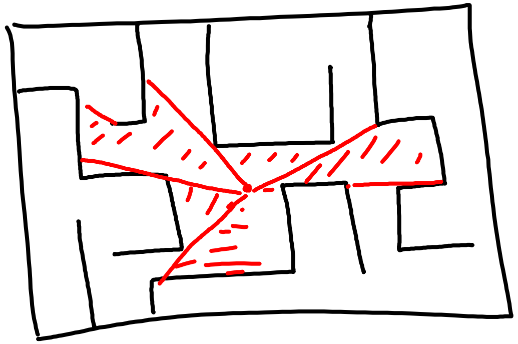
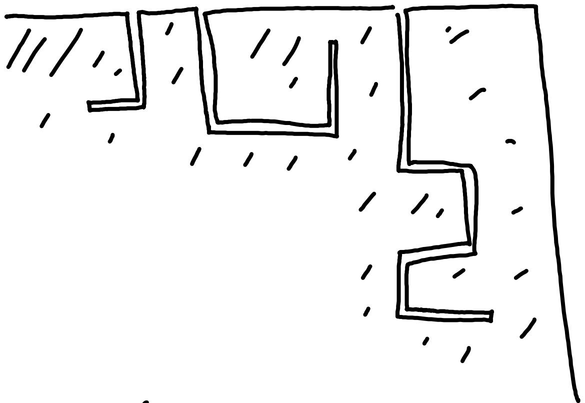
TRIANGULACE ① MNOHOÚHELNÍKŮ

Motivace: hlídání galerie kamerami



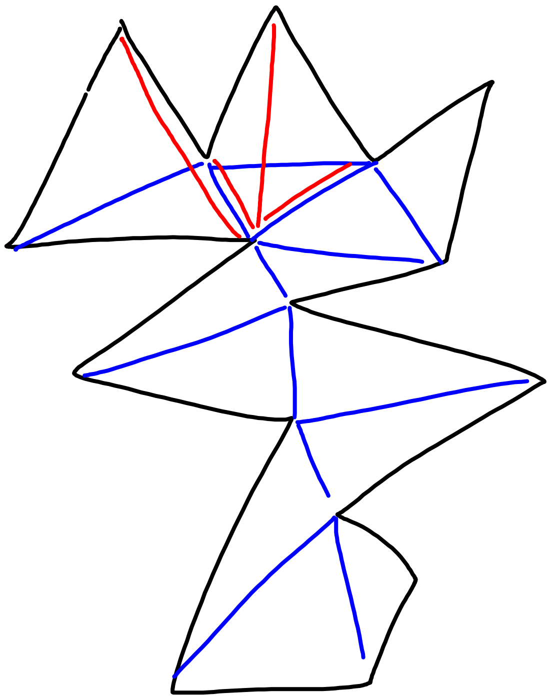
Galerie



nekonvexní mnohoúhelník

2

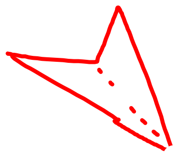
2



Mějme nekonvexní mnohoúhelník. Chceme jej triangulovat tak, aby vrcholy trojúhelníků ležely ve vrcholech mnohoúhelníka.

Triangulací je více, nicméně všechny mají stejný počet trojúhelníků.

Pro n-úhelník je těchto trojúhelníků



$n-2$

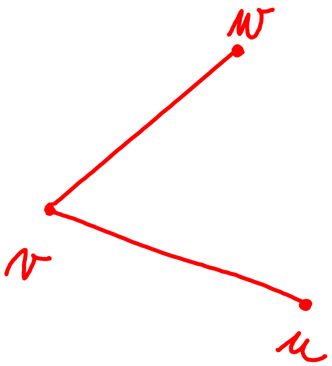
(3)

Důkaz předchozího tvrzení indukcí:

$n = 3$ zřejmé

nechť tvrzení platí pro $k < n$, $n \geq 4$.

