

Univerzita Karlova

Přírodovědecká fakulta



## Úloha 4 - Množinové operace s polygony

Algoritmy počítačové kartografie

Tomáš Hřebec, Kateřina Obrazová

Praha 2022

# 1 Zadání úlohy

## 1.1 Povinná část

*Vstup:* Nekonvexní polygony  $P, Q$ ,

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

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.

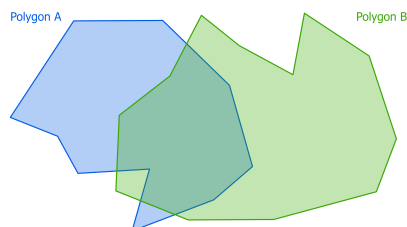
Poznámka: pro výše uvedené kroky je nutné mít řádně odladěny algoritmy z úlohy 1.

## 1.2 Volitelná část

V této úloze nebyly řešené bonusové úlohy.

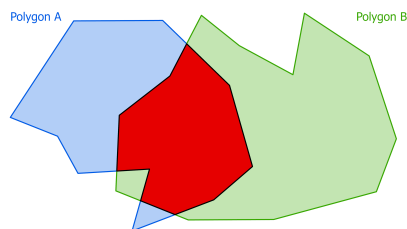
## 2 Formulace problému

Na vstupu je dvojice množin. Tyto množiny jsou uzavřené a ohraničené polygony (oblasti)  $A$  a  $B$ . (Obr. 1)  
Hledáme základní množinové operace mezi nimi - průnik, sjednocení a rozdíl.



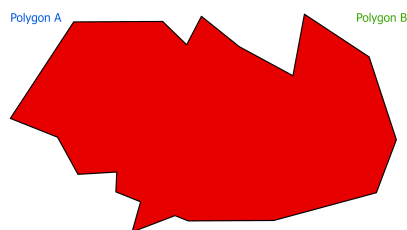
Obr. 1 - Příklad polygonů  $A$  a  $B$

Průnik množin  $A$  a  $B$  (angl. *Intersection*) je množinou všech prvků obsažených v množině  $A$ , a zároveň i v množině  $B$ . (Obr. 2) Značí se jako  $A \cap B$ . (Moravec 2008)



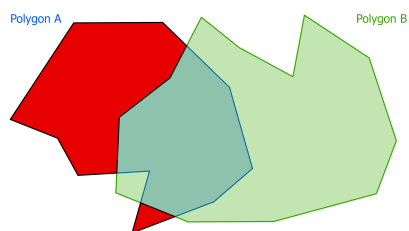
Obr. 2 - Průnik polygonů  $A$  a  $B$

Sjednocení množin  $A$  a  $B$  (angl. *Union*) obsahuje jen a pouze takové prvky, které patří alespoň do jedné z množin  $A$  a  $B$ . (Obr. 3) Značí se jako  $A \cup B$ . (Moravec 2008)

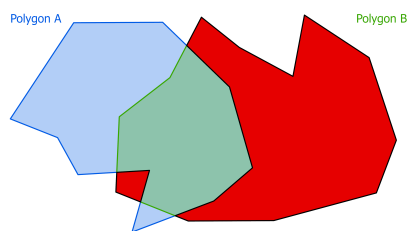


Obr. 3 - Sjednocení polygonů  $A$  a  $B$

Rozdíl množin  $A$  a  $B$ , respektive  $B$  a  $A$  (angl. *Difference*), je množinou obsahující všechny prvky množiny  $A$  (resp.  $B$ ) s výjimkou těch, které jsou zároveň prvky množiny  $B$  (resp.  $A$ ). Značí se jako  $A \cap \overline{B}$  (Obr. 4), respektive  $\overline{A} \cap B$  (Obr. 5). (Moravec 2008)



Obr. 4 - Rozdíl polygonů  $A$  a  $B$



Obr. 5 - Rozdíl polygonů  $B$  a  $A$

### 3 Popis algoritmu

Základem tohoto algoritmu je *QPointFB*, což je nový datový typ nesoucí informaci o parametrech  $\alpha, \beta$ . Tyto parametry charakterizují polohu průsečíků dvou hran uvnitř těchto dvou hran. Datový typ *QPointFB* navíc obsahuje informaci o poloze bodu vůči polygonu. Do algoritmu vstupují polygony  $A = \{p_i\}_{i=1}^n$  a  $B = \{q_j\}_{j=1}^m$ , jejichž první a poslední bod je totožný a jehož orientace je proti směru hodinových ručiček.

