INOVACE BAKALÁŘSKÝCH A MAGISTERSKÝCH STUDIJNÍCH OBORŮ
NA HORNICKO-GEOLOGICKÉ FAKULTĚ
VYSOKÉ ŠKOLY BÁŇSKÉ - TECHNICKÉ UNIVERZITY OSTRAVA

Algoritmizace prostorových úloh

Grafové úlohy

Daniela Szturcová

evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Tento projekt je spolufinancován Evropským sociálním fondem a státním rozpočtem České republiky.
ESF napomáhá rozvoji lidských zdrojů a podnikatelského ducha.

Graf

Laicky lze graf popsat jako puntíky propojené čarami.

Z matematického hlediska se jedná o strukturu, jejímiž vlastnostmi se zabývá oblast diskrétní matematiky.

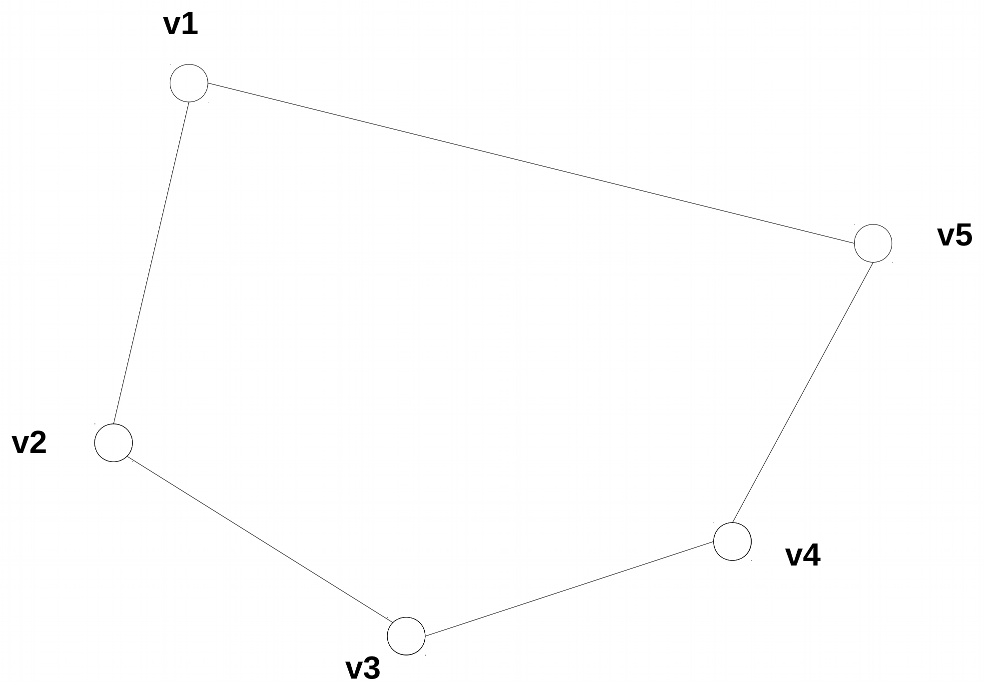
Definice grafu

Graf je uspořádaná dvojice G = (V, E), kde

- V je množina vrcholů, a
- E je množina hran (každou hranu lze též chápat jako podmnožinu - dvojici vrcholů).

Typy grafů

Pokud pracujeme s grafem, jehož uzly ani hrany nemají zvláštní vlastnosti, jedná se o obecný graf.

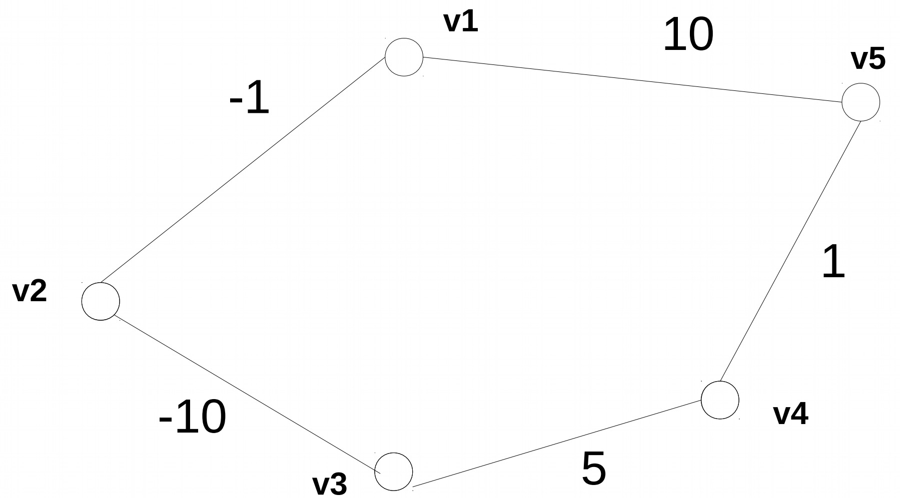


Typy grafů – ohodnocený graf

Má-li každá hrana v grafu přiřazenou nějakou hodnotu, hovoříme o *ohodnoceném grafu*.

