Licenciatura em Ciência de Dados - 2º ano

Trabalho individual 2

Unidade Curricular de Otimização Heurística

22 de maio de 2024

Discentes: João Dias nº 110305

2023/2024

iscte INSTITUTO UNIVERSITÁRIO DE LISBOA

Índice

a) Descreva por palavras em que consiste uma solução admissível para o problema.

Para dar resposta a este problema, uma solução admissível consiste em selecionar **quatro zonas da herdade** que atendam aos seguintes critérios:

1. As quatro zonas escolhidas **não podem ser adjacentes** umas às outras, ou, por outras palavras e por uma questão de simplificação do problema, não podem ter fronteiras em comum.
2. A **soma dos índices de impacto ambiental** das quatro zonas selecionadas **não pode ultrapassar o valor de 8**. Note-se que cada zona tem um índice de impacto ambiental que varia de 1 a 4, e a seleção deve garantir que a soma destes índices seja, no máximo, 8.

O objetivo é selecionar as quatros zonas de forma a **maximizar** a soma das **rentabilidades** esperadas, tendo em conta as restrições/critérios acima definidos.

b) Defina uma heurística que lhe permita determinar uma solução admissível para o problema.

Tendo em conta o que foi feito no ponto a), para a definição da heurística vamos percorrer as seguintes etapas (note-se que poderiam ter sido definidas outras regras e outras heurísticas desde que os pontos descritos na alínea a) fossem respondidos):

1. Seleção **aleatória** inicial de **duas zonas** florestais:
    a. Da lista completa de 30 zonas, **selecionar aleatoriamente** uma delas.
    b. **Verificar** se a zona escolhida está na lista de **zonas não disponíveis**, já que não podem existir fronteiras em comum e não se pode selecionar duas vezes a mesma zona. Se a zona não estiver disponível, voltar ao passo 1 a).

c. Somar o valor do impacto ambiental da zona escolhida com o total da **soma dos índices de impacto ambiental** das zonas já selecionadas e verificar se é **menor ou igual que 6** (explicação para o valor máximo de referência ser 6 no ponto 1 e)). Se o valor for superior a 6, voltar ao passo 1 a).
d. **Armazenar a zona escolhida** e as **zonas adjacentes** na lista de zonas não disponíveis para que seja possível fazer a verificação descrita no ponto 1 b). Armazenar a zona escolhida numa lista de zonas escolhidas.
e. **Repetir** este procedimento **até se obterem aleatoriamente duas zonas** florestais cuja soma dos seus índices de impacto ambiental seja menor ou igual que 6. Note-se que a definição do valor máximo de referência 6 é para **permitir** que possam ser **ainda adicionadas 2 zonas** nas etapas seguintes, com valores mínimos de impacto ambiental de 1, de forma a que seja possível satisfazer o objetivo de selecionar quatro zonas.

2. Seleção da 3ª zona:
    a. Se a **dimensão** da **lista de zonas escolhidas** for **igual a 2**, **ordenar de forma decrescente** as zonas pelos valores da **rentabilidade esperada** e percorrendo esta lista ordenada selecionar a **zona** com **maior rentabilidade** esperada. Em caso de **empate** na rentabilidade, selecionar a zona com **menor impacto ambiental**, se possível. Caso não seja possível selecionar **aleatoriamente** uma das observações empatadas.
    b. **Verificar** se a zona escolhida em 2 a) está na lista de **zonas não disponíveis**. Se a zona não estiver disponível, voltar ao passo 2 a), e escolher a **zona imediatamente a seguir no ranking**.
    c. Verificar se a soma dos índices de impacto ambiental das duas zonas selecionadas aleatoriamente com o valor do índice de impacto ambiental da 3ª zona escolhida **é menor ou igual que 7**, de forma a **deixar "espaço"** para o **impacto ambiental** mínimo

da seleção da **4ª e última zona**. Se o valor for superior a 7, voltar ao passo 2 a) e escolher a zona imediatamente a seguir no ranking.

d. **Armazenar** a **zona escolhida** e as **zonas adjacentes** na lista de zonas não disponíveis. E, armazenar a zona escolhida na lista de zonas escolhidas.

