Odhad TSP - Haskell
Zápočtový program na Neprocedurální programování

Tomáš Krejčí <tomas789@gmail.com>

7. června 2013

1 Problém obchodního cestujícího

TSP (Traveling Salesman Problem, česky Problém obchodního cestujícího) je úloha o hledání nejkratší hamiltonovské kružnice v zadaném grahu s ohodnocenými hranami.

Úloha je inspirována problémem, kdy má obchodní cestující objet N měst tak, aby každé navštívil právě jednou a zároveň aby najezdil co nejméně. Převedeno do řeči teorie grafů máme graf $G = (V, E, c)$ takový, že $|V| = N$, $E = \binom{V}{2}$ a $c : E^2 \to \mathbb{R}$. Případně c může zobrazovat do libovolné jiné množiny na níž máme definovány operace $\oplus$ a $\preceq$.

1.1 Speciální případy

Speciálním případem TSP může být například situace, kdy e splňuje trojúhelníkovou nerovnost

$$\forall x, y, z \in V \ : \ c(\{x, z\}) \preceq c(\{x, y\}) \oplus c(\{y, z\})$$

případně c je metrika.

Dalším speciálním případem je situace, kdy $V \subset \mathbb{R}^n$. c poté může být například klasická euklidovská norma.

1.2 Heuristiky

TSP je výpočetně velmi složiný problém, kdy k nalezení optimálního řešení obsahujícího N měst je třeba vyšetřit exponenciálně mnoho měst.

Nalezení nějaké dobré aproximace však může být mnohem snazší úloha. Jednou z prvních možností je *nearest-neighbour algoritmus* který schematicky zapsáno vypadá následovně

1. $P = v \in V$ libovolně
2. **while** $P \neq V$ **begin**
3. označ $v \in V \backslash P$ s nejmenší vzdáleností k P
4. $P = P \cup v$
5. **end**

Další možností jsou takzvané konstrukční heuristiky. Jejich schéma ja takové, že začneme s nějakou podcestou (podmnožina cesty v celém grafu) a postupně ji rozšiřujeme vkládáním nějakého vrcholu na vhodné místo dokud nemáme celou cestu. Tyto heuristiky se liší v několika významných aspektech.

- Jak vybrat počáteční podcestu
- Jak zvolit vrchol, který budeme vkládat
- Kam zvolený vrchol do cesty vložit

Počáteční podcesta je obvykle nějaké množina tří libovolně volených vrcholů nebo ve speciálním případě *Euklidovského TSP* je dobrou volbou začít napčíklad s *konvexním obalem* všech bodů, protože posloupnost vrcholů z konvexního obalu je vždy obsažena v optimálním řešení.

Zvolený bod je obvykle vkládán na pozici v podcestě kde bude míc co nejmenší příspěvek na jejím prodloužení.

Možností jak vybrat bod, který se bude vkládat do cesty je několik. Uvedu zde dva.

- **Nejlevnější vkládání**: Ze všech bodů vybereme ten, který způsobí nejmenší prodloužení cesty. Tento přístup je inspirovaný hladovými algoritmy.

- **Nejvzdálenější vložení**: Vložíme bod, jehož nejmenší vzdálenost od podcesty je největší. Tímto přístupem se snažíme v již pro malé cesty co do počtu vrcholů přiblížit celkovému tvaru cesty.

2 Zadání

Cílem práce je implementovat nějakou vhodnou heuristiku TSP v programovacím jazyce Haskell.

3 Implementace

Program implementuje konstruktivní heuristiku nazvanou *farthest insert* a jako počáteční množinu vrcholů vybírá koncexní obal množiny všech vrcholů. Vrcholy jsou implementovány jako body v $\mathbb{R}^2$ a jejich vzdálenost určuje uživatelem definovaná funkce.

Uveďme nějaké základní typové konstruktory

```
data Point a = P a a deriving (Show,Eq)
data Tour a = T [Point a] deriving (Show)
distance :: Floating a => Point a -> Point a -> a
```

kde distance je funkce určující vzdálenosti mezi dvěma body.

4 Použití

Výsledný program přijímá na vstupu seznam bodů takových, že na každém řádku je jeden bod P. Formát řádku je "$P(x)$ $P(y)$". Výstup je ve stejném formátu seřazený tak, jak jdou jednotlivé body zasebou ve výsledné cestě.