HSEA-第四次作业-实验报告

191250026 丁云翔

任务一

对POSS算法的源码分析如下。

数据集

```
[y,X] = libsvmread('sonar_scale');
```

优化目标

```
pos=find(offspring==1);
coef=lscov(X(:,pos),y);
err=X(:,pos)*coef-y;
offspringFit(1)=err'*err/m;
```

计算代价

```
k=8;
T=round(n*k*k*2*exp(1));
for i=1:T
    ...
```

解的去除

```
temp=find(fitness(:,2)<=k);
j=max(fitness(temp,2));
seq=find(fitness(:,2)==j);
selectedVariables=population(seq,:);
```

任务二

对NSGA-II和MOEA/D算法的源码分析如下。

NSGA-II

种群大小和最大迭代轮数

```
pop_size = 20
max_gen = 921
```

种群初始化和迭代

```
#Initialization
min_x=-55
max_x=55
solution=[min_x+(max_x-min_x)*random.random() for i in range(0,pop_size)]
gen_no=0
while(gen_no<max_gen):
    ...
```

突变和交叉

```
#Function to carry out the crossover
def crossover(a,b):
    r=random.random()
    if r>0.5:
        return mutation((a+b)/2)
    else:
        return mutation((a-b)/2)

#Function to carry out the mutation operator
def mutation(solution):
    mutation_prob = random.random()
    if mutation_prob <1:
        solution = min_x+(max_x-min_x)*random.random()
    return solution
```

MOEA/D

种群大小和种群初始化

```
self.populationSize_ = int(0)

# Stores the population
self.population = []

if population:
    self.population = population
    fitnesses = self.toolbox.map(self.toolbox.evaluate, self.population)
    for ind, fit in zip(self.population, fitnesses):
        ind.fitness.values = fit

    self.populationSize_ = mu
```

最大迭代轮数

```
self.maxEvaluations = -1
if maxEvaluations == ngen == 0:
    print
    "maxEvaluations or ngen must be greater than 0."
    raise ValueError
if ngen > 0:
    self.maxEvaluations = ngen * self.populationSize_
else:
    self.maxEvaluations = maxEvaluations
```

迭代

```
while self.evaluations_ < self.maxEvaluations:
    ...
```

突变和交叉

```
child = None
children = []
parents = [None] * 3

candidates = list(self.population[:])
parents[0] = deepcopy(candidates[p[0]])
parents[1] = deepcopy(candidates[p[1]])

children = self.toolbox.mate(parents[0], parents[1])

# Apply mutation
children = [self.toolbox.mutate(child) for child in children]
```

任务三

基于分解策略进行改进

本思路基于参考文献4。

由POSS算法的说明文档所述，POSS算法的计算复杂度为2enk^2，仍存在可以加速的空间。

可以在POSS算法的基础上，通过限定候选集P的大小，多次运行POSS算法去逐段找到中间解，而不再是只运行一次POSS算法直接去找到所有中间解。

根据参考文献4的实验证明，改进后的算法在获得与POSS算法相同近似性能下界（但实际性能其实略有下降）的同时，运行时间随着分解个数的增加超线性下降。

参考文献

1. 《Subset Selection by Pareto Optimization》，Chao Qian，Yang Yu，Zhi-Hua Zhou
2. https://github.com/haris989/NSGA-II
3. https://github.com/mbelmadani/moead-py
4. 《基于分解策略的多目标演化子集选择算法》，钱超，周志华