# Úvod

Cílem projektu je v programovacím jazyce C++ vytvořit konzolovou aplikaci, která implementuje heuristiky pro řešení problému obchodního cestujícího. Výsledky jednotlivých heuristik jsou následně porovnány mezi sebou z hlediska časové a prostorové složitosti. Pro instance problému s nízkým počtem vrcholů vstupního grafu je také provedeno porovnání kvality výsledné cesty s přesnou verzí algoritmu.

## Problém obchodního cestujícího

Problém obchodního cestujícího je matematicky formulován takto: V daném ohodnoceném úplném [grafu](https://cs.wikipedia.org/wiki/Graf_(teorie_grafů)) najděte nejkratší [hamiltonovskou kružnici](https://cs.wikipedia.org/wiki/Hamiltonovská_kružnice). To znamená, že na vstupu je graf, který obsahuje n vrchlolů a všechny vrcholy jsou propojeny se všemi ostatnímy. Graf tedy obsahuje ((n-1)\*n)/2 hran. Všechny tyto hrany musí mít definované ohodnocení. V obecné variantě problému není vyžadováno, aby v grafu platila trojúhelníková nerovnost. Pokud ale platí, mluvíme pak o metrickém problému obchodního cestujícího. Cílem je poté v tomto grafu najít takovou trasu, která prochází všemi vrcholy grafu právě jednou a má co nejkratší ohodnocení.

# Popis algoritmů

Aplikace implementuje tři metody řešení problému obchodního cestujícího. Jedná se o algoritmy Double-tree a k-OPT, ty k řešení používají heuristiku a tedy negarantují nalezení nejlepšího řešení, pouze se mu snaží co nejvíce přiblížit. Třetí algoritmus je brute-froce prohledávání, které garantuje nalezení optimální řešení.

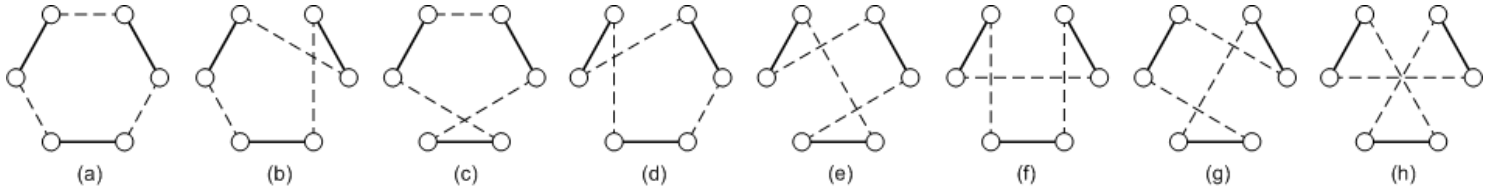
## Brute-force

Algoritmus brute-force pracuje na principu prohledávání celého stavového prostoru. Funguje tedy tak, že postupně vygeneruje a vypočítá metriku všech možných permutací propojení vrcholů. U každé permutace si ověří, jestli není lepší než aktuální nejlepší a pokud ano, tak ji nastaví jako novou nejlepší. Po dokončení algoritmu je tedy zajištěno, že všechny možné trasy byly vyzkoušeny a výsledek je optimální.

## k-OPT

Tato metoda pracuje na principu postupného přepojování k hran za podmínky, že metrika nové trasy s přepojenými hranami je menší a trasa se přitom nerozdělí na více částí.

První krok spočívá ve vytvoření validní počáteční trasy. Na jejím tvaru nezáleží, proto ji zvolíme náhodně, musí ale splňovat to, že se jedná o hamiltonovskou kružnici. Před dalším krokem je nutné připravit si množinu všech validních přepojení k hran.

S touto množinou vypočtenou je možno začít přepojovat hrany. Algoritmus postupně postupuje grafem a generuje všechny unikátní k-tice hran, které se v grafu mohou vyskytnout. Pro každou k-tici se vyzkouší všechny její možné přepojení a vypočítá se jejich metrika. Nyní existuje více možností jak postupovat dále. Některé verze algoritmu tímto způsobem vyzkouší všechny k-tice se všemi možnými propojeními a uloží si tu s nejlepším snížením metriky, tu poté přepojí. Naše verze funguje tak, že po nalezení prvního propojení s nižší metrikou okamžitě přepojíme a cyklus resetujeme. Tohle se opakuje, dokud nedojdeme do stavu, kdy jsme prošli celým grafem a nenašli žádné přepojení, které by vylepšilo metriku. Zbylá trasa je vrácena jako výsledek.

## Double-tree

Doplnit

# Implementace a výpočet teoretických složitostí

Popis implementace obecne jak se to spousti knihovny atd…

V – počet vrcholů

E – počet hran

## Brute-force

Test

Časová složitost:

Při generování všech validních tras nezáleží na rotaci, tedy různé vrcholy startu trasy nemusíme brát v úvahu. Tedy počet permutací = (V-1)!. Protože ale nezáleží ani na směru trasy je toho číslo ještě nutné vydělit dvěma. Počet unikátních permutací v grafu s V vrcholy je výsledku roven ( (V-1)!/2).

Pro každou tuto permutaci je následně nutné vypočítat metriku postupným sečtením metrik jednotlivých hran, což zabere V kroků.

Výsledná časová složitost je tedy O(( (V-1)!/2)\*V)

prostorova asi V+E

## k-OPT

Test

## Double-tree

test

# Měření

obecne

## Způsob měření

Konkretni mereni a vzsledky mereni

## Výsledky

Zavery y mereni

# Zdroje

zformatovat

# http://rtime.felk.cvut.cz/~hanzalek//KO/TSP\_e.pdf

https://cs.wikipedia.org/wiki/Úplný\_graf  
<https://cs.wikipedia.org/wiki/Problém_obchodního_cestujícího>

<https://www.algoritmy.net/article/5407/Obchodni-cestujici>

https://pdfs.semanticscholar.org/ab7c/c83bb513a91b06f6c8bc3b9da7f60cbbaee5.pdf