TRABALHO 1:
BUSCA COMBINATORIAL
DRAFT 0

RAUL H.C. LOPES

1. Preliminares

Neste trabalho você exercitará o seu conhecimento de busca em árvores combinatoriais de pesquisa. O objetivo consistirá em implementar uma heurística para melhorar um **tour** solução de um **TSP**.

Este trabalho é estritamente individual.

2. O Problema do Caixeiro Viajante

Os exercícios deste trabalho lidam com o *Problema do Caixeiro Viajante*, *Traveling Salesman Problem*, de agora em diante chamado **TSP**. Este problema demanda encontrar o caminho mais curto para visitar cada cidade de um conjunto dado exatamente uma vez e retornar ao ponto de partida. Para representar cidades, usaremos uma abstração: um grafo não dirigido.

Um grafo não dirigido é um par (V, E), onde V é um conjunto de pontos e E é um conjunto de pares de pontos, cada par de pontos sendo chamado de arestas. Para nosso trabalho valem as seguintes restrições:

- uma aresta (u, v) indica que existe uma conexão direta do ponto u ao ponto v;
- o grafo é geométrico: os pontos são elementos de um espaço Euclideano de duas dimensões;
- as coordenadas dos pontos são pares de inteiros positivos;
- para qualquer par de pontos u e v do grafo existe uma aresta (u, v), que tem um custo associado dado pela distância entre u e v;
- o grafo é não direcionado e a existência de uma aresta (u, v) indica a existência de outra aresta (v, u) com mesmo custo que a primeira.

Um caminho em um grafo é uma seqüência de pontos do mesmo. Um circuito Hamiltoniano C é uma seqüência de pontos com as seguintes restrições:

- cada ponto do grafo inicial figura exatamente uma vez no circuito;
- o circuito indica a seqüência de pontos a visitar para sair de um vértice inicial, o primeiro da seqüência dado, ir até o último vértice da seqüência, visitando todos os outros na ordem dada por C, e voltar ao ponto de partida.

Um circuito Hamiltoniano, que chamaremos simplemente de **tour**, consiste em uma permutação do conjunto de pontos de grafo dado. Assumindo que $C!i$ denote o i-ésimo elemento do tour C, o custo do tour é dado por:

$$cost.C = (+i : 0 \leq i < n-1 : dist(C!i, C!(i+1))) + dist(C!(n-1), C!0)$$

onde $dist(u, v)$ é a distância geométrica de u a v.

O problema do **TSP**, como colocado neste trabalho, consiste, então, em determinar o **tour** de menor custo.

A heurística *k-opt* usa como passo básico a retirada de k arestas não consecutivas de um **tour** previamente construído e inserção de k novas arestas, produzindo um novo **tour** de peso menor do que o inicial. Esse passo é repetido até explorar todas as alternativas de remoção de k arestas que possam melhorar o **tour**. Note que:

- o valor de k permanece fixo durante uma execução completa de *k-opt*.
- um conjunto de arestas só é removido para a inserção de novas arestas se o novo **tour** for melhor.

O artigo descreve a heurística *k-opt* e apresenta sugestões de estruturas para implementá-la de forma eficiente.

3. Os exercícios

Exercício 1. *Implemente em Haskell um programa para cálculo de* ***tour*** *do* ***TSP*** *que use a heurística* k-opt*.*

Exercício 2. *Implemente em Haskell, usando* monads *ou* threads *para sincronizar a busca na árvore, um programa para cálculo de* ***tour*** *do* ***TSP*** *que use a heurística* k-opt*.*

Exercício 3. *Apresente testes comparativos sobre:*

- *melhora da qualidade de* ***tour*** *obtida;*
- *eficiência comparada das duas heurísticas.*

4. Definição do número de créditos

O número de créditos atribuídos a cada exercício está definido na tab. 1. Note que os créditos atribuídos aos exercícios de teste de desempenho

Tarefa	Valor
ex. 1	40
ex. 2	20
ex. 3	20
desempenho	20

TABELA 1. Tabela de Créditos por Tarefa

e correção das diferentes estruturas dependerão da apresentação de artigo em LaTeX, apresentando tabelas comparativas e conclusões dos experimentos realizados por cada grupo. Não hesite em discutir com outros colegas (de outros grupos) o tipo de teste a realizar. Não será considerado plágio mais de um grupo apresentar extamente os mesmos testes e até estrutura de artigo. Será certamente plágio coincidência em excesso nos resultados de testes! Em resumo, dê desde o início do trabalho ênfase especial à escrita do artigo.

O último item da tabela refere-se à qualidade que será aferida usando os seguintes princípios:

- cada aluno indicará no respectivo artigo o programa P em que quer ser avaliado;
- P receberá um crédito de qualidade em uma série de testes definidos pelo **Senhor do Castelo**;
- o crédito de qualidade de um teste será dado pela divisão do tempo gasto no teste pelo valor da melhora da qualidade do **tour** obtido;
- P receberá como valor final de qualidade a média dos créditos de qualidade obtidos;
- o aluno melhor classificado receberá 20 créditos de desempenho; os outros receberão créditos proporcionais à razão da qualidade respectiva pela qualidade do melhor trabalho.

5. A submissão para correção

Submeta seu trabalho por e-mail entre os dias 05/08/05 e/ou 06/08/05. Trabalhos submetidos fora dessa data serão considerados nulos.

A mensagem de submissão do seu trabalho deverá conter:

- **Subject: lpii.tp1**
- *Attachment*: **tp1.tar.bz2**

O corpo da mensagem será desprezado. O arquivo anexado deverá ser obrigatoriamente denominado

tp1.tar.bz2

e conterá todo os fontes necessários para compilar seu trabalho prático, incluindo fontes em LaTeX, **Haskell**, **Makefile**e arquivo de identificação.

As seguintes regras devem ser rigorosamente seguidas:

- Nenhum compilado deve ser enviado.
- O *tarball* do seu trabalho conterá a seguinte estrutura:
  - arquivo de **id**
    Conterá uma linha inicial com *e-mail* de contato do grupo e, depois, uma linha de identificação para elemento do grupo, contendo número de matrícula e nome completo separados por '**:**' (dois pontos.) Por exemplo:
    ```
    jolero@gmail.com
    99900089:Ze' do lero
    ```
  - arquivo **Makefile**
  - diretório **doc**
    Conterá os fontes de toda a documentação do trabalho.
- seu *tarball* será aberto e compilado com as seguintes linhas:
  ```
  tar -jxf tp0.tar.bz2
  make all
  ```
  Essas linhas gerarão os seguintes arquivos:
  - **kopt**
    Executável que implementa heurística do exercício 1.
  - **monad**
    Executável que implementa heurística do exercício 2.
  - **artigo.pdf**
    Artigo em formato **PDF**, que avalia qualidade de **tour** e desempenho dos algoritmos.