Univerzita Karlova
Přírodovědecká fakulta

Algoritmy počítačové kartografie

Technická zpráva ke čtvrté semestrální úloze

Množinové operace s polygony

David Šklíba a Filip Zadražil
1. N-GKDPZ
Praha 2022

1 Zadání a údaje o bonusových úlohách

Úloha č. 4: Množinové operace s polygony

Vstup: nekonvexní polygony P, Q,

Výstup: množina k *polygonů* $P' = \{P'_1, ..., P'_k\}$.

S využitím algoritmu pro množinové operace s polygony implementujte pro dvojici polygonů P, Q následující množinové operace:

- průnik polygonů,
- sjednocení polygonů,
- rozdíl polygonů.

Jako vstupní data použijte existující kartografická či syntetická data reprezentující množiny P, Q, která budou načítána ze dvou textových souborů ve Vámi zvoleném formátu.

Grafické rozhraní realizujte s využitím frameworku QT, výsledky množinových operací vizualizujte.

Poznámka: pro výše uvedené kroky je nutné mít řádně odladěny algoritmy z úlohy 1.

Hodnocení:

Krok	**Hodnocení**
Množinové operace: průnik, sjednocení, rozdíl	20b
Ošetření singulárních případů (kolineární segment, sdílený vrchol, atd.)	*+10b*
Konverze výstupů množinových operací na polygony.	*+10b*
Výpočet průsečíků segmentů algoritmem Bentley & Ottman.	*+15b*
Max celkem:	**55b**

V rámci vzniklého programu nebyly řešeny **žádné** bonusové úlohy.

2 Popis a rozbor problému

Při množinových operacích s polygony máme k dispozici dva ohraničené regiony A, B v $\mathbb{R}^2$ a hledáme jejich sjednocení (Union), průnik (Intersection) či rozdíl (Difference). Tuto problematiku lze označit za základní analytický nástroj GIS, neboť takovéto operace jsou pro práci geoinformatika takřka nepostradatelné a tvoří základ všech geoinformačních softwarů.

Matematický zápis těchto tří základních operací matematicky značíme takto:

- Sjednocení: $C = A \cup B$
- Průnik: $C = A \cap B$
- Rozdíl: $C = A \cap \overline{B}$ resp. $C = B \cap \overline{A}$

Princip užitého algoritmu funguje pro jednoduché nekonvexní polygony (včetně otvorů). Máme dva polygony $A = \{p_i\}_{i=1}^{n}, B = \{q_j\}_{j=1}^{m}$, kde A, B jsou reprezentovány kruhovými seznamy (první a poslední bod jsou totožné) a body v nich obsažené mají CCW orientaci.

Samotný algoritmus má několik fází. V následujících podkapitolách si je blíže představíme.

2.1 Výpočet průsečíků

V první části procházíme hrany polygonů A, B a hledáme jejich průsečík(y), které označujeme indexem b_{ij}. Máme tedy dvě hrany e_i, e'_j s krajními body označenými $P_i = [x_i, y_i], P_{i+1} = [x_{i+1}, y_{i+1}]$, resp. $Q_j = [x_j, y_j], Q_{j+1} = [x_{j+1}, y_{j+1}]$. Nyní je nutné vypočítat jejich směrové vektory $\overrightarrow{u}, \overrightarrow{v}, \overrightarrow{w}$:

$$\overrightarrow{u} = (x_{i+1} - x_i, y_{i+1} - y_i), \qquad \overrightarrow{v} = (x_{j+1} - x_j, y_{j+1} - y_j), \qquad \overrightarrow{w} = (x_i - x_j, y_i - y_j).$$

Z jednotlivých složek vektorů následně vypočítáme determinanty k_1, k_2, k_3

$$k_1 = v_x w_y - v_y w_x, \qquad k_2 = u_x w_y - u_y w_x, \qquad k_3 = v_y u_x - v_x u_y.$$

