Pseudocódigo e explicação

Explicação do Problema

No problema UVA 10039 – Railroads, é necessário determinar a conexão mais rápida entre duas cidades, usando os horários de trem disponíveis. Para cada caso de teste, o input fornece:

- **Lista de Cidades:**

Uma lista com os nomes das cidades.

- **Horários de Trem (Schedules):**

Cada schedule contém um número de paradas. Para cada parada é informado o horário (no formato HHMM) e o nome da cidade em que o trem para.

De cada schedule, são extraídas as conexões entre paradas consecutivas, desde que o horário de chegada seja igual ou superior ao de partida.

- **Horário de Partida, Origem e Destino:**

Após os schedules, é informado o horário de partida, a cidade de origem e a cidade de destino.

O objetivo é encontrar, a partir do horário de partida, a rota que permita chegar ao destino o mais cedo possível. Se não houver conexão possível, a saída deve indicar “No connection”.

No caso de haver uma conexão, a saída deve mostrar:

- A hora e a cidade de onde o primeiro trem foi embarcado (Departure).
- A hora e a cidade onde se chega ao destino (Arrival).

Explicação da Solução

A solução utiliza os seguintes passos:

1. **Leitura e Processamento do Input:**

O programa lê o arquivo **input.txt** e extrai, para cada cenário, os nomes das cidades, os schedules (horários de trem) e os dados de partida (horário, cidade de origem e destino).

2. **Construção das Conexões:**

Para cada schedule, são criadas conexões (arestas) entre cada par de paradas consecutivas. Cada conexão é representada por uma tupla:

(horário de partida, horário de chegada, cidade destino)

Essas conexões são armazenadas em um dicionário que mapeia cada cidade para suas conexões de saída.

3. **Busca pela Conexão mais Rápida:**

Utiliza-se uma abordagem inspirada no algoritmo de Dijkstra, onde cada estado é definido por:

- A cidade atual.
- O horário atual (em minutos).
- O horário da primeira partida tomada (quando se sair da cidade de origem).

O estado inicial é definido com o horário de partida fornecido e a cidade de origem, sem ter ainda embarcado num trem (primeiro horário é None). Para cada conexão que pode ser tomada (aquela cujo horário de partida é maior ou igual ao horário atual), o algoritmo atualiza o melhor horário de chegada para a cidade destino, mantendo o horário da primeira partida (caso ainda não tenha sido definido, ele passa a ser o horário daquele trem).

Ao final, se o destino foi alcançado, a solução imprime o horário de partida do primeiro trem (formatado no padrão HHMM) e o horário de chegada no destino, caso contrário, imprime “No connection”.

Pseudocódigo

```
Ler T (número de cenários)

Para cada cenário:
    Ler n (número de cidades)
```

```
Ler os nomes das cidades e armazená-los em uma lista

Ler m (número de schedules)
Para cada schedule:
    Ler k (número de paradas)
    Para cada parada:
        Ler o horário (HHMM) e o nome da cidade
        Converter o horário para minutos
    Adicionar o schedule à lista de schedules

Ler start_time, source e destination

Para cada cidade, inicializar outgoing[city] como lista vazia
Para cada schedule:
    Para cada par de paradas consecutivas:
        Se (horário_chegada >= horário_partida):
            Adicionar à lista outgoing[origem] a conexão (horário_partida, horário

Inicializar o estado inicial: (start_time, source, None)
best[source] = (start_time, None)

Enquanto houver estados para processar (usando heap):
    Extrair o estado com menor horário atual
    Se a cidade for destination, encerrar a busca
    Para cada conexão saindo da cidade atual:
        Se o horário de partida da conexão >= horário atual:
            Definir new_first_dep = (estado.first_dep se definido, senão o horário
            Se (horário de chegada da conexão) for melhor que best[destino]:
                Atualizar best[destino] e adicionar novo estado ao heap

Se best[destination] não foi atualizado (infinito):
    Imprimir "No connection"
Senão:
    Formatar e imprimir:
    "Scenario X"
    "Departure <first_dep_formatado> <source>"
```

```
        "Arrival <arrival_time_formatado> <destination>"

Fim
```

Referências

- https://github.com/morris821028/UVa/blob/master/volume100/10039 - Railroads.cpp
- https://github.com/shahed-shd/Online-Judge-Solutions/blob/master/UVa-Online-Judge/10039 - Railroads.cpp
- https://mypaper.pchome.com.tw/zerojudge/post/1324398988