#### 3.1 Výpočet průsečíků

Na začátku je potřeba vypočítat průsečík  $b_{ij}$  hran  $A, B$ , pokud existuje. Nejprve je potřeba zjistit vzájemnou pozici hran, jelikož chceme pouze ty, které se protínají. Vypočítají se směrové vektory  $\vec{u}, \vec{v}, \vec{w}$  pro hrany  $e_i = (p_i, p_{i+1})$ ,  $e'_j = (q_j, q_{j+1})$ , kde určíme  $p_i = [x_i, y_i]$ ,  $p_{i+1} = [x_{i+1}, y_{i+1}]$ ,  $q_j = [x_j, y_j]$ ,  $q_{j+1} = [x_{j+1}, y_{j+1}]$ :

$$\vec{u} = (x_{i+1} - x_i, y_{i+1} - y_i),$$

$$\vec{v} = (x_{j+1} - x_j, y_{j+1} - y_j),$$

$$\vec{w} = (x_i - x_j, y_i - y_j).$$

Poté dopočítáme determinanty  $k_1, k_2, k_3$

$$k_1 = v_x \cdot w_y - v_y \cdot w_x,$$

$$k_2 = u_x \cdot w_y - u_y \cdot w_x,$$

$$k_3 = v_y \cdot u_x - v_x \cdot u_y,$$

z nichž vypočítáme parametry  $\alpha, \beta$

$$\alpha = \frac{k_1}{k_3},$$

$$\beta = \frac{k_2}{k_3}.$$

V případě  $k_1 = 0, k_2 = 0, k_3 = 0$  jsou hrany  $e_i, e'_j$  kolineární.

V případě  $k_1 = 0, k_2 = 0$  jsou hrany  $e_i, e'_j$  rovnoběžné.

V předchozích případech průsečík neexistuje. V případě, kdy průsečík existuje jako  $0 \leq \alpha \leq 1$  a  $0 \leq \beta \leq 1$ , je potřeba přepočítat jeho souřadnice  $X, Y$

$$X = p_{1x} + \alpha u_x,$$

$$Y = p_{1y} + \alpha u_y.$$

V každém dalším případě jsou hrany  $e_i, e'_j$  mimoběžné.

### 3.2 Vkládání průsečíků do polygonů

Může nastat situace, kdy hrana  $e$  polygonu  $A$  protne hranu  $e'$  polygonu  $B$  více než jednou (a naopak). Kvůli tomu je nutné testovat hranu  $A$  vůči všem hranám polygonu  $B$ . Pokud existuje průsečík  $b_{ij}$ , bude přidán do polygonu  $B$  na pozici  $i + 1$ . Aby byl průsečík přidán i do polygonu  $A$ , musí být průsečík přidán do slovníku  $D$ . Slovník  $D$  je přidán do polygonu  $A$  po nalezení všech průsečíků.

### 3.3 Ohodnocení vrcholů polygonů

Dále bude probíhat ohodnocení vrcholů  $A, B$ , které závisí na určení polohy polygonu vůči druhému polygonu. O poloze  $e_i = (p_i, p_{i+1})$  k  $B$  rozhoduje poloha vnitřního bodu, která se vypočítá jako

$$\overline{p_i} = 0,5(p_i + p_{i+1}).$$

Analogicky poloha  $e'_j = (q_j, q_{j+1})$  k  $A$  se vypočítá jako

$$\overline{q_i} = 0,5(q_j + q_{j+1}).$$

Každému  $\overline{p_i} \in A$  určíme polohu vzhledem k  $B$  a každému  $\overline{q_i} \in B$  určíme polohu vzhledem k  $A$ . Z ohodno-

covací funkce  $g$  víme, že

$$g(e_i, B) = g(\overline{p_i}, B) = \begin{cases} \text{vně, } \overline{p_i} \notin B, \\ \text{na hranici } A/B, \overline{p_i} \in \partial B, \\ \text{uvnitř, } \overline{p_i} \in B. \end{cases} \quad (1)$$