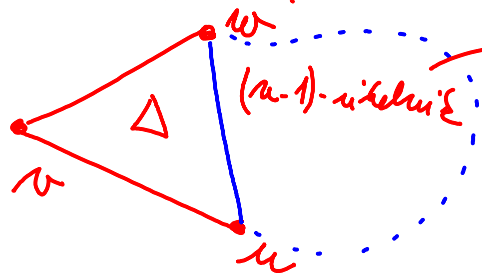
n-úhelník, vezmeme vrchol nejvíce vlevo v



u, w sousední vrcholy

2 možnosti

(1) úsečka uw leží celá v našem mnohoúhelníku

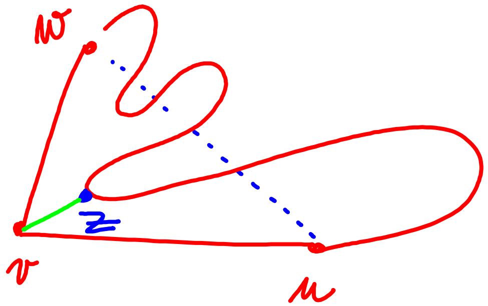


→ triangulace pomocí $(n-3)$-trojúhelníků

Celý n-úhelník lze triangulovat pomocí $n-2$ Δ.

4

(2) úsečka uw neleží celá v našem mnohoúhelníku



Z vrcholů mnohoúhelníka, které jsou v $\triangle$ $v\,u\,w$ vezmeme ten, který je nejdál od úsečky $u\,w$ ... označme ho z

Tato úsečka vz nemůže protínat hranice našeho mnohoúhelníka

vz dělí náš mnohoúhelník na 2 mnohoúhelníky

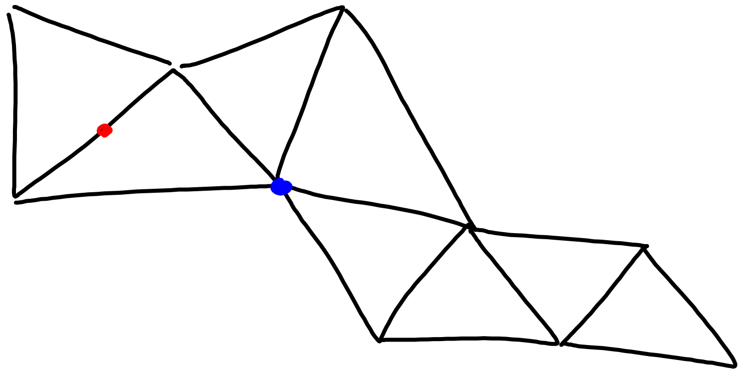
n_1-úhelník a n_2-úhelník

$$n_1+n_2 = n+2, \quad n_1<n,\ n_2<n$$

Tedy podle ind. předpokladu lze oba mnohoúhelníky triangulovat na $n_1-2 + n_2-2$ trojúhelníků, tj dohromady na $n_1+n_2-4 = n+2-4 = n-2$ ✓

5

Zpět ke galerii – máme-li její triangulaci, pak rozmístíme kamery



$n \cdot 2$ △

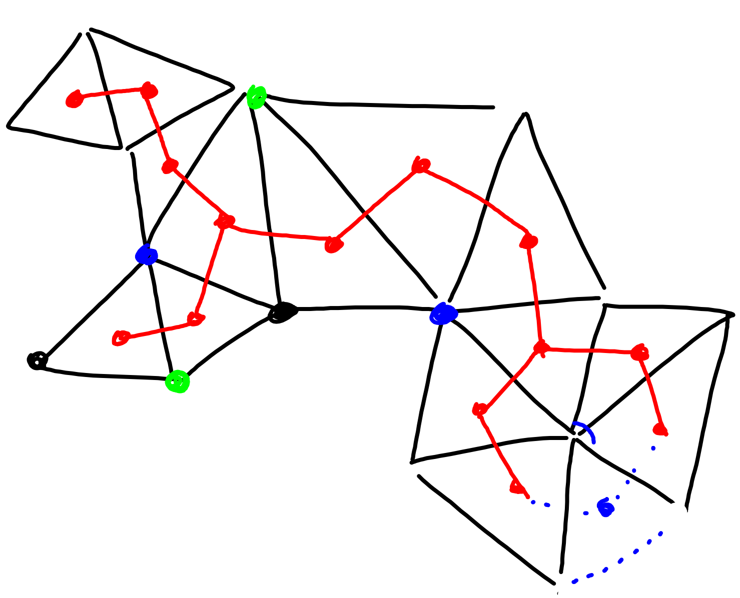
$\sim \frac{n \cdot 2}{2}$ kamer

Pokud bychom uzly △ obarvili 3 barvami tak, aby každý △ obsahoval uzly všech tří barev