3. Seleção da 4ª zona:
    a. Se a **dimensão** da **lista de zonas escolhidas** for **igual a 3**, **ordenar** de forma **decrescente** as zonas pelos valores da **rentabilidade** esperada e percorrendo esta lista ordenada selecionar a **zona com maior rentabilidade** esperada. Em caso de **empate** na rentabilidade, selecionar a zona com **menor impacto ambiental**, se possível. Caso não seja possível selecionar **aleatoriamente** uma das observações empatadas.
    b. **Verificar** se a zona escolhida em 3 a) está na lista de **zonas não disponíveis**. Se a zona não estiver disponível, voltar ao passo 3 a), e escolher a **zona imediatamente a seguir no ranking**.
    c. Verificar se a soma dos índices de impacto ambiental com o valor do índice de impacto ambiental da 4ª zona escolhida é **menor ou igual que 8**, de forma a respeitar a restrição do problema. Se o valor for superior a 8, voltar ao passo 3 a), e escolher a zona imediatamente a seguir no ranking.
    d. **Armazenar** a **zona escolhida** e **devolver a lista com as zonas escolhidas**.

c) Tendo em conta a alínea b), determine uma solução admissível para o problema

Tendo em conta a heurística definida na alínea b), começamos por selecionar **aleatoriamente duas zonas não adjacentes** e cuja soma dos **valores dos seus impactos ambientais** seja **menor ou igual que 6**. Por exemplo:

- Lista de zonas escolhidas <- [A1, A30]
- Lista de zonas não disponíveis <- [A1, A7, A2, A30, A29, A25]
- Soma do impacto ambiental <- 2 (impacto ambiental de A1) + 3 (impacto ambiental de A30) = 5

Após este primeiro passo, e já com a definição aleatória das duas primeiras zonas, passa-se para a 2ª etapa, onde se procurará **fazer a escolha da 3ª zona**, começando por **ordenar as zonas <u>disponíveis</u>** pela sua **rentabilidade esperada** (ordem **decrescente**) de forma a obter a tabela 1:

Zona	Rentabilidade Esperada	Impacto Ambiental
A14	5000	4
A9	4700	4
A12	4700	4
A11	4600	4
A23	4600	3
A8	4300	2
A20	4200	1
A21	4200	4
A18	3300	1
A6	3200	2
A5	3000	2
A19	3000	1
A16	2900	1

A22	2400	1
A4	2100	4
A26	2000	2
A3	1900	3
A15	1900	1
A24	1800	1
A13	1700	1
A27	1700	1
A28	1700	3
A17	1300	4
A10	1100	3

Tabela 1 - Zonas ordenadas pela rentabilidade esperada

Como na **etapa 2**, definimos que a **soma** dos **índices de impacto ambiental** das duas zonas selecionadas aleatoriamente com o valor do índice de impacto ambiental da 3ª zona escolhida tem de ser **menor ou igual que 7** - percorrendo a tabela 1 ordenada pela rentabilidade - não se podem adicionar as primeiras zonas da tabela - A14, A9, A12, A11 e A23 - por terem impactos ambientais de 4 ou 3. Como a soma do impacto ambiental das zonas A1 e A30 é 5, se considerássemos as opções A14, A9, A12 e A11 ficaríamos com um impacto ambiental total de 5 + 4 = 9, que não respeitava a restrição imposta pelo problema. Por outro lado, se considerássemos a zona imediatamente a seguir a estas (A23), ficaríamos com um total de 5 + 3 = 8, não deixando espaço para a seleção da 4ª e última zona. Nesse sentido, seguiu-se com a zona imediatamente a seguir em termos de rentabilidade esperada: a zona A8.

- Lista de zonas escolhidas <- [A1, A30, A8]
- Lista de zonas não disponíveis <- [A1, A7, A2, A30, A29, A25, A8, A9, A13, A14]
- Soma do impacto ambiental <- 5 + 2 = 7

Na 3ª e última etapa vamos proceder à escolha da 4ª e última zona. A tabela 2 mostra a lista ordenada pela rentabilidade esperada das zonas disponíveis (excluindo

as observações associadas às zonas que se encontram na lista de zonas não disponíveis).

