

Nomes:

Felipe Guilermmo Santuche Moleiro - 10724010 - felipemoleiro@usp.br

Mateus Prado Santos - 10851707 - mateus.prado@usp.br

Matheus Lopes Rigato - 10260462 - matheus_rigato@usp.br

Victor Vieira Custodio Reis - 10734686 - reisvictor@usp.br

Vinicius Ricardo Carvalho - 10724413 - vinicius_carvalho@usp.br

Relatório de Trabalho

Programação Matemática - SME0110

Etapa 1) Tarefas 1 e 2

Na Primeira parte do trabalho, fizemos uma modelagem bem simples, com restrições de subciclo, em que criamos uma restrição para cada subciclo possível existente impedindo que o número de caminhos nele seja igual ao de vértices, desta forma, impedindo subciclos. A modelagem foi retirada dos slides mostrados em aula e do pdf disponibilizado como material extra, “A comparative analysis of several asymmetric traveling salesman problem formulations”. Sendo assim, a modelagem ficou da seguinte forma:

Parâmetros:

c_{ij} : Custo de mover o telescópio de uma galáxia i para outra galáxia j com:

$i = 1, \dots, n$

$j = 1, \dots, n$

e $i \neq j$

onde $c_{ij} \in \mathbb{R}$;

N : Conjunto que possui todas as galáxias do problema,

ou seja, $N = \{1, 2, \dots, n\}$ onde cada elemento i é uma das n galáxias;

S : Um subconjunto de N com número de elementos de 2 até $|N| - 1$;

Variáveis:

x_{ij} : Variável binária que indica se o telescópio se moveu da galáxia i até a galáxia j , com:

$i = 1, \dots, n$

$j = 1, \dots, n$

e $i \neq j$

onde $x_{ij} \in \{0, 1\}$;

Função Objetivo:

$$\min \sum_{i=1}^n \sum_{j=1, j \neq i}^n c_{ij} x_{ij}, \quad i, j \in N$$

// Minimizar o custo ao mover o telescópio por todo o trajeto, sendo que não devem ser considerados casos onde j é igual a i , já que esses são casos onde estamos tentando mover o telescópio para a posição em que ele já está;

Restrições:

$$\sum_{j=1, j \neq i}^n x_{ij} = 1, \quad i = 1 \dots n$$

// A partir de uma galáxia i , o telescópio só se move para uma única outra galáxia j ;

$$\sum_{i=1, i \neq j}^n x_{ij} = 1, \quad j = 1 \dots n$$

// O telescópio só se move para uma galáxia j vindo de uma única outra galáxia i ;

// As duas restrições acima, em conjunto, garantem que sempre há apenas uma entrada e apenas uma saída de cada galáxia no trajeto do telescópio;

$$\sum_{i, j \in S, i \neq j} x_{ij} \leq |S| - 1, \quad S \subseteq N - \{1\}, \quad |S| \geq 2$$

// Garante que não há sub-ciclos, ou seja, queremos todas as galáxias conectadas em um único ciclo;

// Com esta última restrição, garantimos que o número de caminhos, em qualquer subconjunto S , seja menor que o número de elementos nesse subconjunto. Desta forma, garantimos que não será fechado nenhum subciclo em nenhum subconjunto S , ou seja, só será fechado o ciclo no conjunto N , de todos os elementos.

Problemas Dessa Modelagem:

Essa modelagem funciona para casos muito pequenos, como para o caso do toy problem, mas para casos maiores temos um problema de muitas restrições sendo criadas, o que causa o algoritmo ser muito lento e usar muita memória do computador, tornando muitos problemas inviáveis de serem resolvidos utilizando essa modelagem.

