ELE083 Computação Evolucionária

Trabalho Prático - Makespan

Davi Pinheiro Viana - 2013029912
Rafael Carneiro de Castro - 2013030210

Minas Gerais, Brasil

Keywords: Makespan, Computação Evolucionária, Evolução Diferencial

1. Introdução

O trabalho final da disciplina de Computação Evolucionária consiste no emprego de uma das técnicas estudadas em sala de aula no decorrer do semestre para a solução de um problema de Engenharia. Tal problema é *Job Shop Scheduling Problem* (JSSP).

O *Job Shop Scheduling Problem* é um problema clássico de otimização combinatória e possui diversas aplicações nas indústrias e empresas. Seu objetivo é obter uma sequência de tarefas a serem executadas de forma a maximizar a utilização dos recursos disponíveis. Por recursos, entende-se como máquinas, pessoas, ou ambos.

A nível de contexto, imagina-se uma linha de produção com X etapas, que devem ser executadas em ordem. Tem-se como entrada um arquivo texto contendo Y linhas, que representam Y pedidos, com X números inteiros em cada linha. Estes números representam o tempo que o pedido daquela linha vai gastar em cada etapa. O objetivo é retornar uma sequência de produção dos pedidos em tempo ótimo, de forma que o conjunto de pedidos será entrega o mais rápido possível.

Para este trabalho, considera-se que a linha de produção tem três etapas. Sendo assim, os arquivos de entrada contêm três colunas.

2. Desenvolvimento

Nesta seção serão apresentadas as decisões tomadas para a implementação de um algoritmo que soluciona o problema descrito na *Introdução*.

2.1. Representação de Indivíduos

Para a decisão da forma de se representar indivíduos, primeiro é importante levar em consideração a entrada do algoritmo. Conforme especificações, a entrada é dada por arquivos textos no formato que pode ser visto na Figura 1. Conforme descrito na *Introdução*, cada linha representa um pedido, e cada coluna representa o tempo gasto em uma etapa de produção (máquina). Os arquivos *entrada_3.txt*, *entrada_10.txt* e *entrada_25.txt* foram disponibilizados junto à especificação deste trabalho.

entrada_25.txt

```
19,11,13
9,27,20
14,12,13
7,19,7
28,25,28
21,30,30
7,10,10
17,30,24
16,24,6
30,20,15
19,28,20
12,6,26
9,23,26
22,16,7
19,14,26
24,22,21
10,30,9
10,21,13
29,22,24
27,29,6
26,14,11
28,30,11
17,14,29
9,30,13
26,15,12
```

entrada_10.txt

```
11,22,29
26,30,13
19,16,17
9,29,9
20,25,28
8,6,30
30,28,26
30,17,26
23,26,12
6,10,19
```

entrada_3.txt

```
10,14,13
15,13,15
10,30,6
```

Figura 1: Exemplos de arquivos de entrada

Em virtude de este ser um problema combinatório cuja saída é uma sequência de produção dos pedidos, os indivíduos podem ser representados por uma sequência de Y números inteiros que vão de 1 até Y, onde Y é a quantidade de pedidos na entrada. Esta sequência é a que otimiza a produção dos pedidos em questão. Sendo assim, as soluções candidatas do problema são representadas por vetores de números inteiros.

2.2. Algoritmo Evolucionários Escolhido

Uma das técnicas e algoritmos estudados em sala de aula foi a chamada *Evolução Diferencial*. Esta é uma técnica amplamente utilizada na solução de problemas através da Computação Evolucionária, possuindo alto desempenho e implementação simples. Sendo assim, esta foi a técnica usada como base para a implementação de uma algoritmo que soluciona o problema em questão neste trabalho.

Conforme visto em sala, um pseudo-código para a *Evolução Diferencial* pode ser observado na Figura 2. No código da figura, é importante se destacar a equação:

$$u_{t,i,j} = x_{t,r1,j} + F(x_{t,r2,j} - x_{t,r3,j})$$

onde $u_{t,i,j}$ representa um indivíduo que sofreu mutação baseado na manipulação vetorial de outros três indivíduos selecionados aleatoriamente.

Enquanto algum critério de parada não for satisfeito **faça**
Para $i = 1$ até N **faça**
Selecione aleatoriamente $r1, r2, r3 \in \{1, \ldots, N\}$
Selecione aleatoriamente $\delta_i \in \{1, \ldots, n\}$
Para $j = 1$ até n **faça**
Se $\mu_{[0,1]} \leq C \vee j == \delta_i$ **então**
$u_{t,i,j} = x_{t,r1,j} + F(x_{t,r2,j} - x_{t,r3,j})$
Senão
$u_{t,i,j} = x_{t,i,j}$
Fim se
Fim para
Se $f(u_{t,i}) \leq f(x_{t,i})$ **então**
$x_{t+1,i} \leftarrow u_{t,i}$
Senão
$x_{t+1,i} \leftarrow x_{t,i}$
Fim se
Fim para
$t \leftarrow t + 1$
Fim enquanto

Figura 2: Pseudo-código para a evolução diferencial

2.3. Operador de Seleção

O operador de seleção do algoritmo de evolução diferencial implementado pode ser visto na condicional **Se** $f(u_{t,i}) <= f(x_{t,i})$ **então**. Nesta condicional, entende-se por $f(x)$ como a avaliação do *fitness* de um indivíduo. Assim, um indivíduo mutante é selecionado se seu *fitness* é menor que o *fitness* de seu correspondente na população original.