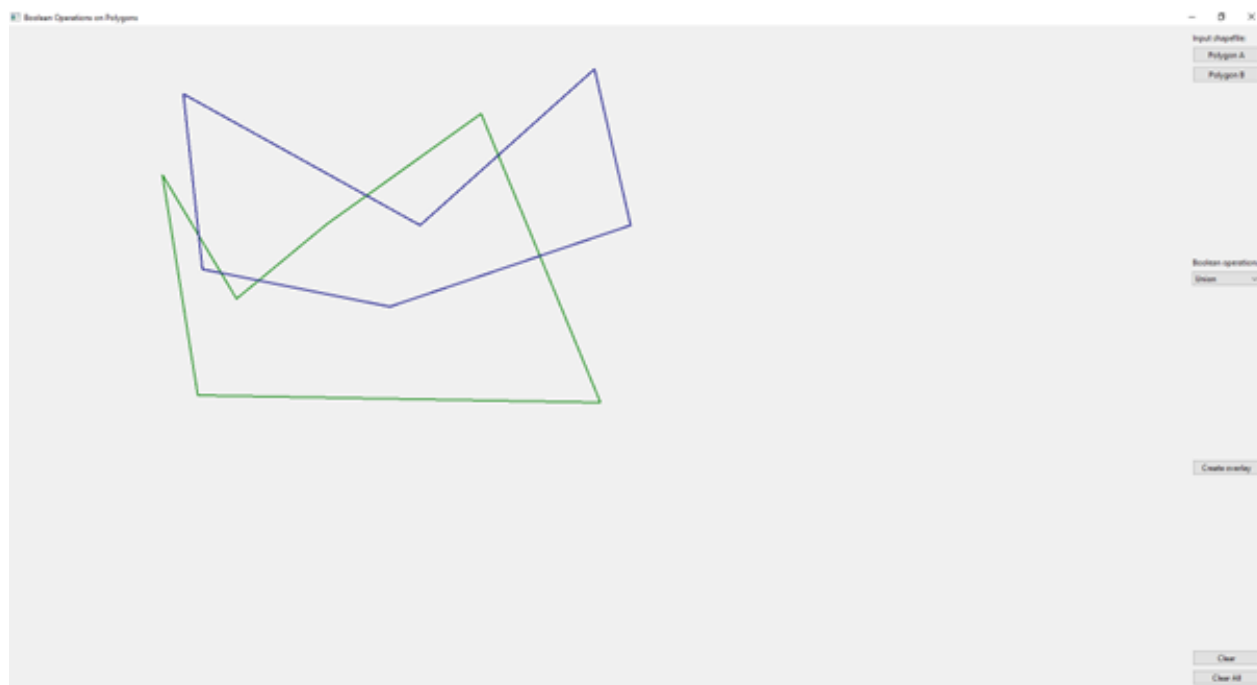
$$g(e'_j, A) = g(\overline{q_j}, A) = \begin{cases} \text{vně, } \overline{q_j} \notin A, \\ \text{na hranici } A/B, \overline{q_j} \in \partial A, \\ \text{uvnitř, } \overline{q_j} \in A. \end{cases} \quad (2)$$

### 3.4 Výběr hran podle pozice

Následuje výběr hrany podle pozice. Jsou vybrány hrany z obou polygonů, které měly k druhému polygonu konkrétní pozici (určené výše). Bude se procházet přes všechny vrcholy polygonu. Pokud se rovná pozice bodu se zadanou pozicí, tak se z tohoto bodu a jeho následujícího bodu vytvoří hrana. Z polygonu  $A$ , respektive  $B$ , jsou v případě průniku vybrané ty hrany uvnitř polygonu  $A$ , respektive  $B$ . V případě sjednocení jsou z polygonu  $A$ , respektive  $B$ , vybrány ty hrany mimo polygon  $A$ , respektive  $B$ . V případě rozdílu  $B - A$  jsou z polygonu  $B$  vybrány hrany mimo polygon  $A$ , které jsou zároveň uvnitř polygonu  $B$ . Opačně to platí v případě rozdílu  $A - B$ .

