Gymnázium Evolution Jižní Město

Eric Dusart

Polytop maximální dimenze a minimálního obvodu s vrcholy v dané množině bodů.

Ročníková práce

Školitel práce:

Mgr. Adam Klepáč

Školní rok: 2023/2024

Poděkování

Děkuji bohovi Adamovi. KLEPY = BŮH!!!

Prohlášení

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

V Praze dne 24. března 2024 Eric Dusart

Abstrakt

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Klíčová slova: graf, polytop, algoritmus

Abstract

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Keywords: graph, polytope, algorithm

Obsah

Použitá notace

Symbol	Význam
$\mathbb{R}^+$	$\{x \in \mathbb{R} \mid x \geq 0\}$
$\binom{n}{k}$	Kombinační číslo: $\frac{n!}{k!(n-k)!}$
$p(a, b, c)$	Obvod trojúhelníku a, b, c.
$\lvert a \rvert$	Absolutní hodnota z a.
$min(), max()$	Funkce minimum a maximum.

Základní definice

Definice 1 (Polytop). *Polytop* dimenze $n \in \mathbb{N}$ je uzavřená podmnožina $P \subseteq \mathbb{R}^n$ definovaná induktivně:

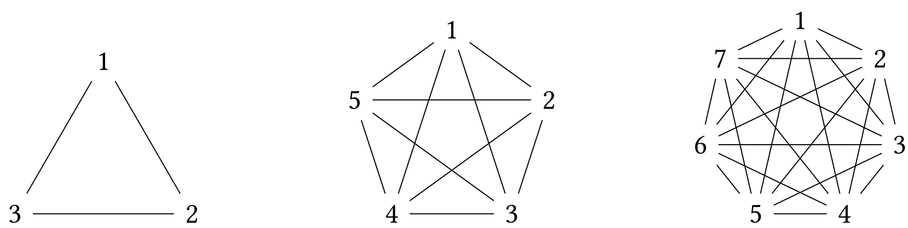
- Polytop dimenze 1 je úsečka.
- Polytop dimenze n je slepením polytopů dimenze $n-1$, jež spolu mohou sdílet stěny libovolné dimenze, kde *stěnou* polytopu rozumíme jeho libovolnou podmnožinu jsoucí rovněž polytopem. Zároveň neexistuje nadrovina (podprostor dimenze $n-1$), která by obsahovala všechny jeho vrcholy. [Ada24]

Definice 2 (Bod). Bod je uspořádaná n-tice $(x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$. Ve $2D$ budu používat značení $a = (a_x, a_y)$

Definice 3 (Vzdálenost). Zobrazení $d : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}^+$ nám určí vzdálenost dvou bodů $u, v \in \mathbb{R}^n$ podle předpisu $d(u, v) \coloneqq \sqrt{(v_1 - u_1)^2 + (v_2 - u_2)^2 + \ldots + (v_n - u_n)^2}$.

Definice 4 (Ohodnocený graf). $G = (V, E, w)$ je ohodnocený graf, kde V je množina vrcholů, E je množina dvojprvkových podmnožin $E \subseteq \binom{V}{2}$ a w je libovolné zobrazení $E \longrightarrow \mathbb{R}^+$, které hranám přiřazuje jejich váhu.

Definice 5 (Úplný ohodnocený graf). Úplný ohodnocený graf $G = (V, E, w)$ má každé dva vrcholy spojeny hranou, neboli $E = \binom{V}{2}$. Takový graf můžeme také zapsat jako $K_n \coloneqq (V, \binom{V}{2}, w)$.



Obrázek 1: Úplné grafy K_3, K_5 a K_7

Definice 6 (Podgraf). Podgraf $H = (V, E)$ je tvořen podmnožinou vrcholů a hran jiného grafu. Hrany grafu H musí mít oba vrcholy v množině vrcholů V.

Definice 7 (Cesta). Cestou v grafu nazveme posloupnost **různých** vrcholů $v_1, \ldots, v_n$, pokud $\forall i \in \{1, \ldots, n-1\}$ platí $\{v_i, v_{i+1}\} \in E$.