Pak stačí dát kamery do uzlů jedné barvy $\Rightarrow \left\lceil \frac{n}{3} \right\rceil$ kamer

(6)

Obarvení vrcholů uděláme pomocí duálního grafu

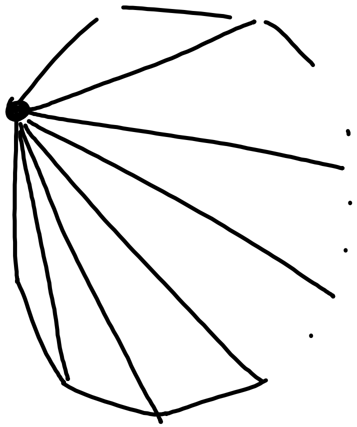


Duální graf je souvislý
strom (není v něm kružnic
(díky tomu, že náš mnohoú
neobsahuje žádnou "díru"

Algoritmus pro triangulaci n-úhelníku

P n-úhelník

je-li P konvexní, je úloha triviální

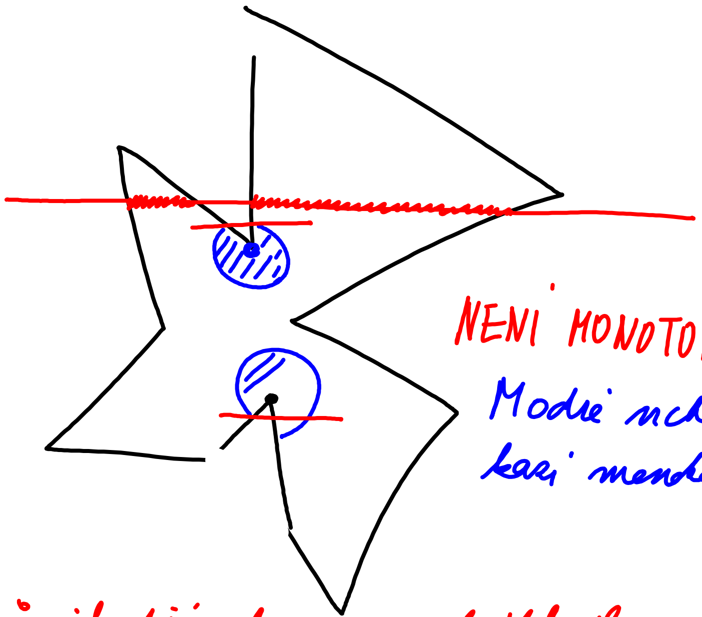


Další třída mnohoúhelníků, které lze relativně jednoduše triangulovat (v čase $O(n)$) jsou tzv.:

monotónní mnohoúhelníky (vzhledem k ose y)