## 4 Dokumentace

Program je rozdělen do 10 modulů. Modul `mainform.py` je převážně vygenerován pomocí softwaru QT Creator a slouží k vytvoření uživatelského rozhraní a k volání jednotlivých funkcí pomocí tlačítek v UI. (Obr. 1) Modul `algorithm.py` obsahuje algoritmy pro množinové operace s polygony. Modul `draw.py` slouží především pro propojení předešlých dvou modulů a k zajištění vizualizace vstupu a výstupu. Modul `input.py` zajišťuje nahrání vstupních dat v požadovaném formátu. Modul `QPointFB.py` upravuje třídu `QPointF` tak, aby body obsahovaly informaci o poloze vůči druhému polygonu a informace o poloze na hranách polygonů (od počátku hrany). Modul `edge.py` definuje třídu `Edge`, která definuje datový typ hrany. Hrana v sobě nese informaci o počátečním a koncovém bodě. Moduly `booleanoperations.py`, `lineandlineposition.py`, `pointandpolygobposition.py` a `pointlineposition.py` definují třídy odvozené z třídy `Enum`. Mají za úkol přiřadit hodnotám jejich slovní název.



Obr. 1 - Náhled uživatelského rozhraní

### 4.1 Modul `mainform.py`

Tento modul obsahuje třídu `Ui_Mainform`, která je vytvořena automaticky pomocí QT Creatoru. Tato třída byla posléze doplněna o metody, které se spustí při interakci s uživatelským rozhraním. Jedná se o metody



*inputA*, *inputB*, *clickCreateOverlay*, *clickclear* a *clickclearALL*. Metody *inputA* a *inputB* slouží k vyvolání funkce *loadFile* z modulu `input.py`, čímž nahrají vstupní polygony. Metoda *clickCreateOverlay* provede zvolenou množinovou operaci se vstupními polygony. Metoda *clickclear* slouží k odstranění výsledku a metoda *clickclearALL* pro odstranění všeho.

## 4.2 Modul `input.py`

Tento modul obsahuje metodu *loadfile*. V této metodě je nejprve načtena cesta ke vstupním datům. Když není zvolena žádná cesta, metoda se ukončí. Dále je načten vstupní textový dokument, z něhož jsou získány vrcholy polygonu.

## 4.3 Modul `draw.py`

Tento modul obsahuje třídu `Draw`, kterou dědí od třídy `QWidget` z knihovny `PyQt6`. Hlavní funkcí tohoto modulu je vykreslit vstupní polygony a následně výsledky množinových operací. Polygony *A* a *B* jsou uloženy v proměnných *polA* a *polB* ve formátu listů obsahující jednotlivé vrcholy, které jsou ve formátu *QPointFB*. Výsledek množinové operace je v proměnné *res*, která je opět list, který obsahuje hrany ve formátu *edge*.

## 4.4 Modul `algorithm.py`

Zde se nacházejí jednotlivé funkce pro množinové operace s polygony. V modulu se nachází jediná třída *Algorithms*, která obsahuje níže uvedené metody.

Metoda *getPointAndLinePosition* má na vstupu 3 body *QPointFB*, kde 2 body definují přímku a zbylý bod definuje analyzovaný bod. Tato metoda zjistí, ve které polorovině se analyzovaný bod nachází. Výsledek je vrácen ve formátu výčtové třídy definované v modulu `pointlineposition.py`. Hodnoty jsou definovány takto:

- `Left_HP` (1) pro bod v levé polorovině,
- `Right_HP` (-1) pro bod v pravé polorovině,
- `On_Line` (0) pro kolineární bod.

Metoda *get2LinesAngle* má na vstupu 4 body *QPointFB*, které definují 2 přímky, u nichž je zjišťován úhel, který spolu svírají.

Hlavní metoda této třídy - *getPositionPointAndPolygon* - zjišťuje vzájemnou polohu daného bodu ve formátu *QPointFB* a polygonu definovaného jako list *QPointFB*. Výstup je ve formátu výčtové třídy definované v modulu *pointandpolygonposition.py*. Vracené hodnoty jsou:

- Inside (1) pro bod uvnitř,
- Outside (0) pro bod vně,
- Boundary (-1) pro bod na hraně polygonu.

Tato metoda je založena na algoritmu *Winding Number*.

Metoda *get2LinesIntersection* vrací vzájemnou polohu dvou vstupních úseček, které jsou definovány čtyřmi body *QPointFB*. Výsledek je opět ve formátu výčtové třídy definované v modulu *lineandlineposition.py*, a to jako:

- Parallel (1) pro rovnoběžné úsečky,
- Skew (2) pro mimoběžné úsečky,
- Colinear (3) pro úsečky kolineární,
- Intersect (4) pro úsečky s jedním průsečíkem.