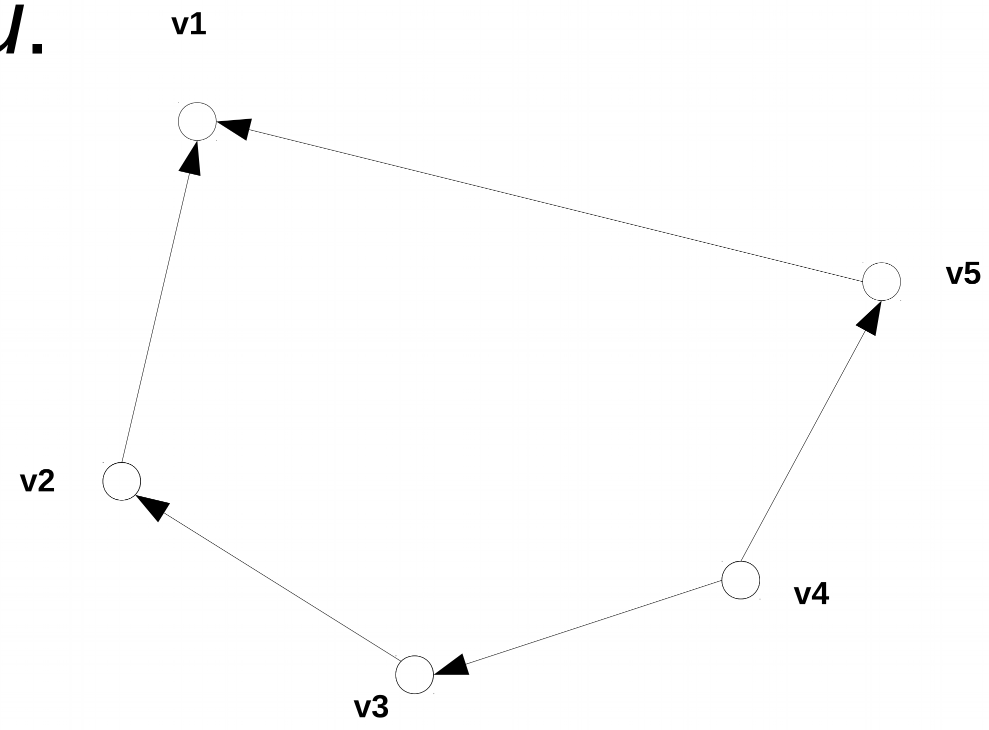
Příklad: Platba mýtného

- Ohodnocení hran kladné – jsem výběrčím mýta,
- Ohodnocení záporné – jsem plátcem mýta.



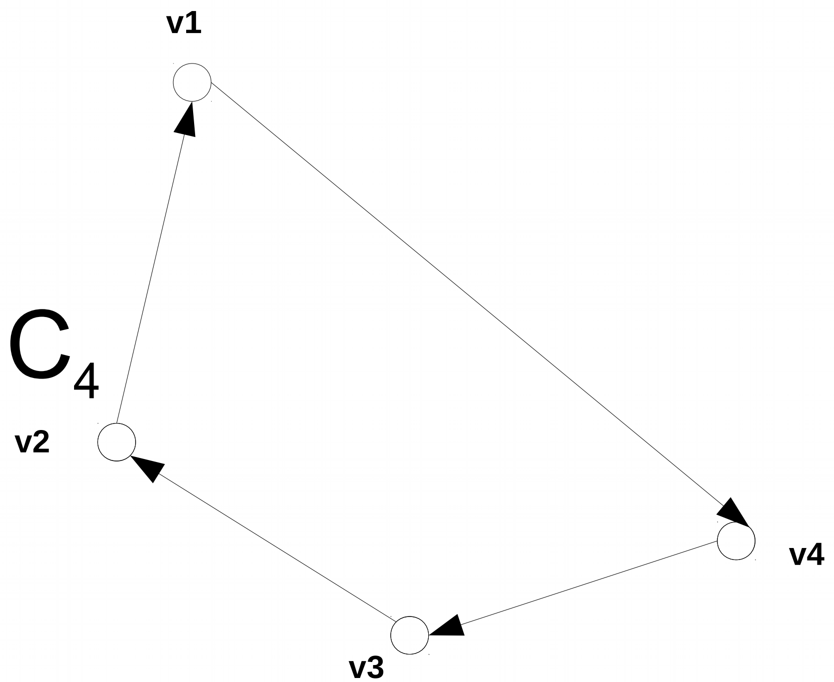
Typy grafů - orientovaný graf

Má-li každá hrana v grafu přiřazen směr, pak hovoříme o *orientovaném grafu*.



