Universidade Federal da Paraíba

Centro de Informática

Disciplina: Análise e Projeto de Algoritmos (2020.2)

Professor: Bruno Petrato Bruck

Projeto Final

Descrição do problema

Robin começou a trabalhar em uma startup que está se especializando em realizar diversos tipos de pesquisas (de opinião, mercado, satisfação, etc) na Paraíba. Geralmente, quando um novo cliente contrata a empresa de Robin, são realizadas diversas reuniões com o cliente para definir o questionário que será aplicado, o público alvo, orçamento, prazos, etc. Após essa fase inicial, é feita a seleção de um conjunto de endereços nos quais serão aplicados o questionário. Esses endereços podem estar espalhados em uma cidade ou até mesmo no estado, e são escolhidos baseado no público alvo e nos objetivos da pesquisa. Nessa fase, diariamente, os agentes da empresa recebem um conjunto de endereços que devem visitar e aplicar o questionário.

Após algumas semanas trabalhando na empresa, Robin percebeu uma oportunidade para ajudar a empresa a concluir as pesquisas mais rapidamente e reduzir custos de modo geral. Ele notou que na fase de aplicação dos questionários, a seleção dos pontos que cada agente visita diariamente é feita de forma muito manual e baseada somente na experiência da pessoa responsável por essa tarefa. Para piorar, a ordem de visitação dos endereços e a rota percorrida fica totalmente a critério do agente. Robin, tendo cursado a disciplina de Análise e Projeto de Algoritmos durante a graduação e aprendido sobre algoritmos eficientes, logo percebeu que esse processo poderia ser otimizado e possivelmente resultaria em uma redução de custos.

Antes de apresentar sua proposta aos fundadores da empresa, ele reservou um tempo para pensar em uma descrição mais precisa do problema e expor sua ideia claramente. Robin escreveu o seguinte: "Seja $G = (V, A)$ um grafo onde, V é o conjunto de vértices que representa os endereços, e A é o conjunto de arcos do grafo. Um arco $(i, j) \in A$ representa o caminho que o agente deve percorrer para ir do endereço i ao j e o tempo (em minutos) para percorrer tal arco é dado por t_{ij}. Além disso, a empresa pode estipular que qualquer agente pode realizar no máximo p questionários por dia. Sendo assim, nesse problema deseja-se atribuir um conjunto de endereços para cada um dos agentes e estabelecer a rota que eles deverão seguir (o percurso) de forma a minimizar o tempo total gasto e garantir que: (i) todos os endereços são visitados exatamente uma vez; (ii) cada agente inicia e termina sua rota na sede da empresa; (iii) a rota de cada um dos agentes não visita mais que p endereços. Considerando que os agentes são contratados conforme a demanda, vamos assumir que não existe um limite para o número de agentes disponíveis."

Instruções

O projeto final deve ser realizado em grupo de **3 integrantes** e vale 10 pontos, relativos à terceira nota da disciplina. Cada grupo deve desenvolver um algoritmo eficiente de busca local (ou metaheurística) para o seguinte problema de otimização descrito acima.

Cada grupo deverá participar da ***Copa APA***. Essa competição é uma atividade que faz parte do Projeto Final e visa estimular e motivar os estudantes a desenvolver o melhor algoritmo possível para a resolução de instâncias (cenários) difíceis do problema. É **importante enfatizar** que a nota do projeto **não depende** do desempenho na Copa. Entretanto, os grupos que ficarem nas três primeiras posições do ranking receberão pontos extras na menor nota da disciplina de acordo com o seguinte:

- 1° lugar: 3 pontos
- 2° lugar: 2 pontos
- 3° lugar: 1 ponto

Em relação à linguagem de programação, poderão ser utilizadas **somente** uma das seguintes linguagens: C/C++, Python ou Julia.

Etapas e prazos

Esse projeto contém os seguintes entregáveis:

- Implementação de ao menos uma heurística de construção;
- Implementação dos movimentos de vizinhança (**Mínimo 3**).
- Implementação do algoritmo de busca local chamado VND (Variable Neighborhood Descent);
- Implementação de uma metaheurística (OPCIONAL);
- Resultados computacionais: **criar uma tabela** que contenha os resultados obtidos pela(s) heurística(s) construtiva(s) e pelo VND, e que compare tais resultados com a solução ótima de cada instância. Essa tabela deverá conter os seguintes dados para cada heurística construtiva e para o VND:
  - Média do valor da solução (em no **mínimo 10 execuções** para cada instância)
  - Melhor solução encontrada
  - Média do tempo gasto pelo respectivo algoritmo
  - GAP para a solução ótima

observação: Caso decida implementar a metaheurística, é necessário adicionar uma coluna de resultados para ela na tabela.

O prazo de entrega do projeto é até o dia **15 de Julho de 2021**. Devem ser enviados via Moodle, o código do projeto, um arquivo pdf com a tabela de resultados computacionais e um link para o vídeo de apresentação.

Avaliação

Cada grupo deverá gravar um vídeo de no máximo 15 minutos apresentando seu projeto. A nota do projeto leva em consideração diversos critérios, como demonstração de entendimento do código na apresentação, qualidade do código, eficiência dos algoritmos implementados, qualidade dos resultados obtidos, dentre outros.

Dicas

Como calcular o valor da medida GAP: Suponha que desejamos calcular o valor GAP para o resultado da heurística construtiva para a instância chamada *nome_instancia*. Supondo que o valor encontrado pela heurística para essa instância é dado por $valor_{heuristica}$ e o valor ótimo para essa instância é $valor_{otimo}$, o cálculo do GAP é realizado da seguinte forma:

$$gap = \left(\frac{valor_{heuristica} - valor_{otimo}}{valor_{otimo}} \right) \times 100$$

Note que o valor do gap é dado em percentagem (%) e indica a "distância" da solução, no caso, da heurística construtiva para o valor ótimo.

Para calcular o GAP dos resultados obtidos pelo VND basta substituir $valor_{heuristica}$ pelo valor encontrado pela VND.

Exemplo de tabela de resultados:

		Heurística construtiva			VND		
	ótimo	valor solução	tempo	gap	valor solução	tempo	gap
instancia1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
instancia2	0.0	0.0	0.0	0.0	0.0	0.0	0.0
instancia3	0.0	0.0	0.0	0.0	0.0	0.0	0.0
instancia4	0.0	0.0	0.0	0.0	0.0	0.0	0.0