Poté pomocí determinantů vypočítáme parametry α, β, které určují polohu průsečíku v rámci oněch dvou hran

$$\alpha = \frac{k_1}{k_3}, \qquad \beta = \frac{k_2}{k_3}.$$

V případě, že:

- $k_1 = 0, k_2 = 0 a k_3 = 0$: hrany e_i, e'_j jsou kolineární, nemají společný průsečík,
- $k_1 = 0, k_2 = 0$: hrany e_i, e'_j jsou rovnoběžné, nemají společný průsečík,
- $\alpha \in \langle 0, 1\rangle$ a $\beta \in \langle 0, 1\rangle$: průsečík existuje.

Ve všech jiných případech (mimo výše uvedené) jsou hrany mimoběžné.

2.2 Update seznamů A, B

Vzhledem k tomu, že polygonu B může protínat hranu e_i více hranami, než jen jednou z nich, je nutné testovat každou hranu polygonu A vůči všem hranám polygonu B.

V případě, že nalezneme průsečík b_{ij}, můžeme průběžně automaticky přidávat tento bod do polygonu B mezi body q_j a q_{j+1}. Pro polygon A ukládáme průsečíky b_{ij} do seznamu bodů $M(\alpha)$. Po nalezení všech průsečíků b_{ij} hromadně updatujeme polygon A. Seznam bodů $M(\alpha)$ poté přidáme mezi body p_j a p_{j+1}.

2.3 Ohodnocení vrcholů A, B

V této části nejprve musíme určit polohu hrany e_i vůči B pomocí libovolného vnitřního bodu $\overline{p}_i$ a analogicky polohu hrany e'_j vzhledem k A pomocí bodu $\overline{q}_j$. Poté každému bodu $\overline{p}_i \in A$ určíme jeho polohu vzhledem k B a každému bodu $\overline{q}_j \in B$ vzhledem k A. Ohodnocovací funkce pak má tvar

$$g(\overline{p}_i, B) = \begin{cases} \overline{p}_i \notin B, \\ \overline{p}_i \in \delta B. \\ \overline{p}_i \in B \end{cases} \qquad g(\overline{q}_j, A) = \begin{cases} \overline{q}_j \notin A, \\ \overline{q}_j \in \delta A. \\ \overline{q}_j \in A \end{cases}$$

Vzhledem k tomu, že námi zpracovaný algoritmus singularity neošetřuje, rozlišujeme polohu hran pouze vnitřní a venkovní.

2.4 Výběr hran dle ohodnocení

V této fázi dochází k výběru hran polygonů, které mají vůči druhému polygonu určitou pozici. Tento výběr hran závisí na typu prováděné množinové operace. Z příslušných hran jsou následně sestaveny fragmenty. Pozice vybraných hran na základě zvolené množinové operace jsou k vidění v Tabulce 1.

Operace	polygon A	polygon B
$C = A \cap B$	vnitřní	vnitřní
$C = A \cup B$	vnější	vnější
$C = A \cap \overline{B}$	vnější	vnitřní
$C = B \cap \overline{A}$	vnitřní	vnější
$C = A \triangle B$	vnitřní + vnější	vnitřní + vnější

Tabulka 1: Pozice vybraných hran polygonů na základě zvolené množinové operace

2.5 Ukázka pseudokódu

Níže, pod názvem Algorithm 1, se nachází pseudokód pro nalezení sjednocení a průniku dvou polygonů.

