Trabalho Prático 2 - ACO

Renato Sérgio Lopes Júnior - 2020667570

1 Introdução

Neste trabalho foi implementado um algoritmo de colônia de formigas (ACO) para resolver o problema do *Job-Shop Scheduling* (JSSP), que consiste em encontrar a melhor alocação de tarefas em diferentes máquinas, de maneira a reduzir o tempo gasto para completar os *jobs*.

Nas próximas seções, serão apresentados detalhes sobre a modelagem, implementação e os experimentos realizados.

2 Modelagem

Para modelar o problema JSSP como um ACO, precisamos definir como cada solução será representada, a função de *desirability* e como os feromônios serão representados e atualizados.

A solução será dada pela ordem na qual as tarefas dos *jobs* foram executadas, que irá determinar o valor de *makespan*, o tempo gasto para completar todas as operações de todos os *jobs*. Assim, a cada instante de tempo, a formiga deve decidir quais operações executar dentre as disponíveis considerando as restrições do problema (ordem das operações e disponibilidade das máquinas). Com base nessa observação, a solução encontrada por cada formiga será caracterizada pelas operações dos jobs que foram iniciadas a cada instante.

A função de *desirability* irá, em conjunto aos feromônios, orientar a escolha das operações pelas formigas. Ela é definida como o inverso do tempo de processamento gasto por cada iteração. Assim, ela é verificada localmente e, quanto menor for esse tempo de processamento, maior será a *desirability* de iniciar a operação.

Os feromônios irão guiar as formigas em direção ao caminho da melhor solução. Logo, há um valor de feromônio π_{ij} associado a cada *job* i e intervalo de tempo j, que indica quantas formigas iniciaram o *job* i no instante j. Assim, a matriz de feromônios terá dimensão (num_jobs x max_time), onde max_time é o tempo de máximo de excução (que é o makespan do pior caso, onde todos os *jobs* são executados sequencialmente, dado pela soma de todos os custos de todas as operações).

A atualização dos feromônios é feita usando o método *Max-Min*. Assim, os feromônios são limitados ao intervalo $[\tau_{\min}, \tau_{\max}]$, onde $\tau_{\min}$ e $\tau_{\max}$ são parâmetros do programa. Todos os feromônios são inicializados com $\tau_{\max}$ e são atualizados a cada iteração de acordo com a taxa de evaporação e com o caminho utilizado pela formiga que alcançou o melhor resultado, considerando todas as iterações. Todos os feromônios deste caminho são acrescidos de $(1/q) * \text{num\_ants}$, onde q é o *makespan* encontrado pela melhor formiga e num_ants é a quantidade de formigas.

3 Implementação

O algoritmo foi implementado em Python e nesta seção serão apresentados os detalhes dos componentes do programa desenvolvido.

A função `aco`, do arquivo `aco.py`, implementa a lógica principal do ACO. Assim, há um loop principal que controla as iterações do algoritmo. Para cada iteração, os caminhos das formigas são atualizados de acordo com as escolhas de quais operações são executadas em cada instante de tempo. Cada formiga guarda dois vetores de estado: um com o estado de cada máquina, que indica se uma máquina está livre ou quanto tempo falta para ela ficar livre, e um outro com o estado de cada *job*, que indica qual foi a última operação iniciada de cada *job*. Com base nesses vetores de estado, a formiga irá decidir quais *jobs* podem ser iniciados em um dado instante de tempo, com base na disponibilidade das máquinas e no status de execução de cada *job*.

Após decidir quais jobs estão disponíveis, a formiga irá escolher probabilisticamente qual *job* executar, com base nos valores de *desirability* e do feromônio relacionados a cada operação e ao instante de tempo. Assim que todos os *jobs* estiverem concluídos, a formiga termina sua execução e o *makespan* e o caminho encontrados são salvos. Ao fim de uma iteração do ACO, os feromônios são atualizados, descontando a taxa de evaporação e incrementando o valor dos feromônios referentes ao caminho da melhor formiga encontrada, sendo que se considera a melhor formiga encontrada entre todas as iterações executadas.

O arquivo `data.py` contém as funções que lidam com a entrada e processamento dos dados das instâncias. Em `visualization.py`, são criadas visualizações dos resultados obtidos pelo algoritmo.

O programa possui os seguintes parâmetros, que podem ser passados durante a execução de `main.py`:

- `instances_filepath`: caminho para o arquivo contendo as instâncias
- `instance`: qual instância do problema executar
- `output_dir`: diretório onde serão salvos os resultados e visualizações
- `trials`: número de execuções independentes do algoritmo
- `ants`: quantidade de formigas
- `iterations`: quantidade de iterações do ACO
- `pheromones_min`: o valor mínimo de cada feromônio (equivalente a $\tau_{\min}$)
- `pheromones_max`: o valor máximo de cada feromônio (equivalente a $\tau_{\max}$)
- `alpha`: peso associado ao feromônio ao calcular as probabilidades
- `beta`: peso associado a *desirability* ao calcular as probabilidades
- `evaporation_rate`: taxa de evaporação usada durante a atualização dos feromônios

Para executar o programa com os valores padrão dos parâmetros, use o seguinte comando em um terminal: `python main.py`. O arquivo com as instâncias `jobshop1.txt` deve estar no mesmo diretório que o script *main.py*.

4 Metodologia

Foram realizados experimentos variando os parâmetros usando as instâncias `la01` e `la29`, visto que elas são um meio termo entre as demais em relação à quantidade de *jobs* e máquinas, fazendo com que os parâmetros que funcionem bem para elas, provavelmente, também funcionem bem para as demais. Em todos os experimentos, foram realizadas 30 execuções independentes.

Inicialmente, os parâmetros foram definidos conforme a Tabela 1, e o algoritmo foi executado para diferentes valores do número de formigas: 50, 150 e 250.

