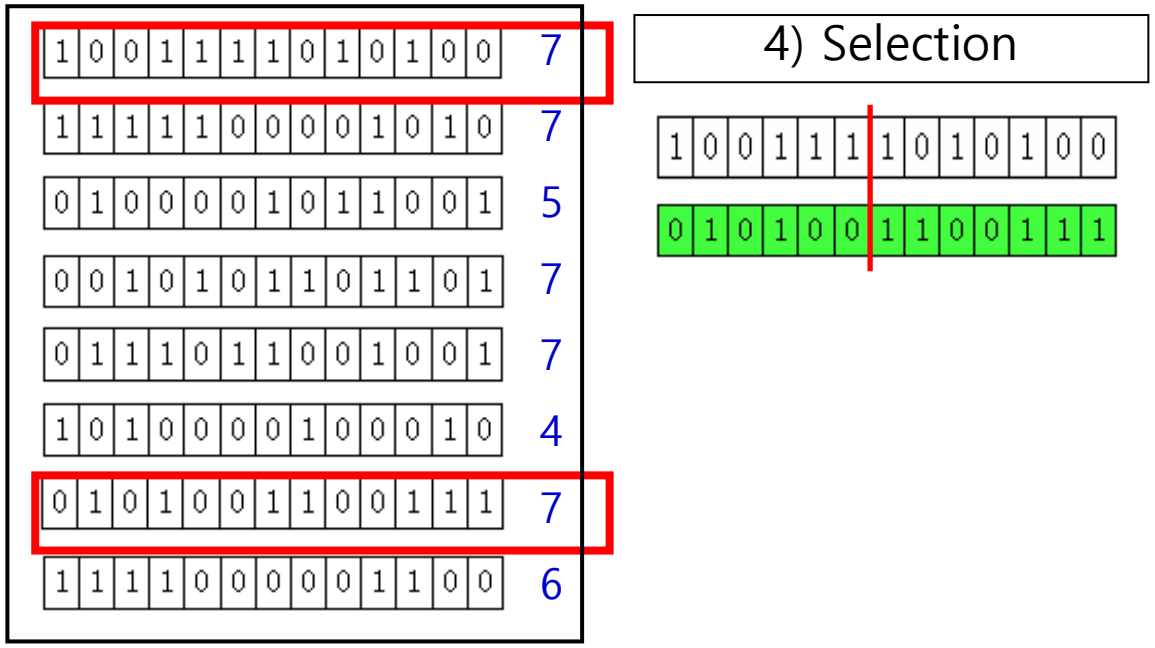
3) Fitness evalution

```python
def evalInd(self, ind):  # ind: [fitness, chromosome]
    ind[0] = self.evaluate(self.decode(ind[1]))
    # Record fitness
```

```python
def decode(self, chromosome):
    r = self._resolution
    low = self._domain[1]  # list of lower bounds
    up = self._domain[2]   # list of upper bounds
    genotype = chromosome[:]
    phenotype = []
    start = 0
    end = r   # The following loop repeats for # variables
    for var in range(len(self._domain[0])):
        value = self.binaryToDecimal(genotype[start:end],
                                     low[var], up[var])
        phenotype.append(value)
        start += r
        end += r
    return phenotype
    # genotype = 유전자형, phenotype = 표현형
```

```python
def binaryToDecimal(self, binCode, l, u):
    r = len(binCode)
    decimalValue = 0
    for i in range(r):
        decimalValue += binCode[i] * (2 ** (r - 1 - i))
    return l + (u - l) * decimalValue / 2 ** r
```

4) Selection 10min

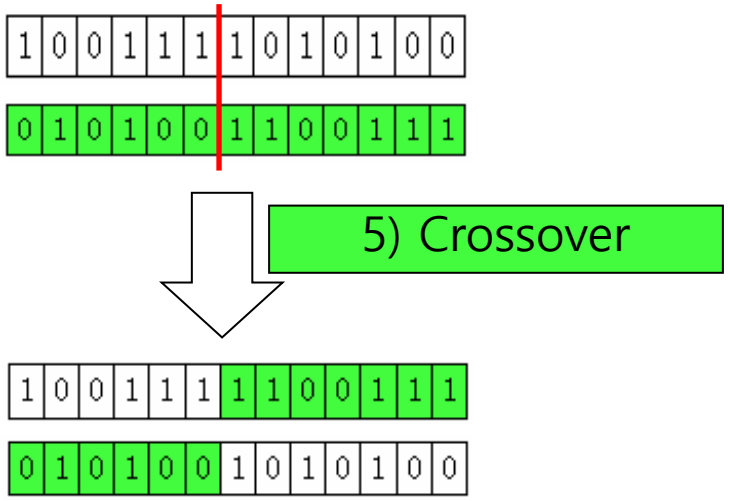



```python
def selectParents(self, pop):
    ind1, ind2 = self.selectTwo(pop)
    par1 = self.binaryTournament(ind1, ind2)
    ind1, ind2 = self.selectTwo(pop)
    par2 = self.binaryTournament(ind1, ind2)
    return par1, par2

def selectTwo(self, pop):
    popCopy = pop[:]
    random.shuffle(popCopy)
    return popCopy[0], popCopy[1]

def binaryTournament(self, ind1, ind2):
    if ind1[0] < ind2[0]:
        return ind1
    else:
        return ind2
```

5) Crossover 5min




```python
def crossover(self, ind1, ind2, uXp):
    # pC is interpreted as uXp# (probability of swap)
    chr1, chr2 = self.uXover(ind1[1], ind2[1], uXp)
    return [0, chr1], [0, chr2]

def uXover(self, chrInd1, chrInd2, uXp): # uniform crossover
    chr1 = chrInd1[:]  # Make copies
    chr2 = chrInd2[:]
    for i in range(len(chr1)):
        if random.uniform(0, 1) < uXp:
            chr1[i], chr2[i] = chr2[i], chr1[i]
    return chr1, chr2
```

3. 제출사항

- 보고서에 6개 문제 X 6개 알고리즘 성능비교
- Average objective value
- Best objective value found
- Average number of evaluations
- (annealing, GA) Average iteration
- discussion

Results of Numerical Optimization

	Convex		Griewank		Ackley	
Steepest Ascent	0.0 0.0	774,692	0.260 0.108	67,144	17.832 14.029	12,182
First Choice	0.0 0.0	274,521	0.254 0.064	38,824	18.214 14.108	14,377
Stochastic	0.0 0.0	2,088,171	0.218 0.096	387,008	18.879 16.729	141,156
Gradient Descent	**0.0** **0.0**	199,935	0.216 0.118	856,635	17.447 8.101	5,234
Simulated Annealing	0.0 0.0	500,000 63,565	0.367 0.145	500,000 18,252	19.319 18.940	500,000 5,541
GA	3.920 0.766	500,000 227,140	**0.036** **0.015**	500,000 220,390	**0.214** **0.141**	500,000 173,900