Algorithm 1 Sjednocení (Union) a průnik (Intersection)

```
1: Vytvoř prázdný seznam hran e
2: Pro kraždou hranu e_1 polygonu pol_A:
3:     Pro kraždou hranu e_2 polygonu pol_B:
4:         if ∃!P = e_1 ∩ e_2 :
5:             přidej P do pol_A
6:             přidej P do pol_B
7: Pro kraždou hranu e_1 polygonu pol_A:
8:     S = (A+B)/2, kde A a B jsou koncové body hrany
9:     if S ∈ pol_B:
10:        if operace Union:
11:            přidej e_1 do e
12:    else if S ∉ pol_B:
13:        if operace Intersection:
14:            přidej e_1 do e
15: Pro kraždou hranu e_2 polygonu pol_B zopakuj kroky 8-14
```

* pokud provádíme operaci rozdíl polygonů $pol_A - pol_B$, přidáváme hranu polygonu pol_A do seznamu hran, pokud platí podmínka na 12. řádku, hranu polygonu pol_B poté přidáváme, pokud platí podmínka na 9. řádku

3 Vstupní a výstupní data

Vstupními daty jsou dva soubory ve formátu *.txt*, které obsahují souřadnice X, Y všech bodů, z nichž se polygony skládají.

Výstupní data program uživateli v žádném souboru nenabídne. Jedná se pouze o vizualizaci výsledku provedené množinové operace ve widgetu. Po zavření programu výsledek zmizí.

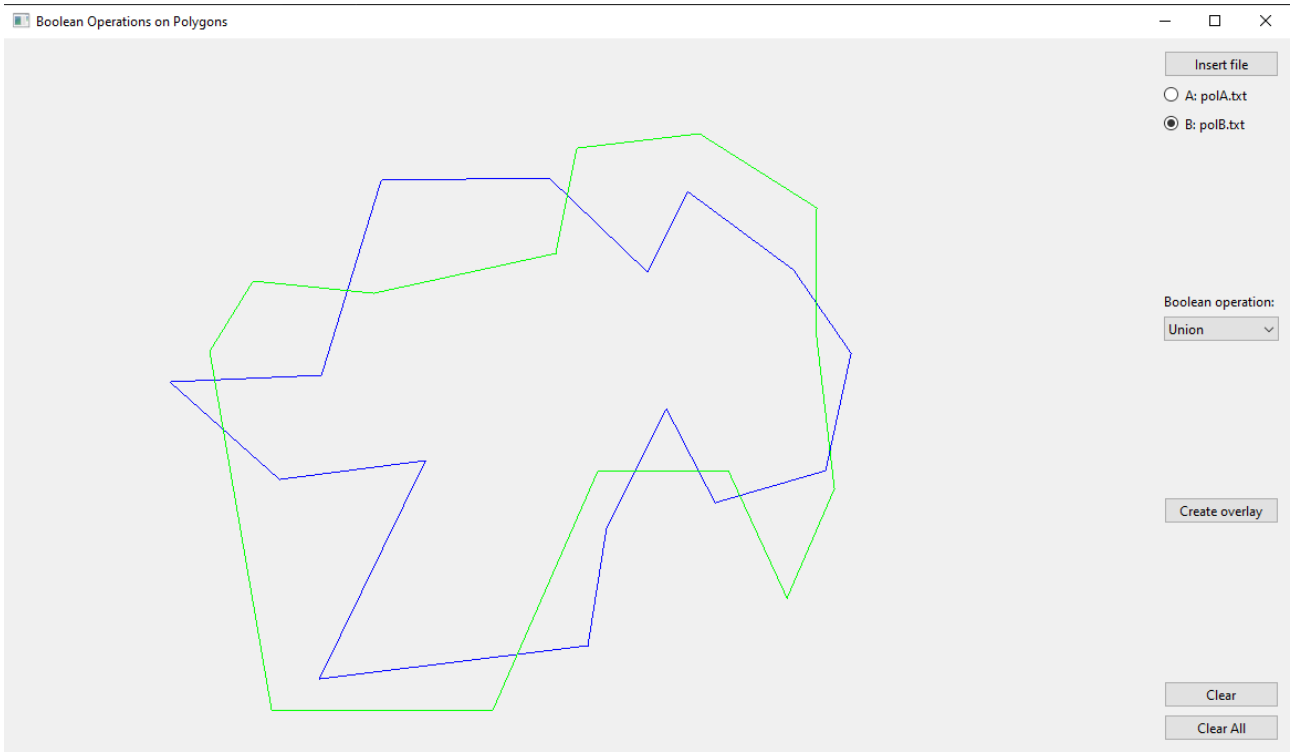
4 Ukázka vytvořené aplikace

Na Obrázku 1 je znázorněn vytvořený widget, který uživateli umožňuje program ovládat a provádět vybrané množinové operace.

