Aplicação do Algoritmo Genético ao Parallel Machines Scheduling Problem

Diego Dimer, Eduardo Paim, Gabriel Tadiello

1 Problema

1.1 O que é

O problema do escalonamento em máquinas paralelas com tempo de setup consiste em escalonar N tarefas (1, ..., n) uma única vez em alguma das M máquinas (1, ..., m). A solução consiste da ordem de atribuição das tarefas em cada máquina, com o objetivo de minimizar o *makespan*: tempo para completar todas as tarefas paralelamente. São dados de entrada do problema o número de máquinas e tarefas, além do tempo de processamento P_{jk} da tarefa j na máquina k e o tempo de setup S_{ijk}, representando o tempo necessário de setup para realizar tarefa j após a tarefa i na máquina k. Com os dados $S_{ijk} e P_{jk}$ montamos a matriz G_{ijk}, onde $G_{ijk} = S_{ijk} + P_{jk}$, necessária na formulação do problema.

1.2 Programação Linear Inteira Mista

Antes de implementar a meta-heurística desenvolvida nesse trabalho foi feita a formulação matemática em Programação Linear Inteira Mista, com função objetivo de minimizar o tempo máximo entre todas as máquinas. A formulação, no entanto, só é eficaz para pequenas instâncias do problema, já que PILM tende a consumir muitos recursos computacionais conforme a instância cresce. O problema foi formulado da seguinte maneira:
$j = 1, ..., n$ tarefas
$i = 0, ..., n$ tarefas mais tarefa *"dummy"* em i_0
$k = 1, ..., m$ máquinas

$$\begin{aligned}
\textbf{min} \quad & C_{max} && && (1)\\
\textbf{sujeito a} \quad & \sum_{k=1}^{m}\sum_{\substack{i=0\\ i\neq j}}^{n} x_{ijk} = 1 && \forall j && (2)\\
& \sum_{\substack{i=0\\ i\neq h}}^{n} x_{ihk} - \sum_{\substack{j=0\\ j\neq h}}^{n} x_{hjk} = 0 && \forall h,k && (3)\\
& c_i + \sum_{k=1}^{m} x_{ijk}G_{ijk} + V((\sum_{k=1}^{m} x_{ijk}) - 1) \leq c_j && \forall i,j && (4)\\
& c_j \leq c_{max} && \forall j && (5)\\
& \sum_{j=1}^{n} x_{0jk} = 1 && \forall k && (6)\\
& c_j \geq 0 && \forall j && (7)\\
& c_0 = 0 && && (8)\\
& x_{ijk} \geq 0 && \forall i,j,k && (9)
\end{aligned}$$

onde:
c_{max}: tempo total de completar o escalonamento da máquina mais demorada *(makespan)*
c_j: tempo para completar o job j
G_{ijk}: tempo de processamento e setup para executar o job j após o job i na máquina k
x_{ijk}: 1 se job j é escalonado diretamente após job i na máquina k, 0 caso contrário
x_{0jk}: 1 se job j é o primeiro job escalonado na máquina k, 0 caso contrário
x_{j0k}: 1 se job j é o último job escalonado na máquina k, 0 caso contrário
n: número de jobs
m: número de máquinas
V: número inteiro muito grande

Notas: A função objetivo em (1) é minimizar o tempo máximo que as máquinas demorariam para processar todas as tarefas. A restrição em (2) serve para garantir que cada tarefa é executada apenas uma vez e apenas em uma máquina. A restrição em (3) assegura que cada tarefa tenha uma tarefa antecedendo e precedendo ela se ela for executada na máquina k. A restrição (4) é usada para acumular na variável c_i o tempo de execução daquela máquina (explica-se, por essa restrição, a necessidade de uma tarefa *"dummy"* com tempo de completar em 0 (8)). A restrição (5) garante que c_{max} é sempre o maior entre os c_j. Em (6) garantimos que cada máquina começa com uma tarefa e apenas uma (não há necessidade pela restrição em (4) de se fazer o mesmo para última tarefa. Em (7) e (9) garanta-se que as variáveis não vão assumir valores negativos (além da restrição omitida de que elas sejam inteiras).

1.3 Exemplo

2 Meta-Heurística

2.1 Algoritmo Genético

2.2 Escolha dos parâmetros

2.2.1 Taxa de mutação

2.3 Representação da Solução