8

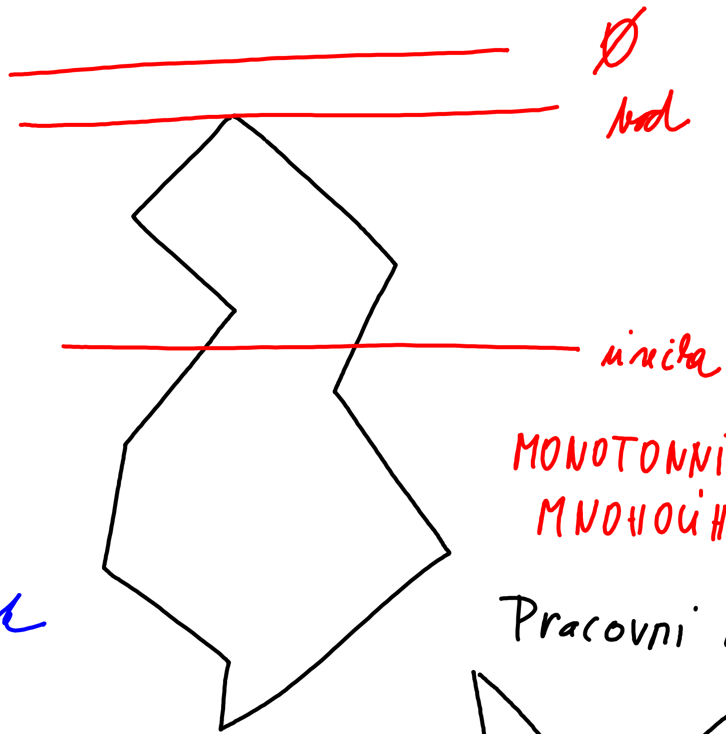
Příklady



NENÍ MONOTONNÍ

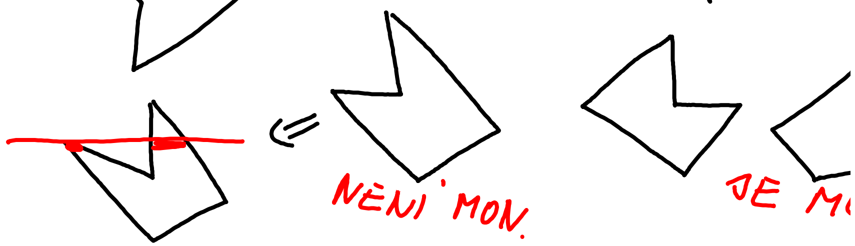
Modré nehody kazí monotonnost

Průnik přímky s mnohoúhelníkem není konvexní množina pro nějakou vodorovnou přímku



MONOTONNÍ MNOHOÚHELNÍK

Pracovní definice



9

Myšlenka algoritmu

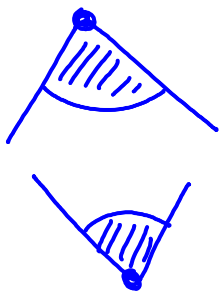
(1) rozdělit daný n-úhelník na monotónní mnohoúhelníky

(2) triangulovat každý monotónní mnohoúhelník zvlášť

Rozklad na monotónní mnohoúhelníky

K tomu potřebujeme klasifikaci vrcholů

1. start
2. end



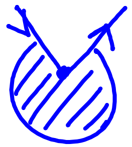
šrafování ukazuje, kde je mnohoúhelník

10

(3) regular

(4) split

(5) merge



 ~~[illegible]~~ z vrcholu vychází hrana nahoru

 z vrcholu jde hrana dolů

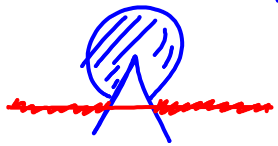
Přesná definice „nahoru" a „dolů" je dána lexikografickým uspořádáním

$$p > q \iff p_y > q_y \text{ nebo } p_y = q_y \text{ a } p_x < q_x$$

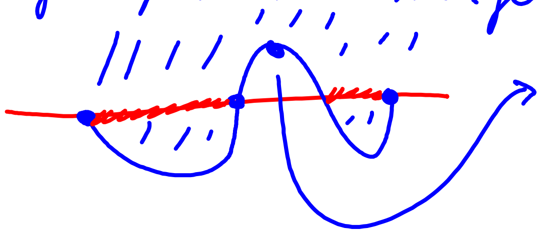
11

Věta: Mnohoúhelník je monotónní právě když nemá žádný split ani merge uzel

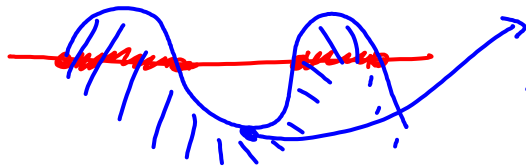
Dk: Má-li jeden takový uzel, není monotónní