V ovládacím panelu po pravé straně nalezneme nahoře tlačítko *Insert file*, které využívá tzv. *radio button*. V závislosti na tom, jaké políčko máme zaškrtnuté (A nebo B), takový v tu chvíli nahráváme polygon. Po nahrání se nám u příslušného písmene objeví název souboru.

Níže v ovládacím panelu nalezneme combo box, který obsahuje 4 množinové operace, které můžeme provádět. Jedná se o Sjednocení, Průnik, Rozdíl A-B a Rozdíl B-A. Spuštění vybrané operace provádíme pomocí tlačítka *Create Overlay*

V nejspodnější části panelu se nacházejí tlačítka *Clear* a *Clear All*. První z nich vymaže výsledek pouze provedené operace a druhá z nich vymaže vše, včetně naimportovaných polygonů.



Obrázek 1: Ukázka vytvořeného widgetu

5 Dokumentace

Skript $MainFormBoolean.py$, přes který se program spouští, obsahuje třídu uživatelského rozhraní $Ui\_mainform$ navrženého pomocí SW $QTCreator$. Pro jednotlivá tlačítka má třída definované metody odkazující na skripty $draw.py$ a $algorithms.py$. Skript $draw.py$ obsahuje třídu $Draw$, která byla navržena pro vizualizaci objektů ve widgetu uživatelského rozhraní.

Parametry třídy $Draw$ jsou:

- seznam $self.polA$ objektů typu $QPointFB$ vstupních bodů prvního polygonu,
- seznam $self.polB$ objektů typu $QPointFB$ vstupních bodů druhého polygonu,
- seznam $self.res$ objektů typu $Edge$ segmentů vrstevnic,

Metodami třídy $Draw$ jsou:

- $insertFile$ - načítá vstupní soubor ve formátu $.txt$, ukládá souřadnice jednotlivých bodů do seznamů $self.polA$ nebo $self.polB$
- $paintEvent$ - vykresluje body, linie a polygony do okna widgetu,
- $getPolygons$ - vrací polygony $self.polA$ a $self.polB$,

- *setResults* - parametru self.res přiřadí seznam hran,
- *clearResults* - vymaže obsah seznamu hran $self.res$,
- *clearCanvas* - vymaže obsah seznamu hran $self.res$ a obsah seznamů bodů polygonů $self.polA$ a $self.polB$,

Skript *algorithms.py* obsahuje třídu *Algorithms*, která byla vytvořena pro implementaci vybraných algoritmů.

Metodami třídy *Algorithms* jsou:

- *getPointAndLinePosition* – určuje pozici bodu vůči přímce,
- *get2LinesAngle* – počítá velikost úhlu svíraného mezi dvěma úsečkami,
- *getPositionPointAndPolygon* - určuje vzájemnou polohu bodu a polygonu metodou winding number,
- *get2LinesIntersection* - zjišťuje souřadnice průsečíku dvou linií, pokud existuje,
- *updateVertices* - přidává průsečíky hran jako nový bod polygonu,
- *setEdgePosition* - stanovuje pozici hrany prvního polygonu vůči polygonu druhému,
- *getEdges* - vrací seznam hran podle jejich pozice,
- *createOverlay* – provádí booleovskou operaci nad dvěma polygony,