Zona	Rentabilidade Esperada	Impacto Ambiental
A12	4700	4
A11	4600	4
A23	4600	3
A20	4200	1
A21	4200	4
A18	3300	1
A6	3200	2
A5	3000	2
A19	3000	1
A16	2900	1
A22	2400	1
A4	2100	4
A26	2000	2
A3	1900	3
A15	1900	1
A24	1800	1
A27	1700	1
A28	1700	3
A17	1300	4
A10	1100	3

Tabela 2 - Atualização das zonas ordenadas pela rentabilidade esperada

Pela restrição da soma do impacto ambiental, não podemos considerar as 3 primeiras zonas da tabela 2 (A12, A11 e A23), já que a soma ultrapassa o valor máximo de referência (8). Nesse sentido, escolheu-se a zona imediatamente a seguir: a zona A20 e atualizou-se a lista de zonas escolhidas para [A1, A30, A8, A20].

Uma potencial **solução admissível** para o problema poderia ser cortar as árvores das zonas (não adjacentes) **A1, A30, A8 e A20**, que teriam um **índice de impacto ambiental total de 2 + 3 + 2 + 1 = 8**, e que teriam uma **rentabilidade esperada de 2600 + 1700 + 4300 + 4200 = 12800€**. Note-se que esta é apenas uma **solução admissível** para o problema, **obtida a partir da heurística** definida na alínea b), **não** tendo de ser por isso necessariamente de ser a **solução ótima** para o problema. Para tal, seria necessário **calcular todas** as **combinações possíveis**.

d) Defina a estrutura de vizinhança de uma solução.

Uma solução s2 é **vizinha** de s1, caso seja **obtida** a partir de s1 por **troca de uma zona selecionada por outra zona não selecionada**, que **não** esteja **adjacente** a nenhuma das zonas já selecionadas e que respeite a **restrição** relativa ao **impacto ambiental**.

Para gerar a vizinhança:

1. Identificar **todas as zonas** que **não estão na solução atual** e que **não são adjacentes** às zonas já **selecionadas**, com recurso a uma lista de zonas não disponíveis, composta pelas zonas que fazem parte da solução admissível e as suas fronteiras.
2. Para cada zona na solução atual, **substituir** essa **zona por uma das zonas identificadas no passo 1** (basta que se faça a **troca de uma zona** para gerar uma **solução vizinha**).
3. Verificar se a **nova solução** gerada pela substituição **respeita** a **restrição do impacto ambiental**. A restrição de **não adjacência** já é **respeitada** pelo facto da **lista gerada** no passo 1 **ter apenas zonas disponíveis e não adjacentes**. Caso a restrição do impacto ambiental não seja respeitada, voltar ao passo 1.
4. Atualizar o valor das listas de zonas escolhidas e de zonas não disponíveis com os novos valores considerados.

e) Tendo em conta a alínea d), determine uma solução vizinha da solução que apresentou na alínea c).

Em primeiro lugar consideremos a solução admissível obtida na alínea c):

- Lista de zonas escolhidas <- [A1, A30, A8, A20]
- Lista de zonas não disponíveis <- [A1, A7, A2, A30, A29, A25, A8, A9, A13, A14, A20, A19, A18, A22, A27]
- Soma do impacto ambiental <- 5 + 2 + 1 = 8

A partir da lista de zonas não disponíveis, indicada acima, selecionamos a **lista de zonas disponíveis** (**não selecionadas e não adjacentes**) que será: [A3, A4, A5, A6, A10, A11, A12, A15, A16, A17, A21, A23, A24, A26, A28]. Depois, vamos **substituir** A1 **por um elemento desta lista**; ou A30; ou A8 e ou por fim A20 por um elemento desta lista.

Se substituirmos na solução atual o A1 pelo A3 da lista de zonas disponíveis, por exemplo, ficamos com a solução [A3, A30, A8, A20]. Verificando **o impacto ambiental** tem-se que 3 (A3) + 3 (A30) + 2 (A8) + 1 (A20) = 9 (solução não admissível porque não respeita a restrição ambiental, sendo > 8).

