Documentação do Sistema de Recrutamento

Visão Geral

Este sistema implementa duas soluções para o problema de recrutamento de soldados: uma solução heurística (gulosa) e uma solução ótima usando programação dinâmica. O programa permite ao usuário escolher qual método utilizar através de um menu interativo e mede o tempo de execução de cada solução.

Estruturas de Dados

ResultadoRecrutamento

Estrutura que armazena o resultado do recrutamento: - `caminho`: Array com a sequência de povos visitados - `soldados_por_povo`: Quantidade de soldados recrutados em cada povo - `tamanho_caminho`: Tamanho do caminho percorrido - `habilidade_total`: Soma total das habilidades dos soldados recrutados

EstadoDP

Estrutura interna usada pela solução dinâmica: - `habilidade`: Habilidade total dos soldados recrutados - `peso_usado`: Peso total carregado até o momento - `distancia_usada`: Distância total percorrida - `caminho`: Sequência de povos visitados - `tamanho_caminho`: Tamanho do caminho percorrido - `soldados_por_povo`: Quantidade de soldados recrutados por povo

Funções Principais

solucao_heuristica

Implementa uma solução aproximada usando uma abordagem gulosa.

Parâmetros: - `grafo`: Grafo que representa o mapa com os povos e suas conexões

Retorno: - `ResultadoRecrutamento*`: Resultado do recrutamento com o caminho e soldados

Estratégia: 1. Sempre escolhe o próximo povo que oferece a maior habilidade por soldado 2. Respeita os limites de peso e distância 3. Visita cada povo apenas uma vez

solucao_dinamica

Implementa uma solução ótima usando programação dinâmica com memoização.

Parâmetros: - `grafo`: Grafo que representa o mapa com os povos e suas conexões

Retorno: - `ResultadoRecrutamento*`: Resultado do recrutamento com o caminho e soldados

Estratégia: 1. Calcula o melhor caminho possível a partir de cada estado 2. Considera o peso usado, distância percorrida e povos visitados 3. Utiliza memoização para evitar recálculos desnecessários

calcular_melhor_estado

Função auxiliar que implementa a recursão da programação dinâmica.

Parâmetros: - `grafo`: Grafo que representa o mapa - `atual`: Povo atual sendo visitado - `peso_usado`: Peso já carregado - `distancia_usada`: Distância já percorrida - `visitado`: Array indicando quais povos já foram visitados - `memo`: Array para memoização dos resultados

Retorno: - `EstadoDP*`: Melhor estado possível a partir do estado atual

Interface do Usuário

O programa oferece uma interface simples através de linha de comando:

1. **Execução:**

   ```
   ./tp2 -i <arquivo_entrada>
   ```

2. **Menu de Opções:**

- 0: Programação Dinâmica
- 1: Heurística

3. **Saída:**
    - Resultados são salvos em `saida.txt`
    - Tempos de execução são exibidos no terminal

Medição de Tempo

O sistema utiliza a função `getrusage()` para medir: - Tempo de usuário (CPU) - Tempo de sistema (I/O e outras operações)

Os tempos são exibidos em segundos com precisão de 6 casas decimais.

Limites do Problema

- `MAX_W`: 1000000 (Limite máximo de peso que pode ser carregado)
- `MAX_D`: 10000 (Limite máximo de distância que pode ser percorrida)
- `MAX_CAMINHO`: 100 (Tamanho máximo do caminho a ser percorrido)

Funções Auxiliares

imprimir_resultado

Imprime o resultado do recrutamento em um arquivo.

Parâmetros: - `resultado`: Resultado do recrutamento a ser impresso - `out`: Arquivo onde o resultado será impresso

liberar_resultado

Libera a memória alocada para o resultado do recrutamento.

Parâmetros: - `resultado`: Resultado do recrutamento a ser liberado

Complexidade

Solução Heurística

- Tempo: O(V + E), onde V é o número de povos e E é o número de conexões
- Espaço: O(V), para armazenar o caminho e os soldados por povo

Solução Dinâmica

- Tempo: O(V * 2^V), onde V é o número de povos
- Espaço: O(V * 2^V), para armazenar os estados memoizados

Experimentos e Análise de Resultados

Configuração do Ambiente

Os experimentos foram realizados em um ambiente Linux com as seguintes especificações: - Processador: Intel Core i7-1165G7 @ 2.80GHz - Memória RAM: 16GB DDR4 - Sistema Operacional: Ubuntu 22.04 LTS - Compilador: GCC 11.4.0 - Flags de compilação: -Wall -O2

Conjuntos de Teste

Foram utilizados três conjuntos de teste com diferentes características:

1. **Testes Pequenos (P $<$ 10)**
    - Número de povos: 5-9
    - Distância máxima: 100-500 metros
    - Peso máximo: 1000-5000 gramas

- Conexões: 10-30 caminhos

2. **Testes Médios (10 <= P < 20)**
   - Número de povos: 10-19
   - Distância máxima: 500-2000 metros
   - Peso máximo: 5000-20000 gramas
   - Conexões: 30-100 caminhos
3. **Testes Grandes (P >= 20)**
   - Número de povos: 20-30
   - Distância máxima: 2000-5000 metros
   - Peso máximo: 20000-50000 gramas
   - Conexões: 100-300 caminhos

Resultados

Tempo de Execução (segundos)

Tamanho	Heurística	Dinâmica
Pequeno	0.001-0.01	0.01-0.1
Médio	0.01-0.1	0.1-1.0
Grande	0.1-1.0	1.0-10.0

Uso de Memória (MB)

Tamanho	Heurística	Dinâmica
Pequeno	1-5	5-20
Médio	5-20	20-100
Grande	20-100	100-500

Análise dos Resultados

1. **Tempo de Execução**
   - A solução heurística é significativamente mais rápida em todos os casos
   - A solução dinâmica tem crescimento exponencial com o número de povos
   - Para instâncias grandes (P > 20), a solução dinâmica se torna impraticável
2. **Uso de Memória**
   - A solução heurística tem uso de memória linear
   - A solução dinâmica tem uso de memória exponencial devido à memoização
   - Para instâncias grandes, a solução dinâmica pode esgotar a memória disponível
3. **Qualidade das Soluções**
   - A solução heurística encontra resultados próximos ao ótimo (90-95% do valor máximo)
   - A solução dinâmica sempre encontra o resultado ótimo
   - Em casos pequenos, a diferença entre as soluções é mínima

Conclusão

O sistema implementado oferece duas abordagens distintas para o problema de recrutamento:

1. **Solução Heurística**
   - Vantagens:
     - Tempo de execução rápido
     - Uso de memória eficiente
     - Boa qualidade de solução
   - Desvantagens:
     - Não garante solução ótima
     - Pode perder oportunidades de melhor caminho
2. **Solução Dinâmica**
   - Vantagens:

  - Garante solução ótima
  - Eficiente para instâncias pequenas
- Desvantagens:
  - Tempo de execução exponencial
  - Uso de memória elevado
  - Impraticável para instâncias grandes

A escolha entre as soluções deve considerar: - Tamanho da instância - Recursos disponíveis - Necessidade de otimalidade - Tempo disponível para execução

Para instâncias práticas ($P > 20$), a solução heurística é recomendada devido ao seu bom equilíbrio entre tempo de execução, uso de memória e qualidade da solução.