Uma maneira de continuar utilizando essa modelagem seria utilizar restrições relaxadas/planos de cortes para esse modelo. Dessa maneira, fazemos o modelo solucionar o problema sem restrições de subciclo e vamos adicionando restrições de subciclos conforme eles vão surgindo. Esta seria uma solução possível, entretanto nosso grupo decidiu utilizar outras formulações encontradas no material complementar da disciplina, “A comparative analysis of several asymmetric traveling salesman problem formulations”. A nova modelagem utilizada foi a Miller, Tucker and Zemlin (MTZ) formulation, que é a seguinte:

Nova Modelagem

Parâmetros:

c_{ij} : Custo de mover o telescópio de uma galáxia i para outra galáxia j com:

$i = 1, \dots, n$

$j = 1, \dots, n$

onde $c_{ij} \in \mathbb{R}$;

Variáveis:

x_{ij} : Variável binária que indica se o telescópio se moveu da galáxia i até a galáxia j , com:

$i = 1, \dots, n$

$j = 1, \dots, n$

onde $x_{ij} \in \{0, 1\}$;

u_i : variável inteira que representa a ordem de visitação de cada nó, do nó 2 ao n (o nó 1 é considerando o primeiro sempre), com $i = 2 \dots n$ e $u_i \in \{1, \dots, n - 1\}$

Função Objetivo:

$$\min \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij}, \quad i, j \in N$$

//Minimizar o custo ao mover o telescópio por todo o trajeto

Restrições:

$$\sum_{j=1}^n x_{ij} = 1, \quad i = 1 \dots n$$

// A partir de uma galáxia i , o telescópio só se move para uma única outra galáxia j ;

$$\sum_{i=1}^n x_{ij} = 1, \quad j = 1 \dots n$$

// O telescópio só se move para uma galáxia j vindo de uma única outra galáxia i ;

// As duas restrições acima, em conjunto, garantem que sempre há apenas uma entrada e apenas uma saída de cada galáxia no trajeto do telescópio;

$$u_i - u_j + (n-1)x_{ij} \leq n-2, \quad i, j = 2, \dots, n$$

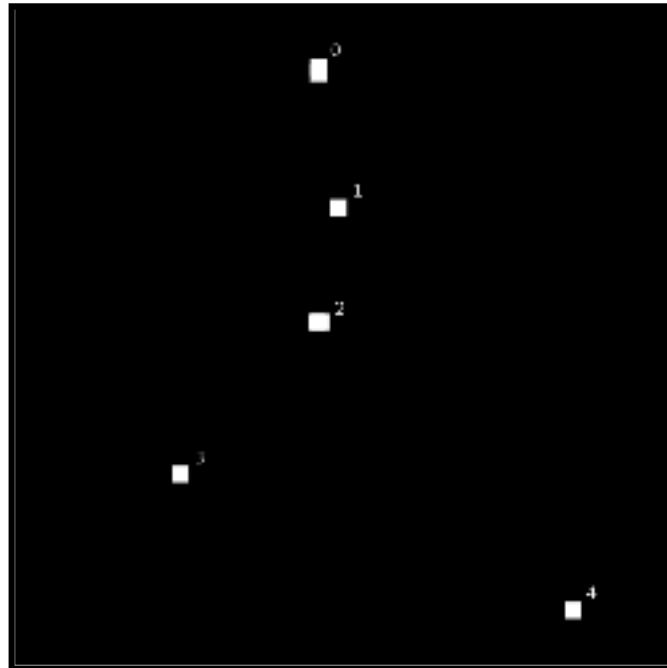
$$1 \leq u_i \leq n-1, \quad i = 2, \dots, n$$

// Garante que não há sub-ciclos, ou seja, queremos todas as galáxias conectadas em um único ciclo;

// Com esta última restrição, garantimos que só pode existir uma conexão entre i e j se j vier depois de i na ordem de visitaç o, e como u_i est  limitado de 1 at  $n-1$, temos uma correspond ncia de uma posi o para cada u_i , e cada u_i vai receber o valor da ordem de visita o.

Toy Problem:

O nosso toy problem é baseado na constelação de Câncer em que temos as estrelas aproximadamente da seguinte forma:



(Obs: imagem está no tamanho 1000x1000(em pixels))

O desenho acima foi feito utilizando-se um programa de edição de imagens(GIMP) inspirado em uma imagem retirada da internet(referência [1]).