Skript *qpointfb.py* obsahuje třídu $QPointFB$ dědící z třídy $QPointF$. Oproti svému rodiči je obohacena o parametry $self.alpha$ a $self.beta$, které vyjadřují hodnoty skalárů. Poslední parametr $self.position$ vyjadřuje prostřednictvím počátečního bodu hrany její polohu vůči jinému polygonu. Metodami třídy jsou:

- *getAlpha* - vrací hodnotu skaláru $self.alpha$,
- *getBeta* - vrací hodnotu skaláru $self.beta$,
- *getPosition* - vrací pozici středového bodu hrany vůči druhému polygonu $self.position$ ve formátu typu PointAndPolygonPosition,
- *setPosition* - nastavuje hodnotu pozice středového bodu hrany vůči druhému polygonu parametru $self.position$.

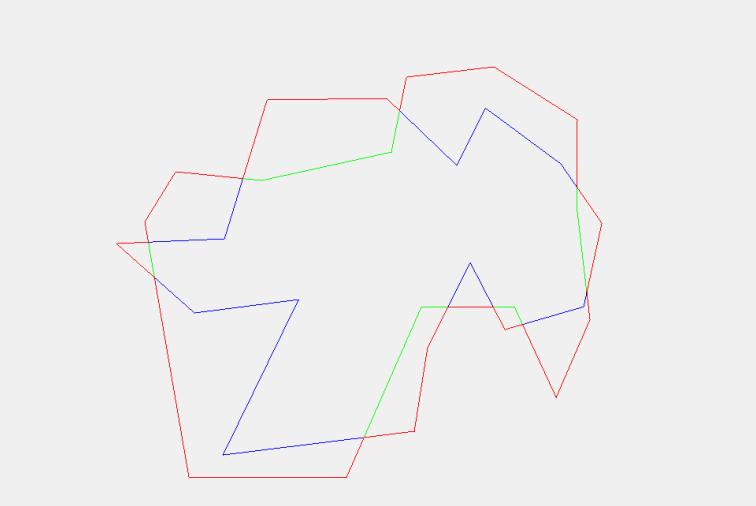
Skript *edge.py* obsahuje třídu Edge reprezentující orientovanou hranu polygonu. Třída přijímá dva vstupní argumenty typu QPointFB, první se ukládá do parametru $self.start$ vyjadřujícího startovní bod hrany, druhý se potom ukládá do paramatru $self.end$ označujícího koncový bod hrany. Metodami třídy *Edge* jsou:

- *getStart* - vrací startovní bod hrany $self.start$,
- *getEnd* - vrací koncový bod hrany $self.end$,

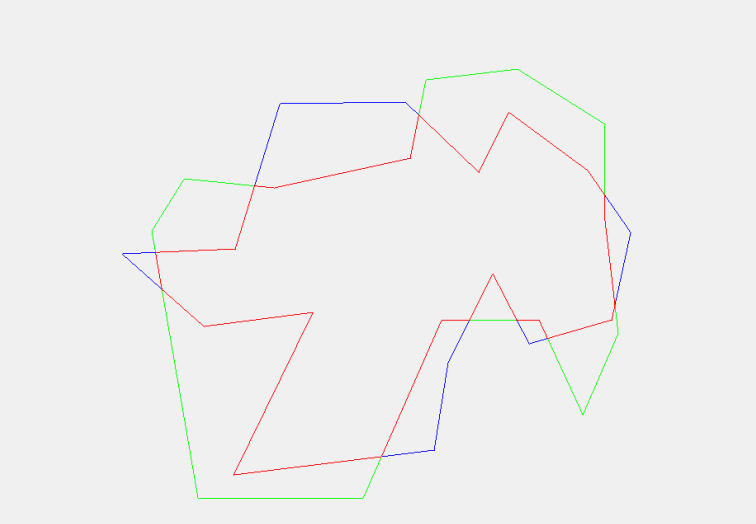
Ostatní třídy programu představují výčtové typy, a to ve skriptech *booleanoperations.py*, *lineandlineposition.py*, *pointandpolygonposition.py* a *pointlineposition.py*.