Metoda *updateVertices* přidá body do hran dvou vstupních polygonů v místě, kde se jejich hrany vzájemně kříží. Vstupní polygony jsou list *QPointFB*. Výstup tato funkce nemá, pouze aktualizuje vstupní polygony. Metoda *setEdgePosition* přiřadí jednotlivým hranám vstupního polygonu polohy vůči polygonu druhému. Vstupem jsou tedy 2 polygony definovány jako list *QPointFB*. Polohy jsou vztaženy ke středu hrany a jsou ukládány do atributu *position* prvního bodu definující hranu. Poloha je získána pomocí metody *getPositionPointAndPolygon*.

Metoda *getEdges* vybere hrany vstupního polygonu, které mají pozici vůči druhému polygonu stejnou, jako je vstupní parametr metody. Na vstupu jsou tedy tři proměnné, polygon list *QPointFB*, žádaná pozice hrany určená pomocí výčtové třídy *PointAndPolygonPosition* a hrany, definované jako list *edges*. Výstupem je aktualizovaný list hran.

Metoda *createOverlay* provede zvolenou množinovou operaci polygonů. Na vstupu jsou 2 polygony jako list *QPointFB* a množinová operace definována výčtovou třídou *BooleanOperation* z modulu *booleanoperation.py*. Výstupem je list hran, které splňují danou množinovou operaci.

## 4.5 Modul Uživatelské datové typy

V modulech `edge.py` a `qpointfb` jsou definovány uživatelské datové typy. Uživatelský datový typ *QPointFB* je odvozen od třídy `QPointF`. Stejně jako *QPointF* nese informaci o poloze bodu v osách  $x$  a  $y$ . Následně ještě přidává hodnotu  $\alpha$  a  $\beta$ , které určují polohu průsečíku dvou hran. Tyto atributy nabývají hodnoty z intervalu  $(0, 1)$  pro body, které jsou průsečíkem hran. Poté je ještě přidán atribut *position*, který reflektuje pozici středu hrany vůči druhému polygonu. Datový typ *Edge* definuje hrany jako začáteční a koncový bod typu *QPointFB*.

Pseudokódy konkrétních metod jsou uvedeny níže.

### ***get2LinesIntersection:***

*Vypočti směrové vektory  $u(p_1p_2), v(p_3p_4), w(p_1p_3)$ , kde  $p_i$  jsou body definující 2 úsečky*

*Vypočti koeficienty  $k_1, k_2$  a  $k_3$*

***Když  $k_3 = 0$ :***

*Vrať úsečky jsou kolinéární, bod průsečíku je None*

*Vypočti parametry  $\alpha$  a  $\beta$*

***Když  $k_1ak_2 = 0$ :***

*Vrať úsečky jsou paralelní, bod průsečíku None*

***Když  $0 \leq \alpha \leq 1$  a  $0 \leq \beta \leq 1$ :***

*Získej hodnotu průsečíku jako:*

$$X = p_{1x} + \alpha u_x, Y = p_{1y} + \alpha u_y$$

*Definuj průsečík *QPointFB*  $q(x, y, \alpha, \beta)$*

*Vrať úsečky mající průsečík  $q$*

**updateVertices:**

*Inicializuj index  $i = 0$*

**Dokud**  $i < \text{počet vrcholů a polygonu } A$ :

*Inicializuj slovník  $D$*

*inicializuj index  $j = 0$*

**Dokud**  $j \leq \text{počet vrcholů b polygonu } B$ :

*Získej průsečík dvou hran  $h_1(a_i, a_{i+1}), h_2(b_j, b_{j+1})$*

**Když** mají právě jeden průsečík:

*Získej polohu průsečíku  $I$  na obou aktuálních hranách jako  $\alpha$  a  $\beta$*