Definice 8 (Váha cesty). Pokud cestu tvoří posloupnost vrcholů $v_1, \ldots, v_n$, tak váha cesty je rovna

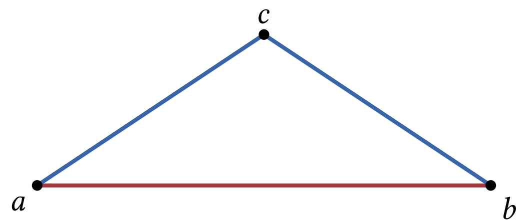
$$w(\{v_0, \ldots, v_n\}) = \sum_{i=1}^{n-1} w(\{v_i, v_{i+1}\}).$$

Definice 9 (Cyklus). Cyklus je posloupnost vrcholů $v_1, \ldots, v_n, v_1$, kde $v_1, \ldots, v_n$ je cesta a poslední dva vrcholy $\{v_1, v_n\} \in E$.

Definice 10 (Váha cyklu). Pokud cyklus tvoří posloupnost vrcholů $v_1, \ldots, v_n$, tak váha cesty je rovna

$$w(\{v_0, \ldots, v_n, v_0\}) = w(\{v_1, v_n\}) + \sum_{i=1}^{n-1} w(\{v_i, v_{i+1}\}).$$

Definice 11 (Trojúhelníková nerovnost). Trojúhelníková nerovnost říká, že pro každé tři různé body a, b, c platí $d(a,b)+d(c,b) \geq d(a,b)$, neboli vzdálenost mezi dvěma body je vždy menší nebo rovna součtu vzdáleností mezi těmito body a třetím bodem.



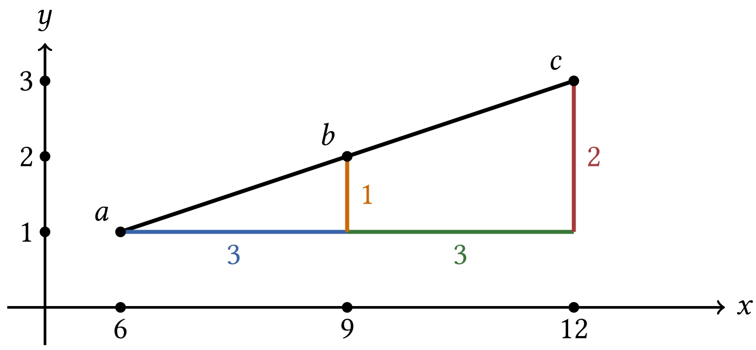
Obrázek 2: Trojúhelníková nerovnost

Definice 12 (Soused). V grafu $G = (V, E, w)$ je bod u soused bodu v, pokud $\{u, v\} \in E$.

Definice 13 (Big O notation). idk

Definice 14 (Obvod trojúhelníku). Definujeme si zobrazení $p : E \longrightarrow \mathbb{R}^+$ podle předpisu: $p(a, b, c) := d(a, b) + d(b, c) + d(a, c)$, které nám určuje obvod trojúhelníku a, b, c.

Definice 15 (Podobnost trojúhelníků a body na jedné přímce). Jsou dány 3 body: $a = (a_x, a_y)$, $b = (b_x, b_y)$, $c = (c_x, c_y)$. Tyto body jsou na přímce, pokud platí, že trojúhelníky $(a_x, a_y), (b_x, a_y), (b_x, b_y)$ a $(a_x, a_y), (c_x, a_y), (c_x, c_y)$ mají stejný poměr stran, neboli platí: $\frac{c_y - a_y}{c_x - a_x} = \frac{b_y - a_y}{b_x - a_x}$.



Obrázek 3: Příklad podobnosti trojúhelníků

Úvod

Najít polytop maximální dimenze a minimálního obvodu v dané množině bodů je složité, hlavně při vyšším počtu bodů. Zdá se, že k vyřešení problému je třeba vyzkoušet všechny možnosti, ale tomu se budeme vyhýbat.

Začněme analýzou tohoto problému. Cílem je najít polytop maximální dimenze a minimálního obvodu v dané množině bodů. Ve 2D budeme problém řešit trochu jinak, než ve vyšších dimenzích.

Část I

Teoretická část

Kapitola 1

Problém v 1D

Problém v 1D je velice jednoduchý. Vstupem bude množina bodů $V \subset \mathbb{R}$ ve které budeme hledat nejkratší úsečku. Výstupem bude úsečka $\{a, b\} \mid a, b \in V$. Následujícím algoritmem můžeme tento problém vyřešit.