Para determinar **uma solução vizinha admissível**, consideremos, por exemplo, **trocar a zona A1** da **solução atual** pela **zona A5** e ficamos com a solução [A5, A30, A8, A20]. Verificando o **impacto ambiental** tem-se que 2 (A5) + 3 (A30) + 2 (A8) + 1 (A20) = 8 (solução admissível porque respeita a restrição ambiental e de não adjacência). Note-se que com esta solução vizinha ter-se-ia uma rentabilidade esperada de 3000 + 1700 + 4300 + 4200 = 13200€, enquanto que na alínea c) a rentabilidade esperada era de apenas 12800€. Assim, a **solução vizinha** [A5, A30, A8, A20] é **melhor** que a **solução que tínhamos** em c) de [A1, A30, A8, A20]. De qualquer forma, esta solução vizinha é apenas mais uma **solução admissível**, não tendo de corresponder necessariamente a uma solução ótima.

Este **procedimento pode ser repetido** várias vezes para todas as zonas na solução atual, gerando uma lista de soluções vizinhas admissíveis que podem ser avaliadas e exploradas para se **procurar por uma solução ótima** (local ou global),

melhor do que a atual ou considerada **suficientemente boa**.

f) Na tentativa de determinar uma solução, o dono da herdade irá definir e implementar um algoritmo de Simulated Annealing. Tendo em conta a estrutura de vizinhança definida na alínea d), apresente o pseudocódigo para a aplicação do algoritmo.

Para implementar o algoritmo de Simulated Annealing (SA) utilizando a estrutura de vizinhança definida na alínea d) e a parametrização fornecida no enunciado, podemos seguir o pseudocódigo abaixo. Vamos supor que temos uma função **f(Solução)** que calcula a qualidade da solução devolvendo **a soma das rentabilidades esperadas** das zonas na solução, e que **GerarVizinho(Solução)** é a função que gera uma **solução vizinha** de acordo com a estrutura de vizinhança definida. Estas duas funções auxiliares encontram-se na parte final do pseudocódigo apresentado de seguida (note-se que para a função GerarVizinho(Solução) vamos ainda utilizar uma outra função auxiliar com o **mapeamento de fronteiras** para garantir que as zonas selecionadas não são adjacentes - esta função também está na parte final do pseudocódigo):

Pseudocódigo

Input:

SoluçãoInicial

*T = (t0, t1, t2, t3, t4), onde t0 = 0.2 * f(SoluçãoInicial) e tk = 0.5^k * t0 para k=1,2,3,4*

mk = 5, para k=0,1,2,3,4

Inicialização:

SoluçãoAtual = SoluçãoInicial

MelhorSolução = SoluçãoInicial

MelhorRentabilidade = f(SoluçãoInicial)

Para k de 0 até 4 faz-se:

Para i de 1 até mk faz-se:

SoluçãoVizinha = GerarVizinho(SoluçãoAtual)

RentabilidadeVizinho = f(SoluçãoVizinha)

Δ = f(SoluçãoAtual) - RentabilidadeVizinho

Se Δ < 0 ou exp(-Δ / tk) > random(0, 1) então:

SoluçãoAtual = SoluçãoVizinha

Se RentabilidadeVizinho > MelhorRentabilidade então:

MelhorSolução = SoluçãoVizinha

MelhorRentabilidade = RentabilidadeVizinho

End for loop

End for loop

Retornar MelhorSolução e MelhorRentabilidade

Função GerarVizinho(Solução):

Escolher uma zona aleatória da solução atual para substituir.

Escolher uma zona da lista de zonas disponíveis que não estão na solução atual e utilizar ZonasAdjacentes(Zona) para verificar a adjacência.

Substituir a zona escolhida na solução atual pela nova zona escolhida.

Se ImpactoAmbientalTotal <= 8:

Retornar a nova solução.

Função f(Solução):

1. Calcular a soma das rentabilidades esperadas das zonas na solução.

2. Retornar esta soma.

Função ZonasAdjacentes(Zona):

1. Retornar as zonas adjacentes da Solução conforme o mapeamento que temos no enunciado

e) Implemente o procedimento definido na alínea f).