*Přidej průsečík do  $D$ , kde key je hodnota,  $\alpha$  item je  $I$*

*Inkrementuj  $j = j + 1$*

*Přidej průsečík do polygonu  $B$  na pozici  $j$*

*Inkrementuj  $j = j + 1$*

**Dokud**  $i \leq \text{počet vrcholů a polygonu } A$ :

**Když** existuje nějaký průsečík v  $D$ :

**Pro** všechny záznamy v seřazených podle hodnoty klíče  $k$  ve slovníku:

*Inkrementuj  $i = i + 1$*

*Vlož do vrchol v do polygonu  $A$  na pozici  $i$*

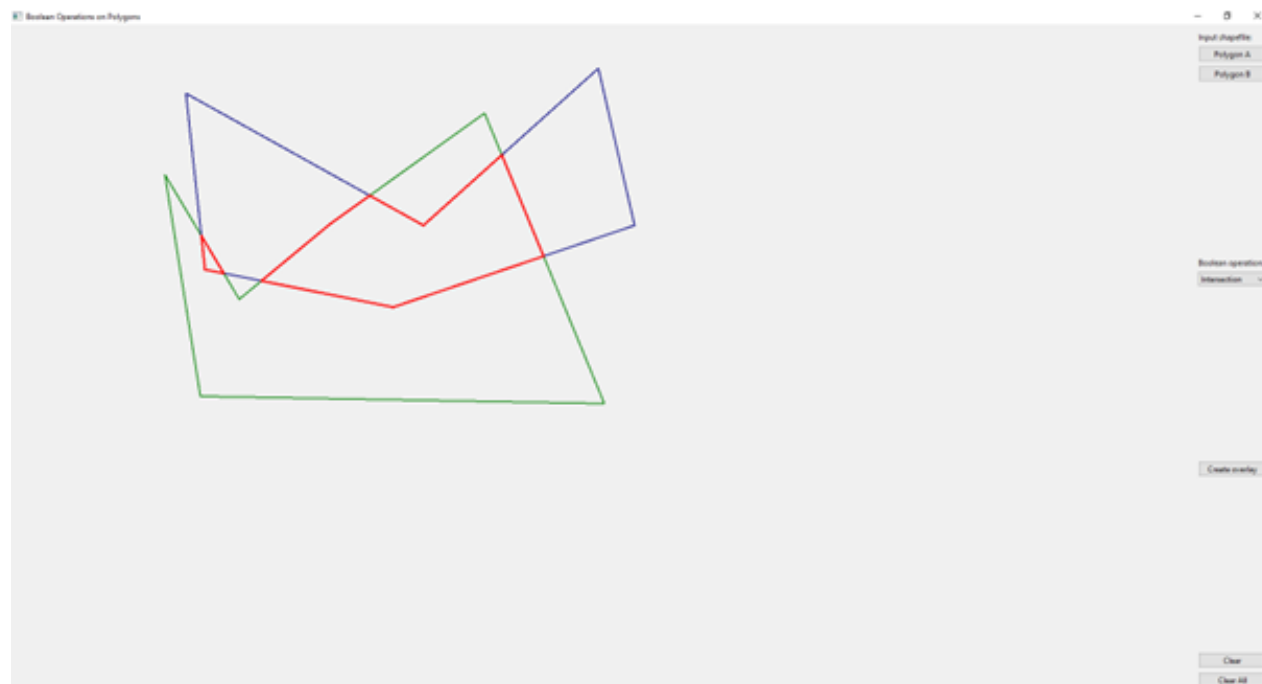
*Inkrementuj  $i = i + 1$*

## 4.6 Vstup

Vstupní polygony jsou nahrány do uživatelského rozhraní pomocí tlačítek *Polygon A* a *Polygon B*. Každé tlačítko slouží pro nahrání jednoho polygonu. Data jsou v textovém dokumentu tak, že jsou zde zapsány na každém řádku souřadnice jednotlivých vertexů polygonu. Souřadnice  $x$  a  $y$  jsou odděleny mezerou. Souřadnice jsou vztaženy k souřadnicím zobrazovacího okna (tzn. od 0, 0 až po maximální rozměry obrazovky).

## 4.7 Výstup

Výstup je zajištěn graficky tak, že jsou zvýrazněny hrany, které splňují zvolenou množinovou operaci. (Obr. 2)



Obr. 2 - Výstup operace Intersection (červeně)

## 5 Závěr

Program vyhodnotí zvolenou množinovou operaci dvou vstupních polygonů. U programu by mohl být vylepšen vstupní formát tak, aby mohly být jednotlivé polygony nahrávány například ve formátu *.shp* či aby byly například v JTSK (pro lepší aplikaci na reálná data).

## 6 Zdroje

MORAVEC, L. (2008) - Webová aplikace pro výuku matematické logiky na střední škole. Diplomová práce.

Univerzita Karlova, Matematicko-fyzikální fakulta, Praha, 168 s.

Přednášky z předmětu *Algoritmy počítačové kartografie*