1.1 Algoritmus

1. Nejprve body seřadíme.
2. Projdeme všechny body $x_1 \dots x_n \in V$ a spočítáme vzdálenost všch po sobě jdoucích bodů: $|x_i - x_{i+1}|$, kde i značí kolikátý bod procházíme. Tuto vzdálenost si uložíme, a pokud je menší, než ta doposud uložená, změníme ji.
3. Po tom, co projdeme všechny body, uložená minimální vzdálenost je řešením problému.

Důkaz. Zřejmý. Algoritmus skončí, protože prochází konečnou množinu bodů a je korektní, protože spočítá všechny vzdálenosti po sobě jdoucích bodů a vybere tu minimální. □

Algoritmus 1: Algoritmus na hledání úsečky s minimální délkou.

input : Množina bodů V v 1D.
output: Dvojice bodů a jejich vzájemná vzdálenost.

Mám tam psát sorting algoritmus nebo můžu počítat, že vstup už bude seřazený? ;
$d \leftarrow (x_0, x_\infty, \infty)$;
for $x, i \in V$ **do**
  $distance \leftarrow |x_i - x_{i+1}|$;
  if $distance < d$ **then**
    $d \leftarrow (x_i, x_{i+1}, distance)$;

Kapitola 2

Problém ve 2D

Problém ve 2D je o něco jednodušší, než ve více dimenzích. Ve 2D budu hledat trojúhelník s minimálním obvodem. První algoritmus, který by nás napadl by byl ten, který vyzkouší všechny trojúhelníky a vybere ten s minimálním obvodem. Ten ale vůbec není časově efektivní. Pokusím se najít, naprogramovat a dokázat nějaký, který je časově efektivní.

2.1 Algoritmus

Nechť V je množina bodů v rovině a pro každé dva body $u, v \in V$ označme $d(u, v)$ jejich vzájemnou vzdálenost. Množinu hran označíme $E = \binom{V}{2}$ a váhu, neboli ohodnocení, nám určuje zobrazení w dané předpisem $w(\{u, v\}) := d(u, v) \mid \forall(u, v) \in E$. Nyní si můžeme definovat graf $G = (V, E, w)$. Tímto je příprava hotova. Následuje hledání cyklu délky 3 s minimální váhou.

Náhodně vybereme jednu hranu $\{u, v\} \in E$ a odebereme ji z množiny hran E. Dále potřebujeme najít cestu s minimální váhou mezi body u a v. K tomuto použijeme Dijkstrův algoritmus, který nám vrátí cestu délky dva. Ta se bude skládat ze dvou hran: $\{u, j\}$ a $\{j, v\}$. Teď musíme zkontrolovat, zda tato cesta s hranou $\{u, v\}$ opravdu tvoří trojúhelník, protože se může stát, že body u, j, v leží na jedné přímce. K tomu využijeme podobnost trojúhelníků. Definujeme si body $a := min(\{u_x, j_x, v_x\}), c := max(\{u_x, j_x, v_x\})$ a bod b je zbývající. Pak platí $a_x < b_x < c_x$. Body se nacházejí na přímce právě tehdy, pokud rovnice $\frac{c_y - a_y}{c_x - a_x} = \frac{b_y - a_y}{b_x - a_x}$ je pravdivá.

Poznámka 1 (jiný způsob). Předchozí způsob funguje perfektně pro počítače, ale pro člověka je v některých případech zbytečně komplikovaný. Tyto případy nastávají, když se body nacházejí na horizontálních nebo vertikálních přímkách. To jde poznat tak, že $u_y = j_y = v_y$ nebo $u_x = j_x = v_x$ V případě, že se body nacházejí na horizontálních přímce, musíme z grafu odebrat hranu $\{min(\{u_y, j_y, v_y\}), max(\{u_y, j_y, v_y\})\}$ a pokud jde o vertikální přímku, musíme z grafu odebrat hranu $\{min(\{u_x, j_x, v_x\}), max(\{u_x, j_x, v_x\})\}$.

Pokud tyto body tvoří trojúhelník, pak tvoří podgraf $T = (V_T, E_T, w)$, kde množina bodů $V_T = \{u, j, v\}$, množina hran $E_T = \{\{u, j\}, \{j, v\}, \{u, v\}\}$ a váha hran se zachová. Pokud tento podgraf bude mít celkovou váhu menší než ten, který jsme doposud našli, uložíme jej a vrátíme hranu $\{u, v\}$ do množiny hran E. Tento postup opakujeme dokud nevyzkoušíme všchny hrany. Výsledkem bude trojúhelník s minimálním obvodem. Na obrázku 2.1 můžete vidět jak algoritmus funguje.