O primeiro passo em termos da implementação do procedimento foi passar os dados do enunciado para um formato que seja possível de ser lido em Python. Para isso, foi criada uma lista de dicionários, onde cada dicionário representa uma zona com suas respectivas propriedades de rentabilidade e impacto ambiental, o que facilitou o acesso e a manipulação dos dados durante a execução do algoritmo. A lista de zonas ficou estruturada da seguinte maneira:

```
zonas = [
    {"zona": "A1", "rentabilidade": 2600, "impacto": 2},
    {"zona": "A2", "rentabilidade": 3400, "impacto": 4},
    {"zona": "A3", "rentabilidade": 1900, "impacto": 3},
    {"zona": "A4", "rentabilidade": 2100, "impacto": 4},
    {"zona": "A5", "rentabilidade": 3000, "impacto": 2},
    {"zona": "A6", "rentabilidade": 3200, "impacto": 2},
    {"zona": "A7", "rentabilidade": 3400, "impacto": 1},
    {"zona": "A8", "rentabilidade": 4300, "impacto": 2},
    {"zona": "A9", "rentabilidade": 4700, "impacto": 4},
    {"zona": "A10", "rentabilidade": 1100, "impacto": 3},
    {"zona": "A11", "rentabilidade": 4600, "impacto": 4},
    {"zona": "A12", "rentabilidade": 4700, "impacto": 4},
    {"zona": "A13", "rentabilidade": 1700, "impacto": 1},
    {"zona": "A14", "rentabilidade": 5000, "impacto": 4},
    {"zona": "A15", "rentabilidade": 1900, "impacto": 1},
    {"zona": "A16", "rentabilidade": 2900, "impacto": 1},
    {"zona": "A17", "rentabilidade": 1300, "impacto": 4},
    {"zona": "A18", "rentabilidade": 3300, "impacto": 1},
    {"zona": "A19", "rentabilidade": 3000, "impacto": 1},
    {"zona": "A20", "rentabilidade": 4200, "impacto": 1},
    {"zona": "A21", "rentabilidade": 4200, "impacto": 4},
    {"zona": "A22", "rentabilidade": 2400, "impacto": 1},
    {"zona": "A23", "rentabilidade": 4600, "impacto": 3},
    {"zona": "A24", "rentabilidade": 1800, "impacto": 1},
    {"zona": "A25", "rentabilidade": 4500, "impacto": 4},
    {"zona": "A26", "rentabilidade": 2000, "impacto": 2},
    {"zona": "A27", "rentabilidade": 1700, "impacto": 1},
    {"zona": "A28", "rentabilidade": 1700, "impacto": 3},
    {"zona": "A29", "rentabilidade": 1500, "impacto": 2},
    {"zona": "A30", "rentabilidade": 1700, "impacto": 3}
]
✓ 0.0s
```

Imagem 1 - Lista de dicionários com a informação de cada zona

De seguida passámos à definição das funções utilizadas.

1. **Função de avaliação da solução (f(solucao))**: Esta função calcula a soma das rentabilidades esperadas das zonas que compõem a solução. Recebe uma lista de zonas (cada uma representada como um dicionário) e retorna a soma dos valores de rentabilidade.

```python
def f(solucao):
    return sum(zona["rentabilidade"] for zona in solucao)
```

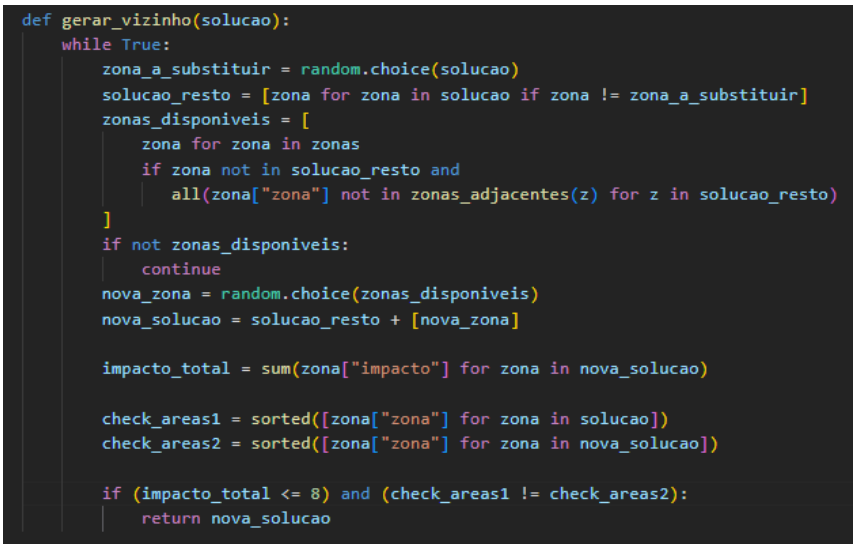
Imagem 2 - Função para calcular o total da rentabilidade total esperada