6 Výsledky

V této části nalezneme na Obrázcích 2-5 ukázky všech čtyř výsledných operací při použití polygonů uložených v přiložených txt souborech $polA$ a $polB$.



Obrázek 2: Union (sjednocení)



Obrázek 3: Intersection (průnik)

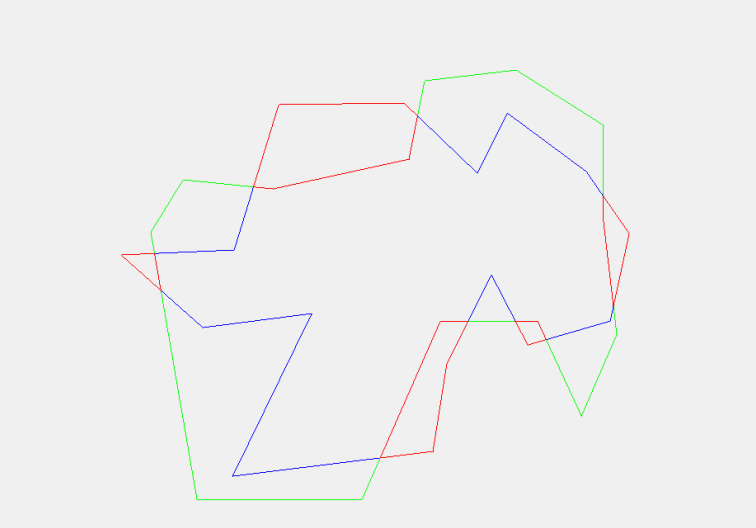
7 Závěr

Program vznikuvší v rámci tohoto úkolu splňuje všechny body stanovené v zadání. Podařilo se naprogramovat algoritmus, který úspěšně provádí čtyři vybrané množinové operace a vizualizuje jejich výsledky ve vytvořeném widgetu. Program je navíc jednoduchý, intuitivní a dá se v něm dobře orientovat.

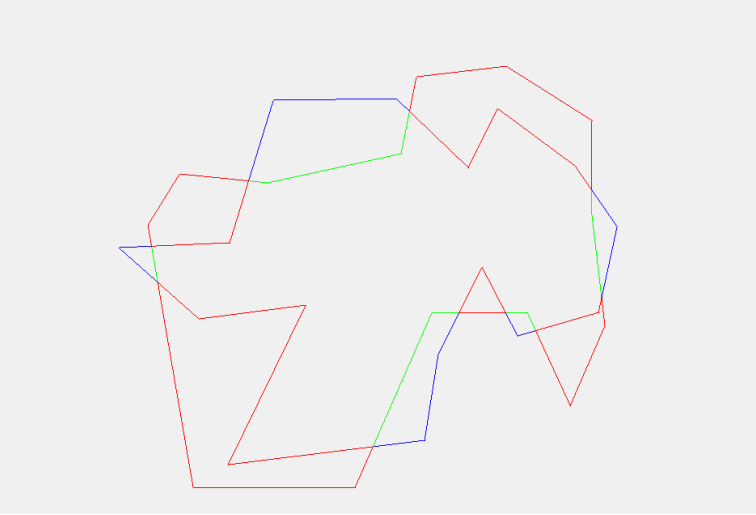
Možné vylepšení programu se ukrývá v ošetření singularit, kterými jsou například kolineární segment, sdílený vrchol atd.

8 Seznam literatury

Zdrojový kód programu vznikl z velké části na základě informací z přednášky a cvičení vedených doc. Bayerem (2022). Informace byly čerpány z prezenčních lekcí a zároveň z prezentace dostupné na odkazu https://web.natur.cuni.cz/ bayertom/images/courses/Adk/adk9.pdf (cit. 28. 5. 2022)



Obrázek 4: Difference A-B (Rozdíl A-B)



Obrázek 5: Difference A-B (Rozdíl B-A)