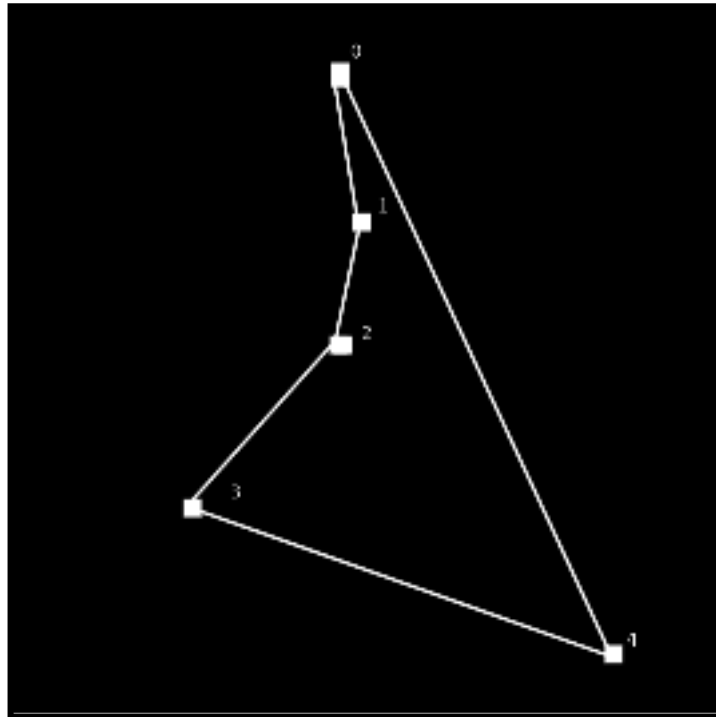
As medidas que fizemos não estão necessariamente iguais às da constelação real, então, utilizando uma ferramenta de medição, calculamos a distância entre os pixels desta imagem e chegamos na seguinte matriz de distância:

Custo	0	1	2	3	4
0	0	215	390	655	915
1	215	0	183	467	717
2	390	183	0	315	592
3	655	467	315	0	627
4	915	717	592	627	0

Com esses dados em mãos, nós temos tudo o que é necessário para computar o problema utilizando nossa modelagem.

Aplicando esses parâmetros no nosso modelo e utilizando a ferramenta de otimização em python(Na primeira parte do trabalho utilizamos o OR-TOOLS com o

SCIP, na segunda utilizamos o python-MIP com CBC), nós chegamos na seguinte resposta:



(Obs: imagem original está no tamanho 1000x1000(em pixels))

Ou seja, $x_{01} = x_{12} = x_{23} = x_{34} = x_{40} = 1$, e as outras variáveis são 0, em que o custo mínimo de distância para percorrer todas as estrelas uma única vez e voltar para a inicial é de: 2255 pixels.

Etapa 2) Tarefas 3,4 e 5

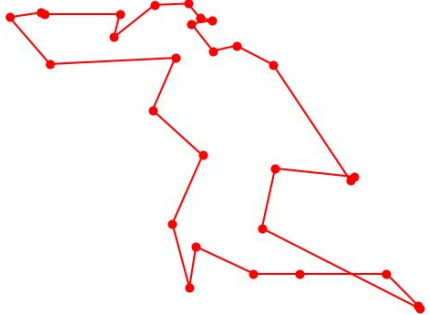
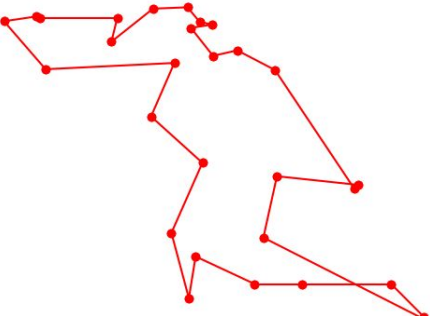
Método de análise em cada Instância