Opačnou implikaci ukážeme nepřímo. Tj. Není-li mnohoúhelník monotónní, obsahuje split nebo merge uzel.



nejvyšší uzel zde je split uzel



nejnižší uzel zde je merge

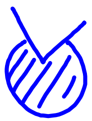
12

<u>Idea algoritmu</u> – metoda zametací přímky, postupné odstraňování split a merge vrcholů

<u>Intuitivní popis</u>

1. Odstranění split vrcholů 

při průchodu zametací přímky spojíme vrchol s <u>vhodným vrcholem</u> nad zametací přímkou. Tím se daný bod stane regulárním ve dvou mnohoúhelnících.

(2) Odstraňování merge vrcholů () je složitější. se děje zpětně, nikoliv při průchodu zametací přímky

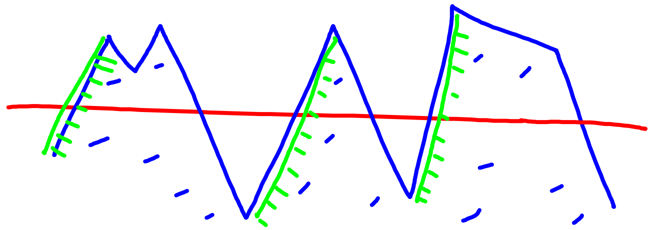
(13)

Potřebné struktury

Fronta Q .... vrcholy mnohoúhelníka uspořádané podle nějakého lexikografic-
kého uspořádání (Na začátku nějaký vrchol, postupně ubýva

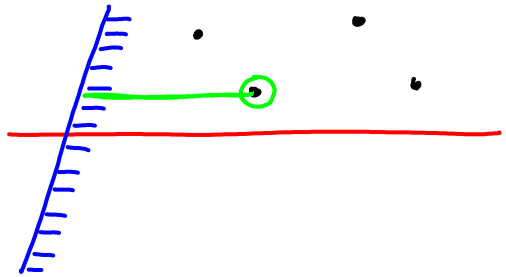
- 1 shora dolů
- 2 zleva doprava
- 3
- 4

Vyvážený binární strom T .... určují pořadí úseček mnohoúhelníka, k
protínají zametací přímku. Ale jenom
těch, které mají mnohoúhelník zprava



14

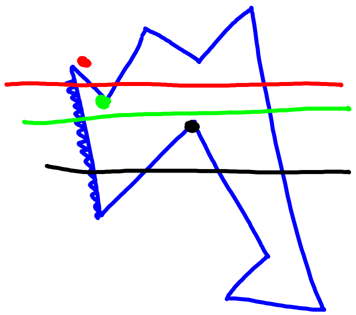
9) ji každou úsečku ve stromu určíme vrchol mnohoúhelníku tzv. helper úseč e



zam. přímka

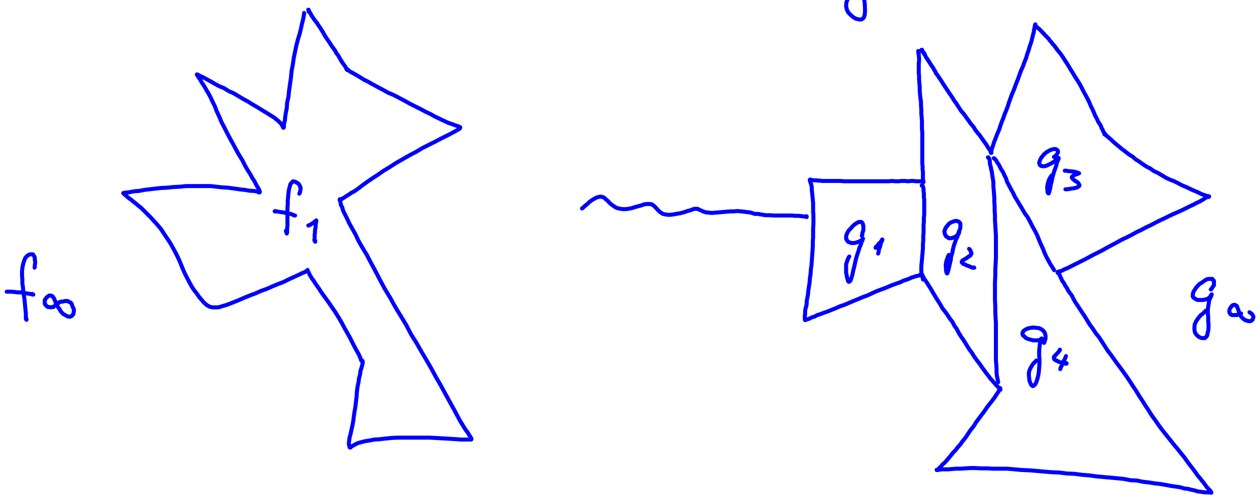
(vpravo od e)