2. **Função para obter as zonas adjacentes (zonas_adjacentes(zona)):** Esta função retorna as zonas adjacentes a uma dada zona, utilizando um dicionário pré-definido que mapeia cada zona para as suas zonas adjacentes. É essencial para garantir que as novas zonas escolhidas não sejam adjacentes às zonas já selecionadas, conforme estabelecido nas restrições do problema.

```python
def zonas_adjacentes(zona):
    adj = {
        "A1": ["A2", "A7"],
        "A2": ["A1", "A8", "A3", "A4"],
        "A3": ["A2", "A4", "A5"],
        "A4": ["A2", "A3", "A5", "A9", "A10"],
        "A5": ["A3", "A4", "A11", "A6"],
        "A6": ["A5", "A12"],
        "A7": ["A1", "A8", "A13"],
        "A8": ["A7", "A2", "A9", "A13", "A14"],
        "A9": ["A8", "A4", "A10", "A14", "A15"],
        "A10": ["A9", "A4", "A11", "A15"],
        "A11": ["A10", "A5", "A12", "A16"],
        "A12": ["A17", "A11", "A6"],
        "A13": ["A18", "A14", "A8", "A7"],
        "A14": ["A13", "A19", "A15", "A8", "A9"],
        "A15": ["A14", "A21", "A16", "A9", "A10"],
        "A16": ["A15", "A17", "A11", "A23"],
        "A17": ["A16", "A12", "A25"],
        "A18": ["A13", "A19", "A20", "A26"],
        "A19": ["A14", "A18", "A20", "A21"],
        "A20": ["A18", "A19", "A22", "A27"],
        "A21": ["A19", "A15", "A23", "A22"],
        "A22": ["A20", "A21", "A24", "A28"],
        "A23": ["A21", "A16", "A25", "A24"],
        "A24": ["A22", "A23", "A25", "A29"],
        "A25": ["A17", "A23", "A24", "A30"],
        "A26": ["A18", "A27"],
        "A27": ["A26", "A20", "A28"],
        "A28": ["A27", "A22", "A29"],
        "A29": ["A28", "A24", "A30"],
        "A30": ["A29", "A25"]
    }
    return adj.get(zona["zona"], [])
```

Imagem 3 - Função para devolver as zonas adjacentes

3. **Função para gerar um vizinho (gerar_vizinho(solucao)):** Esta função gera uma nova solução vizinha da solução atual. Ela faz isso através da escolha aleatória de uma zona da solução atual para substituir e depois através da seleção de uma nova zona que não esteja na solução atual e que não seja adjacente a nenhuma das zonas da solução. A nova zona deve também respeitar a restrição de impacto ambiental. O processo é repetido até se encontrar uma solução admissível e diferente da que foi passada como parâmetro.



```python
def gerar_vizinho(solucao):
    while True:
        zona_a_substituir = random.choice(solucao)
        solucao_resto = [zona for zona in solucao if zona != zona_a_substituir]
        zonas_disponiveis = [
            zona for zona in zonas
            if zona not in solucao_resto and
               all(zona["zona"] not in zonas_adjacentes(z) for z in solucao_resto)
        ]
        if not zonas_disponiveis:
            continue
        nova_zona = random.choice(zonas_disponiveis)
        nova_solucao = solucao_resto + [nova_zona]

        impacto_total = sum(zona["impacto"] for zona in nova_solucao)

        check_areas1 = sorted([zona["zona"] for zona in solucao])
        check_areas2 = sorted([zona["zona"] for zona in nova_solucao])

        if (impacto_total <= 8) and (check_areas1 != check_areas2):
            return nova_solucao
```