Tabela 1: Valores iniciais dos parâmetros

Parâmetro	Valor
iterations	100
pheromones_max	100.0
pheromones_min	10.0
evaporation_rate	0.2
alpha	1.0
beta	2.0

Em seguida foram realizados experimentos variando os parâmetros pheromones_max e pheromones_min usando duas configurações: pheromones_max = 100, pheromones_min = 10 e pheromones_max = 100, pheromones_min = 0.

Por fim, foram realizados experimentos para três valores diferentes do parâmetro evaporation_rate: 0.2, 0.5 e 0.9.

5 Experimentos

Na Tabela 2 são apresentados os valores de *makespan* da melhor solução encontrada e da diferença para o ótimo para cada um dos valores de número de formigas testado na instância la01. Da mesma forma, os resultados variando o número de formigas na instância la29 são apresentados na Tabela 3.

Tabela 2: Resultados obtidos na instância la01 variando o número de formigas

Número de formigas	Melhor Solução Encontrada	Diferença para o ótimo
50	766.00 ± 0.00	100.00 ± 0.00
150	772.00 ± 0.00	106.00 ± 0.00
250	750.00 ± 0.00	84.00 ± 0.00

Após a análise dos resultados obtidos no experimento anterior, o número de formigas foi definido como 500 e o número de iterações como 10, uma vez que

Tabela 3: Resultados obtidos na instância la29 variando o número de formigas

Número de formigas	Melhor Solução Encontrada	Diferença para o ótimo
50	1521.00 ± 0.00	364.00 ± 0.00
150	1509.00 ± 0.00	352.00 ± 0.00
250	1494.00 ± 0.00	337.00 ± 0.00

um número maior de formigas obteve um melhor resultado e o melhor resultado encontrado não alterava muito após as 10 primeiras iterações.

Os resultados obtidos variando os limites para o feromônio são apresentados nas Tabelas 4 e 5.

Tabela 4: Resultados obtidos na instância la01 variando pheromones_min e pheromones_max

Pheromones range	Melhor Solução Encontrada	Diferença para o ótimo
min=0, max=100	750.00 ± 0.00	84.00 ± 0.00
min=10, max=100	774.00 ± 0.00	108.00 ± 0.00

Tabela 5: Resultados obtidos na instância la29 variando pheromones_min e pheromones_max

Pheromones range	Melhor Solução Encontrada	Diferença para o ótimo
min=0, max=100	1513.00 ± 0.00	356.00 ± 0.00
min=10, max=100	1555.00 ± 0.00	398.00 ± 0.00

Com base nesses resultados, os valores foram definidos como pheromones_min=0 e pheromones_max=100.

Os resultados obtidos nos experimentos variando a taxa de evaporação dos feromônios são apresentados nas Tabelas 6 e 7.

Com base nos resultados obtidos, a taxa de evaporação foi definida como 0.2.

Os resultados obtidos nas outras instâncias, ft06 e la40, são apresentados na Tabela 8.

6 Resultados

Analisando as Tabelas 2 e 3, podemos verificar que, como esperado, uma maior quantidade de formigas de fato leva a melhores resultados, uma vez que mais caminhos serão explorados, ao custo de um maior tempo de processamento. Esse experimento também revelou que os valores encontrados não mudam muito ao longo das iterações, o que justifica a redução do número de iterações no restante dos experimentos executados.

A respeito da mudança dos valores de máximo e mínimo do feromônio, as Tabelas 4 e 5 mostram que um valor mínimo menor, que favorece o *exploitation* uma vez que o caminho mais utilizado terá probabilidade bem maior e os caminhos menos populares terão probabilidade baixa, leva a um melhor resultado.

Tabela 6: Resultados obtidos na instância la01 variando a taxa de evaporação de feromônio

Taxa de evaporação	Melhor Solução Encontrada	Diferença para o ótimo
0.2	750.00 ± 0.00	84.00 ± 0.00
0.5	754.00 ± 0.00	88.00 ± 0.00
0.9	759.00 ± 0.00	93.00 ± 0.00

Tabela 7: Resultados obtidos na instância la29 variando a taxa de evaporação de feromônio

Taxa de evaporação	Melhor Solução Encontrada	Diferença para o ótimo
0.2	1513.00 ± 0.00	356.00 ± 0.00
0.5	1501.00 ± 0.00	344.00 ± 0.00
0.9	1519.00 ± 0.00	362.00 ± 0.00

Esse dilema entre *exploitation* e *exploration* também pode ser verificado nos experimentos que variam a taxa de evaporação, nas Tabelas 6 e 7. Note que, quando a taxa de evaporação fica muito alta (0.9), o *exploration* é favorecido, uma vez que os feromônios dos melhores caminhos serão evaporados mais rapidamente. Os resultados mostram que uma taxa de evaporação menor (0.2) obtém melhores resultados.

Analisando a Tabela 8, podemos verificar que os melhores parâmetros encontrados para as instâncias la01 e la29 funcionam de certa forma bem também para as instâncias ft06 e la40.

7 Conclusão

Neste trabalho foi feita a implementação de um algoritmo de colônia de formigas (ACO) para resolver o problema do *Job-Shop Scheduling* (JSSP). Durante esse desenvolvimento foi possível entender mais sobre o funcionamento dos ACO e a execução dos experimentos possibilitou um melhor entendimento sobre a influência de cada parâmetro do algoritmo no resultado encontrado.

Tabela 8: Resultados obtidos na nas instâncias ft06 e la40

Instância	Melhor Solução Encontrada	Diferença para o ótimo
ft06	60.00 ± 0.00	5.00 ± 0.00
la40	1467.00 ± 0.00	245.00 ± 0.00