2.1.1 Algoritmus v pseudokódu

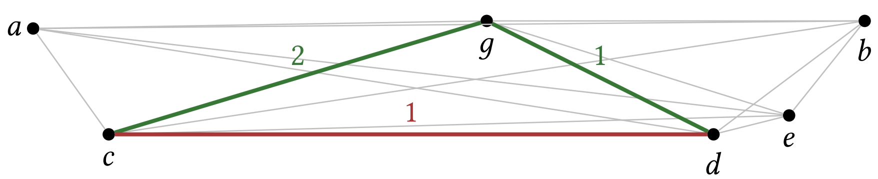
Pseudokód je popis jednotlivých kroků v algoritmu s použitím základní logiky programovacích jazyků. Následuje náš algoritmus napsaný v pseudokódu.

Algoritmus 2: Algoritmus na hledání cyklu délky 3.

input : Množina bodů V v rovině, kde každý bod je reprezentován jako dvojice souřadnic (v_x, v_y)

output: Trojúhelník $T = (V_T, E_T, w)$

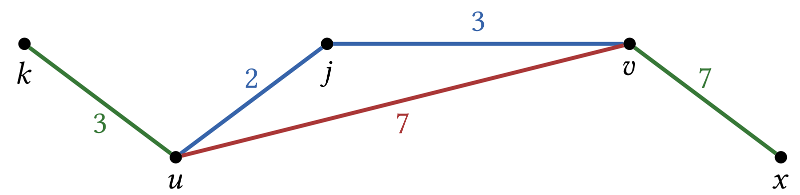
```
for u ∈ V do
    for v ∈ V do
        d(u, v) ← √((v_x − u_x)² + (v_y − u_y)²);
        w({u, v}) ← d(u, v);
E ← (V choose 2);
G ← (V, E, w);
min_T ← ∞;
for {u, v} ∈ E do
    E ← E \ {u, v};
    j ← dijkstra(G, u, v);
    a = min(u_x, j_x, v_x);
    c = max(u_x, j_x, v_x);
    b = {u, j, v} \ {a, c};
    if (c_y − a_y)(b_x − a_x) == (b_y − a_y)(c_x − a_x) then
        if {a, c} == {u, v} then
            continue;
        else
            E ← E ∪ {u, v};
            E ← E \ {a, c};
    else if p(u, j, v) < min_T then
        min_T ← p(u, j, v);
        V_T ← {u, j, v};
        E_T ← {{u, j}, {j, v}, {u, v}};
        T ← (V_T, E_T, w);
        E ← E ∪ {u, v};
    else
        E ← E ∪ {u, v};
return T;
```



Obrázek 2.1: Příklad případu, kdy algoritmus vybral hranu $\{c, d\}$, odebral ji z množiny hran, a pomocí Dijkstrova algoritmu našel nejkratší cestu z bodu c do bodu d. (Pro čitelnost nejsou zobrazeny váhy ostatních hran. Předpokládejme, že jsou vyšší, než 3.)

2.2 Dijkstrův algoritmus

Dijkstrův algoritmus, pojmenovaný po Edsgeru W. Dijkstrovi, je algoritmus na hledání cesty s minimální váhou mezi dvěma body v ohodnoceném grafu, který hraje velkou roli v našem algoritmu. Obecně taková cesta může mít několik vrcholů, ale protože náš graf je převzatý z roviny, povede právě přes jeden vrchol a bude se skládat ze dvou hran. To se dá dokázat pomocí trojúhelníkové nerovnosti. Kdyby náš graf nebyl převzatý z roviny, mohla nastat situace na obrázku 2.2.



Obrázek 2.2: V obecném grafu nemusí platit trojúhelníková nerovnost.

2.2.1 Popis Dijkstrova algoritmu

1. Vytvoříme množinu všech nenavštívených bodů a vybereme startovní a cílový bod. Označíme všechny body nenavštívenými.

2. Každému bodu přiřadíme vzdálenost od počátečního bodu; prozatím na ∞. Vzdálenost počátečního bodu od sebe samého nastavíme na 0.

3. Nejdříve projdeme všechny sousedy počátečního bodu. Každému sousedu spočítáme vzdálenost od počátečního bodu (v tomto případě váha hrany vedoucí z počátečního bodu do souseda) a připíšeme ji sousedovi, bude-li menší než ∞. Až zkontrolujeme všechny jeho sousedy, odebereme počáteční bod z množiny nenavštívených bodů.