helper(e) ... je vrchol mnohoúhelníka, který leží nad zametací přímkou, vrchol mnohoúhelníka může být spojen vodorovnou úsečkou s hranou e a ze všech těchto bodů je nejblíže k zametací přímce



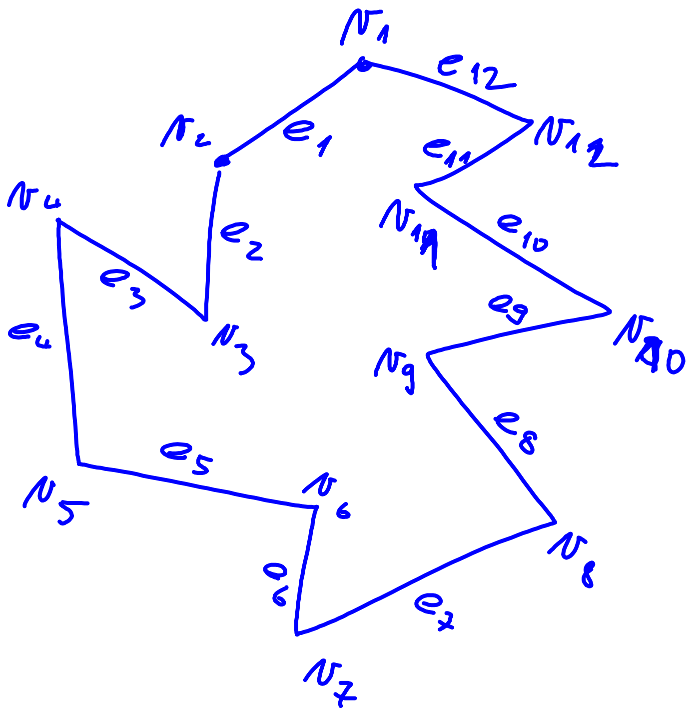
15

Daný mnohoúhelník je popsán pomocí dvojitě vázaného seznamu, při provádění algoritmu doplňujeme nové hrany a oblasti a upravíme dvojitě vázaný seznam, který popisuje rozklad daného mnohoúhelníka na monotónní mnohoúhelníky



(16)

Označení vrcholů a hran podle mého kód, unicé?



Vrcholy uspořádáme v lexikogra[fickém] uspořádání a vložíme do fro[nty] Q

Vezmeme prázdný strom T

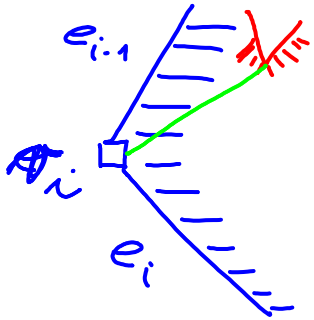
Při průchodu zaznamenávání hran (vrcholem) popíšeme, co je potřeba udělat v závislosti na typu vrcholu.

17

Příchod vrcholu typu start

1. zařadíme e_i do stromu T
2. $v_i \Longrightarrow helper(e_i)$

Příchod regulárním vrcholem (mnohoúhelník vpravo)



1. ~~[illegible]~~ je-li pomocník hrany e_{i-1} vrchol typu merge, spoj s ním v_i
2. Odstraň e_{i-1} ze stromu T
3. Vlož e_i do stromu T
4. helper e_i bude v_i