**Inteligência Artificial**

**Problema da Mínima Latência**



Bianca Faria Dutra Fragoso

Pedro Felipe Magalhães

**Introdução**

Esse trabalho implementa alguns algoritmos que chegam perto da melhor solução para o problema da mínima latência.

O MLP é uma variante do problema do caixeiro viajante, sendo definido a seguir. Seja G=(V, A) um grafo direcionado completo, onde V = {v0, …, vn} é o conjunto de vértices, v0 representa o depósito e os outros vértices, os clientes, e A={(i,j) : i, j ∈ V, i ≠ j} é o conjunto de arcos, sendo que cada um é associado com o tempo de viagem entre os vértices i e j. O objetivo do MLP é encontrar um circuito hamiltoniano em G que minimiza o tempo total de espera (latência) dos clientes. O tempo de latência do i-éssimo cliente, ou l(i), é a soma de todos os tempos de viagem do depósito até o i-éssimo cliente presente no circuito Hamiltoniano.

1. **Busca Gulosa com profundidade limitada e poda**
   1. O algoritmo
   2. Os resultados

|  |  |  |
| --- | --- | --- |
|  | Melhor solução já encontrada | Nossa melhor solução |
| *dantzig42* |  |  |
| *gr48* |  |  |
| *brazil58* |  |  |
| *gr120* |  |  |

* 1. O tempo

|  |  |  |
| --- | --- | --- |
|  | Tempo para achar melhor solução já encontrada | Tempo para acharmos nossa melhor solução |
| *dantzig42* |  |  |
| *gr48* |  |  |
| *brazil58* |  |  |
| *gr120* |  |  |

1. **Simulated Annealing**
   1. O algoritmo

*Simulated Annealing,* é uma [meta-heurística](https://pt.wikipedia.org/wiki/Meta-heur%C3%ADstica) para [otimização](https://pt.wikipedia.org/wiki/Otimiza%C3%A7%C3%A3o) que consiste numa técnica de busca local probabilística, e se fundamenta numa analogia com a [termodinâmica](https://pt.wikipedia.org/wiki/Termodin%C3%A2mica).

A metaheurística usada é uma metáfora de um processo térmico, dito *annealing* ou *recozimento*, utilizado em [metalurgia](https://pt.wikipedia.org/wiki/Metalurgia) para obtenção de estados de baixa energia num sólido. O processo consiste de duas etapas: na primeira, a temperatura do sólido é aumentada para um valor próximo de 1100°C, na segunda, o resfriamento deve ser realizado lentamente até que o material se solidifique, sendo acompanhado e controlado esse arrefecimento.

De forma análoga, o algoritmo de *Simulated Annealing* substitui a solução atual por uma solução próxima (i.e., na sua vizinhança no espaço de soluções), escolhida de acordo com uma função objetivo e com uma variável (dita *Temperatura*, por analogia). Quanto maior for   , maior a probabilidade de aceitar soluções piores dentro da vizinhança calculada. À medida que o algoritmo progride, o valor de   é decrementado, começando o algoritmo a convergir para uma solução ótima.

* + 1. A solução inicial

Como solução inicial, usamos a solução calculada pelo nosso algoritmo já citado antes de Busca Gulosa em profundidade limitada com poda, que já nos fornece uma boa solução e por isso melhora o resultado da *Simulated Annealing*. Testamos antes com soluções iniciais randômicas, mas os resultados foram piores.

* + 1. A temperatura
    2. A probabilidade
    3. As vizinhanças
       1. 2 – Opt
       2. Swap
  1. Os resultados

|  |  |  |
| --- | --- | --- |
|  | Melhor solução já encontrada | Nossa melhor solução |
| *dantzig42* |  |  |
| *gr48* |  |  |
| *brazil58* |  |  |
| *gr120* |  |  |
| *pa561* |  |  |

* 1. O tempo

|  |  |  |
| --- | --- | --- |
|  | Tempo para achar melhor solução já encontrada | Tempo para acharmos nossa melhor solução |
| *dantzig42* |  |  |
| *gr48* |  |  |
| *brazil58* |  |  |
| *gr120* |  |  |

1. **Outros algoritmos**

Alguns algoritmos foram implementados enquanto desenvolvíamos o trabalho, no entanto não deram resultados tão bons quanto os citados acima.

* 1. Busca gulosa usando Kruskal
  2. Busca Local

1. **Conclusão**