4. Potom se přesuneme na nenavštívený bod s minimální vzdáleností od počátečního bodu. Tento bod označíme jako aktuální a začneme kontrolovat jeho sousedy. Je-li součet vzdálenosti od počátečního bodu do aktuálního s váhou hrany vedoucí k sousedu menší než vzdálenost, kterou má u sebe soused uloženou, změníme ji. Je třeba myslet na to, že když přepíšeme vzdálenost souseda od startovního bodu, souseda neoznačujeme za navštíveného. Počáteční bod odebereme z množiny nenavštívených bodů, až zkontrolujeme všechny jeho sousedy.

5. Čtvrtý bod opakujeme, dokud nevybereme za aktuální bod ten cílový. V tomto okamžiku jsme našli nejkratší cestu.

2.3 Důkaz algoritmu

Abychom mohli dokázat korektnost algoritmu, musíme dokázat, že algoritmus skončí a že je správný, to znamená, že dělá přesně co chceme. Dijkstrův algoritmus, který je součástí algoritmu, dokazovat nebudu, protože důkaz je příliš dlouhý a je dostupný v literatuře.

Tvrzení 1. *Dijkstrův algoritmus je korektní.*

Důkaz. Vizte [BS99, s. 113]. □

Tvrzení 2. *Aloritmus na hledání cyklu délky 3 je korekektní.*

Důkaz. Konečnost algoritmu je zřejmá; jediným cyklem v algoritmu je procházení všech stran. Jelikož je množina hran konečná a víme, že Dijkstrův algoritmus je korektní, můžeme říci, že náš algoritmus skončí.

Správnost algoritmu dokážeme sporem. Budeme předpokládat, že existuje trojúhelník x, y, z, který má kratší obvod, než trojúhelník a, b, c, který našel algoritmus, neboli:

$$\exists\{x, y, z\} \subseteq V : p(x, y, z) < p(a, b, c) \mid \forall a, b, c \leftarrow algoritmus.$$

Znamená to, že náš algoritmus vybral jednu stranu trojúhelníku s minimálním obvodem špatně, a tím vznikl trojúhelník s delším obvodem. Jelikož náš algoritmus prochází všechny hrany, tak tam nemohl vybrat špatnou hranu. Tím pádem špatnou hranu musel vybrat když vybíral zbylé dvě hrany. Ty ale vybral Dijsktrův algoritmus, který je dokázán. Tím vznikl spor a algoritmus je dokázán. □

Kapitola 3

Zobecnění na n dimenzí

Problém v n dimenzích bude trochu komplikovanější a časově náročnější, ale převod na grafovou úlohy nám to velice usnadní. Opět budeme hledat cykly, ale tentokrát délky $n+1$. Práce nám ale velice zkomblikuje kontrolování, jestli daný cyklus je polytop dimenze n. Například ve 3D Může nastat situace, kdy algoritmus najde cyklus délky 4, ale bude se jednat o čtyřúhelník ležící v jedné rovině.

3.1 Algoritmus

Najít všechny cykly v grafu a pak vybrat minimální s dobrou vzdáleností? -> příšerný, ale funguje to…

3.2 Nápady s kombinatorikou

Množina všech cyklů délky n v našem grafu je: $C = \binom{V}{n}$. Teď musíme vyřadit cykly, které nemají maximální dimenzi. Pak vybereme $min(w(c)) \mid c \in C \; \forall c$ a to je cyklus minimální váhy s délkou n. Tento algoritmus je ale neefektivní. Zabere $\#V^n$ času, což je zkoušení všech možností.

Část II

Praktická část

Kapitola 4

Program ve 2D

V této kapitole naprogramuji program, který v množině bodů v rovině najde trojúhelník s minimálním obvodem. K naprogramování takového programu použiju programovací jazyk Python.

Závěr

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Literatura

[BS99] Holger Benl a Helmut Schwichtenberg. „Formal Correctness Proofs of Functional Programs: Dijkstra's Algorithm, a Case Study". In: *Computational Logic*. Ed. Ulrich Berger a Helmut Schwichtenberg. Berlin, Heidelberg: Springer, 1999, s. 113–126. ISBN: 978-3-642-58622-4. DOI: 10.1007/978-3-642-58622-4_4.

[Ada24] Adam Klepáč. *Definice polytopu*. 9. led. 2024.