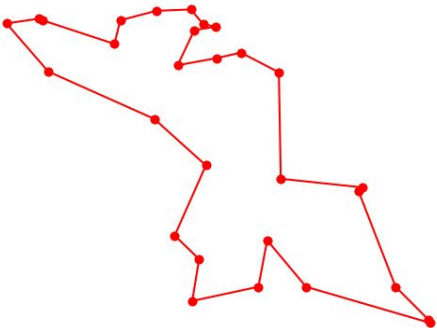
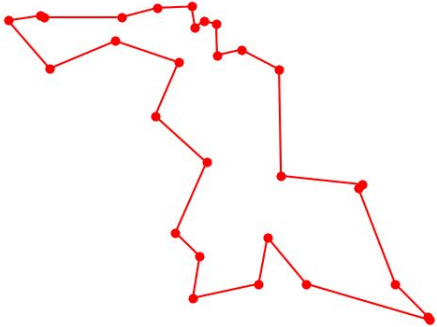
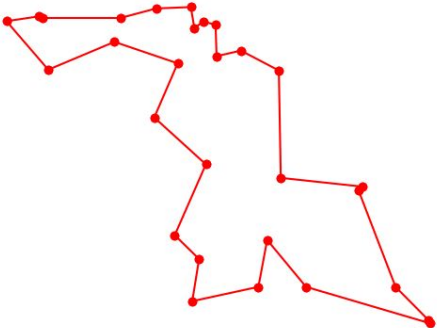
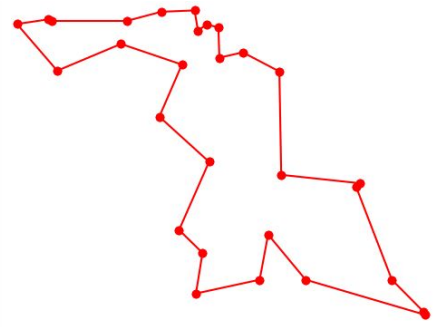
Para cada instância, vamos rodar 6 vezes testando diferentes parâmetros de configuração do solver utilizado (Python-MIP, CBC). Todas as execuções serão feitas em um processador Intel® Core™ i5-3470 CPU @ 3.20GHz, e apesar de ser um processador antigo, conseguimos respostas interessantes com ele. As primeiras 3 execuções serão sem heurística alguma, e apenas vamos modificar a ênfase de busca do algoritmo, ou seja, alteramos entre a ênfase padrão, a ênfase de busca por factibilidade e a ênfase de busca por otimalidade. Para as próximas 3 execuções, iremos alterar a ênfase de busca também, porém utilizando heurísticas para tentar chegar em uma solução melhor.

Heurística utilizada: vizinhança mais próxima e 2-OPT

Essa heurística é um algoritmo que implementa uma abordagem gulosa que foca em ver qual é o ponto mais próximo em cada iteração. Essa heurística tem um resultado bom, mas existem alguns problemas com ela, o principal é o fato do último passo do algoritmo ser forçado(ao chegar no último nó, só se tem uma opção, que é voltar para a origem) e, normalmente, esse último passo adiciona um peso muito grande à caminhada. Para melhorar essa solução encontrada, podemos utilizar juntamente uma heurística de melhoria 2-OPT que tenta fazer trocas entre as arestas duas a duas tentando diminuir o valor da solução. Na prática, essa heurística acaba trocando arestas que se cruzam por arestas separadas.

- **Instância 1: Western Sahara**

Sem heurística e Ênfase padrão:	
Sem heurística e Ênfase em Viabilidade:	

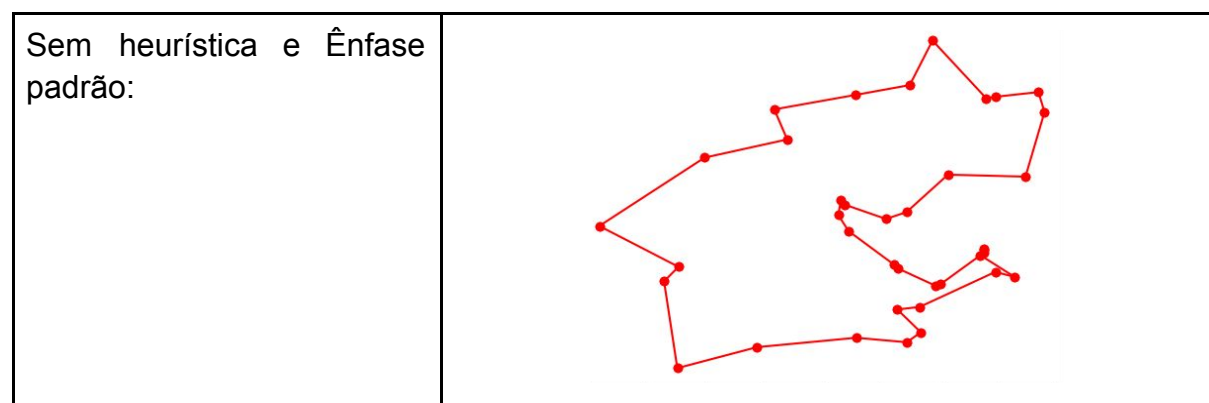
<p>Sem heurística e Ênfase em Otimalidade:</p>	
<p>Com heurística e Ênfase padrão:</p>	
<p>Com heurística e Ênfase em Viabilidade:</p>	
<p>Com heurística e Ênfase em Otimalidade:</p>	