Imagem 4 - Função para gerar soluções vizinhas

4. **Função do Simulated Annealing (simulated_annealing(sol_inicial)):** Esta função implementa o algoritmo do simulated annealing, que começa por calcular a temperatura inicial t0 e gera uma lista de temperaturas T decrescentes. A solução atual e a melhor solução são inicialmente a solução inicial fornecida. Para cada temperatura T[k], a função itera mk vezes (limitado a 5 iterações, conforme definido no enunciado) para gerar uma solução vizinha e calcular a sua rentabilidade esperada. Se a nova solução for melhor ou se a condição probabilística (movimento não melhorativo) for satisfeita, a solução atual é atualizada. A função retorna a melhor solução encontrada, o respectivo valor de rentabilidade esperada dessa solução e um data

frame com todas iterações do algoritmo que gravamos posteriormente em formato txt para analisar o seu comportamento.

```python
def simulated_annealing(solucao_inicial):
    t0 = 0.2 * f(solucao_inicial)
    T = [t0 * (0.5 ** k) for k in range(5)]
    mk = 5

    solucao_atual = solucao_inicial
    melhor_solucao = solucao_inicial
    melhor_rentabilidade = f(solucao_inicial)

    all_sols = pd.DataFrame(columns=['iteracao', 'temperatura', 'localizacao', 'rentabilidade',
                                     'impacto ambiental'])

    for k in range(5):
        for i in range(mk):
            solucao_vizinha = gerar_vizinho(solucao_atual)
            rentabilidade_vizinho = f(solucao_vizinha)
            delta = f(solucao_atual) - rentabilidade_vizinho # por querermos maximizar a rentabilidade esperada

            probabilidade_aceitacao = math.exp(-delta / T[k]) if delta > 0 else 1
            aceitar = delta < 0 or probabilidade_aceitacao > random.random()

            if aceitar:
                solucao_atual = solucao_vizinha
                if rentabilidade_vizinho > melhor_rentabilidade:
                    melhor_solucao = solucao_vizinha
                    melhor_rentabilidade = rentabilidade_vizinho

            # guardar as iterações do algoritmo
            new_row = pd.DataFrame([{
                'iteracao': k * mk + i,
                'temperatura': T[k],
                'localizacao': [zona["zona"] for zona in solucao_vizinha],
                'rentabilidade': f(solucao_vizinha),
                'impacto ambiental': sum(zona["impacto"] for zona in solucao_vizinha)
            }])
            all_sols = pd.concat([all_sols, new_row], ignore_index=True)

    return melhor_solucao, melhor_rentabilidade, all_sols
```

Imagem 5 - Função do simulated annealing

Para testar, inicializamos o algoritmo de simulated annealing com a solução inicial admissível, obtida na alínea c), composta pelas zonas [A1, A30, A8, A20], que corresponde aos índices [0, 29, 7, 19] da lista de dicionários chamada zonas. Note-se que os índices em Python começam em 0 daí a lista de índices ser igual ao número da zona - 1. No exemplo da imagem 6, a melhor solução encontrada pelo algoritmo após mk iterações foi de considerar fazer o corte das árvores nas zonas A20, A24, A8 e A12.

O resultado mostra que o algoritmo conseguiu encontrar uma solução admissível dentro das restrições dadas, maximizando a rentabilidade esperada dentro do número limitado de iterações. Contudo, importa frisar que devido ao baixo número de iterações, não é garantido que esta seja a solução ótima global, já que apenas algumas soluções vizinhas foram exploradas. Para aumentar a probabilidade de encontrar a solução ótima, deveríamos experimentar outras configurações dos parâmetros de arrefecimento (tk, mk), para garantir que exploramos convenientemente a vizinhança.

```
melhor_sol, melhor_val, all_sols = simulated_annealing([zonas[0], zonas[29], zonas[7], zonas[19]])

print('A melhor solução admissível obtida ao fim das mk iterações é:')
for zona in melhor_sol:
    print(json.dumps(zona, indent=4))

print(f"\nA maior rentabilidade esperada obtida ao fim de mk iterações é {melhor_val} €")
```

✓ 0.0s

```
A melhor solução admissível obtida ao fim das mk iterações é:
{
    "zona": "A20",
    "rentabilidade": 4200,
    "impacto": 1
}
{
    "zona": "A24",
    "rentabilidade": 1800,
    "impacto": 1
}
{
    "zona": "A8",
    "rentabilidade": 4300,
    "impacto": 2
}
{
    "zona": "A12",
    "rentabilidade": 4700,
    "impacto": 4
}

A maior rentabilidade esperada obtida ao fim de mk iterações é 15000 €
```

Imagem 6 - Aplicação do simulated annealing com a solução admissível da alínea c)