	Sem heurística Ênfase padrão	Sem heurística Ênfase Viabilidade	Sem heurística Ênfase Otimalidade	Com heurística Ênfase padrão	Com heurística Ênfase Viabilidade	Com heurística Ênfase Otimalidade
Limite Primal	30320,465	30320,465	27601,174	27748,710	27748,710	27748,710
Limite Dual	24624,145	24624,145	25405,069	24715,088	24715,088	25034,424
Gap	18,787 %	18,787 %	7,96 %	10,932 %	10,932 %	9,782 %
Gap Relativo	9,845 %	9,845 %	0,0 %	0,528 %	0,528 %	0,528 %
Nós visitados	316947	316776	293302	197830	197555	221169
Tempo de execução (CPU Time)	30 min	30 min	30 min	30 min	30 min	30 min

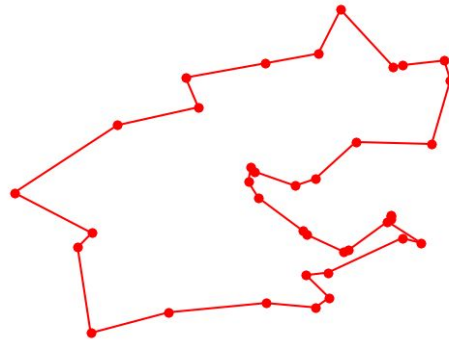
Nesta Instância pequena, rodando sem heurística, conseguimos ver que a busca com ênfase em otimalidade compensou pois, utilizando esse foco, conseguimos encontrar a solução ótima para o problema, apesar do algoritmo não ter chegado a uma conclusão definitiva disso por seu limite dual mostrar a possibilidade de existir algo melhor(apesar de sabermos que não).

Colocando uma heurística como base de busca, melhoramos as respostas para o caso padrão e com factibilidade, porém, para a busca com ênfase em utilidade, tivemos uma pequena piora, muito provavelmente pela heurística ter guiado o algoritmo para um caminho factível, mas que não levaria ao ótimo rapidamente.

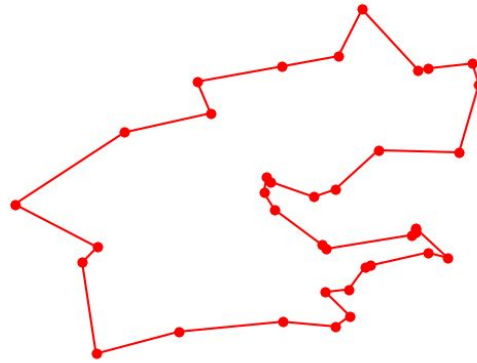
- **Instância 2: Djibouti**



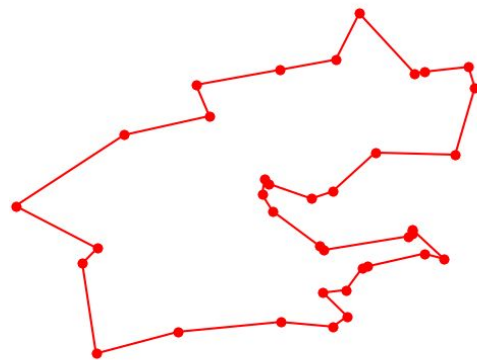
Sem heurística e Ênfase em Viabilidade:



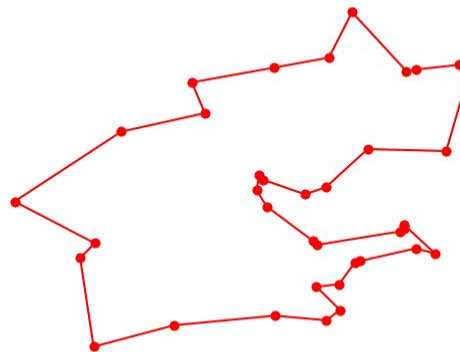
Sem heurística e Ênfase em Otimalidade:

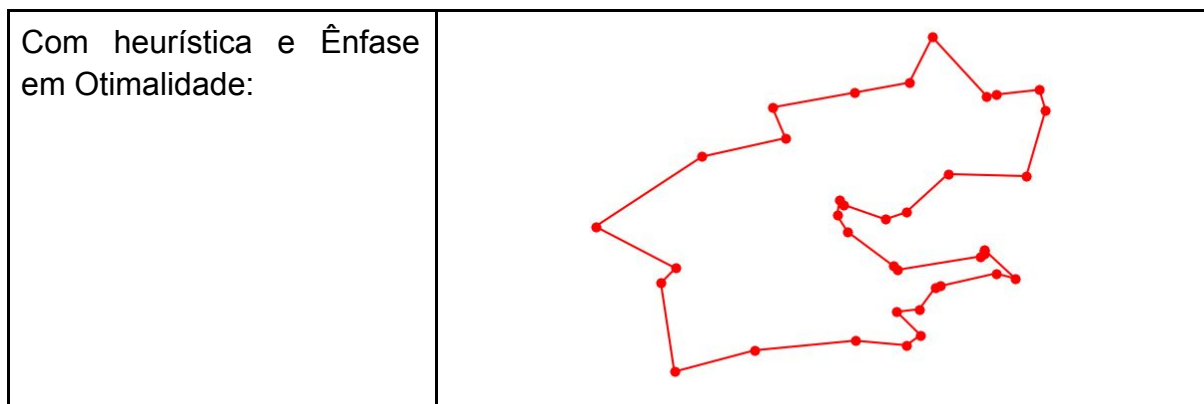


Com heurística e Ênfase
padrão:



Com heurística e Ênfase em Viabilidade:





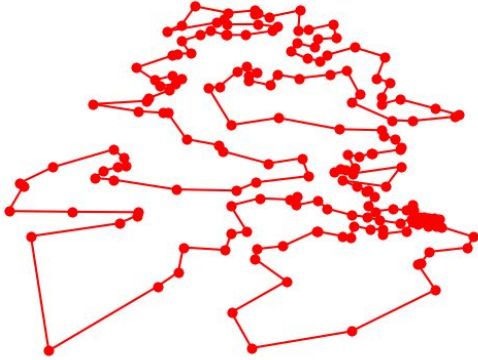
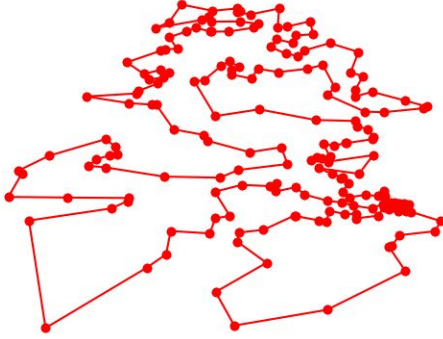
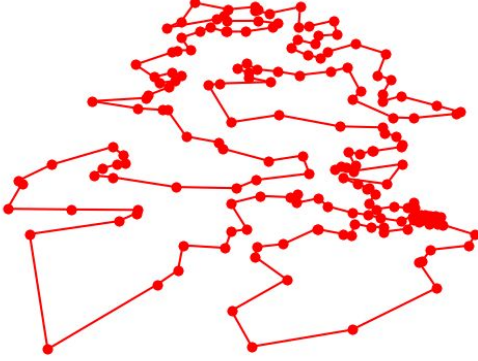
	Sem heurística Ênfase padrão	Sem heurística Ênfase Viabilidade	Sem heurística Ênfase Otimalidade	Com heurística Ênfase padrão	Com heurística Ênfase Viabilidade	Com heurística Ênfase Otimalidade
Limite Primal	6738.707	6738.707	6659.432	6659.432	6659.432	6659.432
Limite Dual	6207.975	6204.841	6418.907	6436.386	6436.386	6659.432
Gap	7,876%	7,922%	3,612%	3,349%	3,349%	0%
Gap Relativo	1,243%	1,243%	0%	0%	0%	0%
Nós visitados	44020	43653	105177	83829	83706	74650
Tempo de execução (CPU Time)	30 min	30 min	30 min	30 min	30 min	17 min

Nesta instância, podemos ver como a busca com ênfase em otimalidade foi boa, já que estamos trabalhando com um problema pequeno e esperamos chegar bem próximo do ótimo. No caso sem heurística, a busca com ênfase em otimalidade foi a única que conseguiu encontrar a solução ótima para o problema(apesar de sem certeza).

Além disso, o uso da heurística permitiu o encontro da solução ótima com total certeza (GAP 0%) no caso com heurística e busca por otimalidade, mostrando a utilidade de utilizar uma heurística de solução inicial para acelerar a busca do algoritmo.

- **Instância 3: Qatar**

Sem heurística e Ênfase padrão:	Não encontrado.
---------------------------------	-----------------

Sem heurística e Ênfase em Viabilidade:	Não encontrado.
Sem heurística e Ênfase em Otimalidade:	Não encontrado.
Com heurística e Ênfase padrão:	
Com heurística e Ênfase em Viabilidade:	
Com heurística e Ênfase em Otimalidade:	

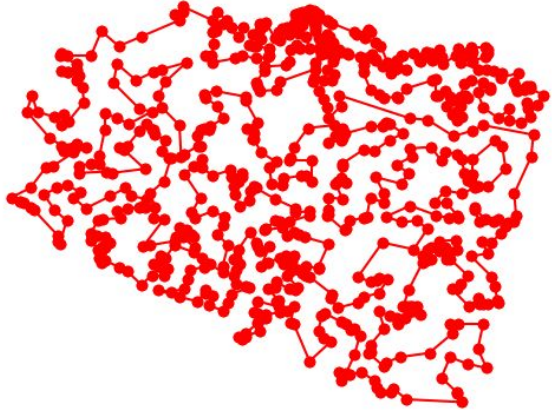
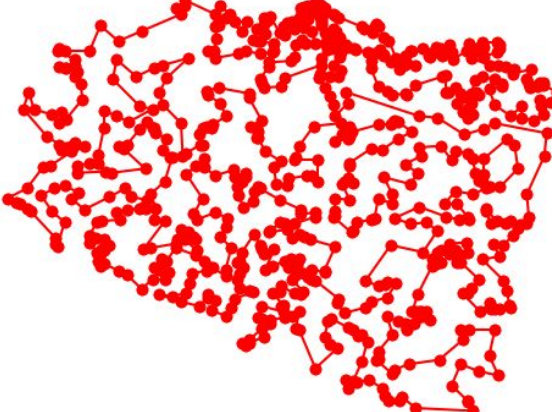
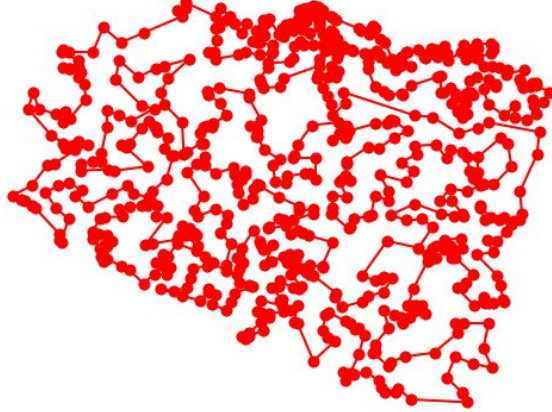
	Sem heurística Ênfase padrão	Sem heurística Ênfase Viabilidade	Sem heurística Ênfase Otimalidade	Com heurística Ênfase padrão	Com heurística Ênfase Viabilidade	Com heurística Ênfase Otimalidade
Limite Primal	Não encontrado	Não encontrado	Não encontrado	9850.653	9850.653	9873.764
Limite Dual	8762.602	8762.602	8946.703	8747.405	8747.405	8948.277
Gap	Infinito	Infinito	Infinito	11,200%	11,200%	9,373%
Gap Relativo	Infinito	Infinito	Infinito	5,331%	5,331%	5,579%
Nós visitados	2668	2664	508	899	877	339
Tempo de execução (CPU Time)	30 min	30 min	30 min	30 min	30 min	30 min

Nesta instância, podemos ver que, conforme vamos entrando em problemas de ordem maior, rodar o algoritmo sem heurística nenhuma pode demorar muito para encontrar uma solução, mesmo utilizando uma ênfase de viabilidade. É possível também visualizar que, sem heurística, obtemos um melhor limite dual com a ênfase em otimalidade, porém, muito provavelmente, essa busca está mais longe de encontrar uma resposta qualquer que a busca com ênfase em viabilidade, entretanto, ela chegaria na solução ótima mais rapidamente.

Utilizando heurísticas, podemos ver que somos capazes de encontrar soluções iniciais e até mesmo melhorá-las um pouco utilizando o solver, além de ter uma perspectiva de quão perto da melhor resposta estamos.

- **Instância 4: Uruguay**

Sem heurística e Ênfase padrão:	Não encontrado.
Sem heurística e Ênfase em Viabilidade:	Não encontrado.
Sem heurística e Ênfase em Otimalidade:	Não encontrado.

<p>Com heurística e Ênfase padrão:</p>	
<p>Com heurística e Ênfase em Viabilidade:</p>	
<p>Com heurística e Ênfase em Otimalidade:</p>	

	Sem heurística Ênfase padrão	Sem heurística Ênfase Viabilidade	Sem heurística Ênfase Otimalidade	Com heurística Ênfase padrão	Com heurística Ênfase Viabilidade	Com heurística Ênfase Otimalidade
Limite Primal	Não encontrado	Não encontrado	Não encontrado	84956,303	84956,303	84956,303
Limite Dual	64767,609	64767,609	64767,609	64932,847	64767,609	64932,847
Gap	Infinito	Infinito	Infinito	23,570 %	31,171 %	23,570 %
Gap Relativo	Infinito	Infinito	Infinito	7,385 %	7,385 %	7,385 %
Nós visitados	0	0	0	0	0	0
Tempo de execução (CPU Time)	30 min	30 min	30 min	30 min	30 min	30 min

Nesta instância, tivemos muitas dificuldades de executar o algoritmo do solver pois o mesmo ficava muito tempo rodando algoritmos de corte no primeiro nó da árvore. Entretanto, mesmo desativando o algoritmo de cortes do solver, ele chegava a visitar por volta de 200 nós da árvore, mas sem nenhuma melhoria na solução.

A única questão que foi possível observar utilizando o solver foi utilizá-lo para tentar verificar o quão boa estava nossa heurística a partir do GAP. Podemos ver que, no último teste, chegamos a um GAP de 23,57%, nos dando confiança de que a solução ótima não está tão longe da encontrada pela heurística. Podemos ver que, mesmo que o algoritmo acabe não achando respostas com casos muito grandes, eles podem nos ajudar a verificar a proximidade de uma heurística à solução real.

Referências:

[1]: Imagem usada como inspiração para o desenho da Constelação de Câncer no toy problem: [Cancer](#).

Carlson, S. (1997). Algorithm of the gods. Scientific American 276 (3), 121–123.

Waterloo, U. (2020). University of Waterloo, [National Traveling Salesman Problems](#).

Temel Öncan, 'I. Kuban Altinel, Gilbert Laporte(2007). A comparative analysis of several asymmetric traveling salesman problem formulations. Computers & Operations Research 36 (2009) 637 – 654.

Explicação e Pseudocodigo do Algoritmo 2-OPT: <https://en.wikipedia.org/wiki/2-opt>

Documentação Python-MIP:

<https://python-mip.readthedocs.io/en/latest/classes.html>