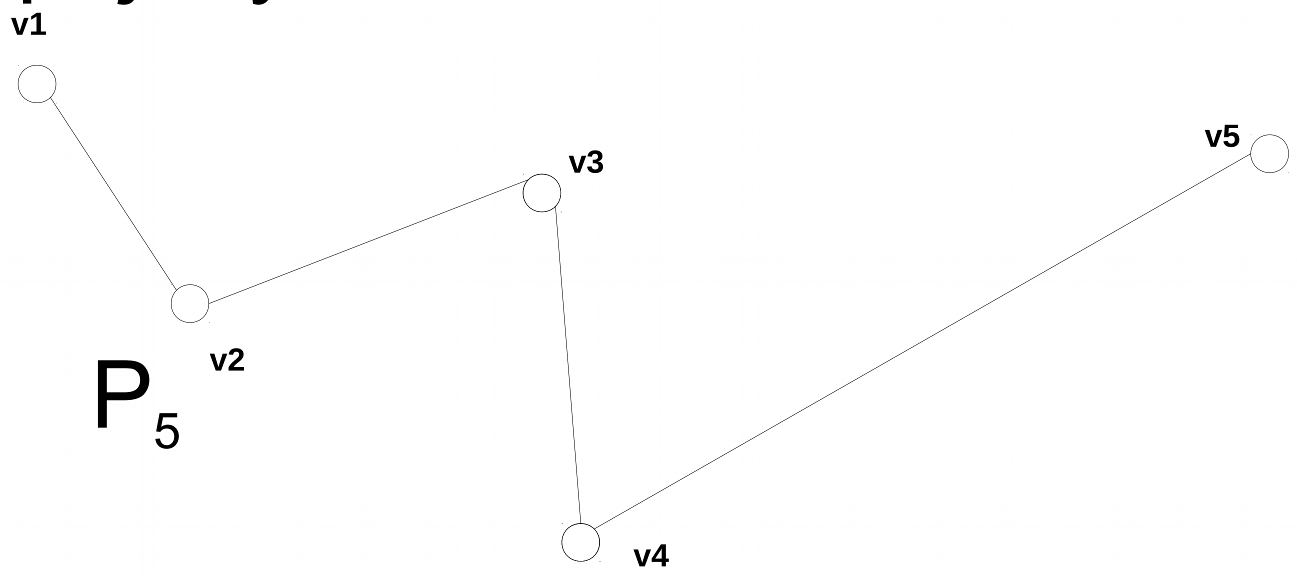
Kružnice (cyklus)

Kružnice(cyklus) délky *n* je tvořena nejméně třemi vrcholy spojenými do jednoho cyklu.



Cesta

Cestou rozumíme graf s *n+1* vrcholy, které jsou postupně za sebou propojeny *n* hranami.

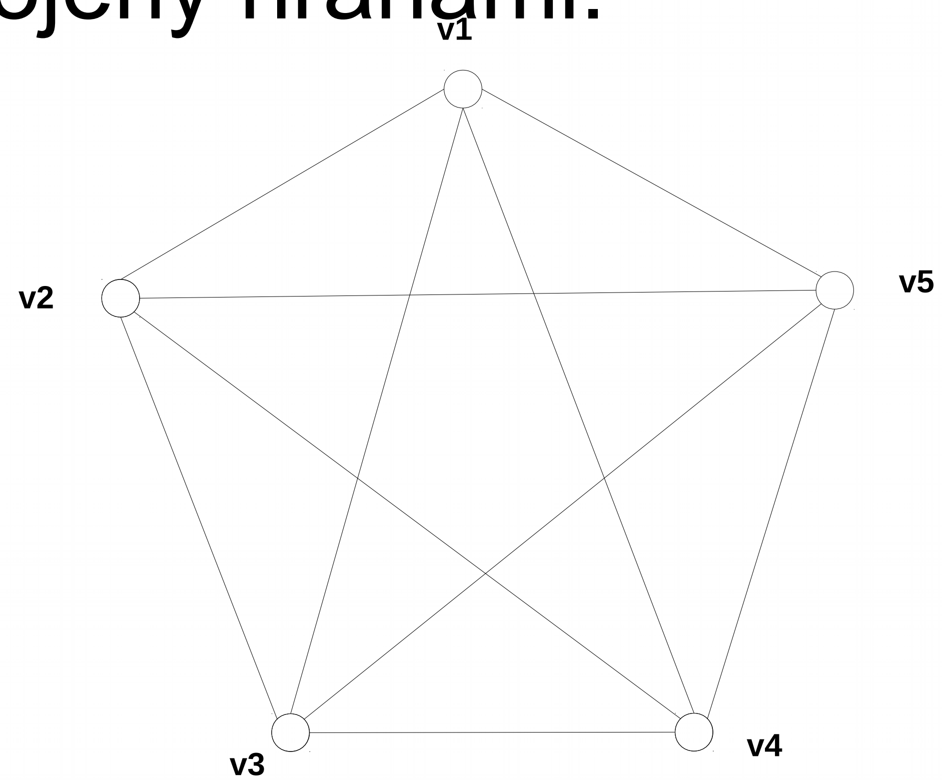


P_5

Úplný graf

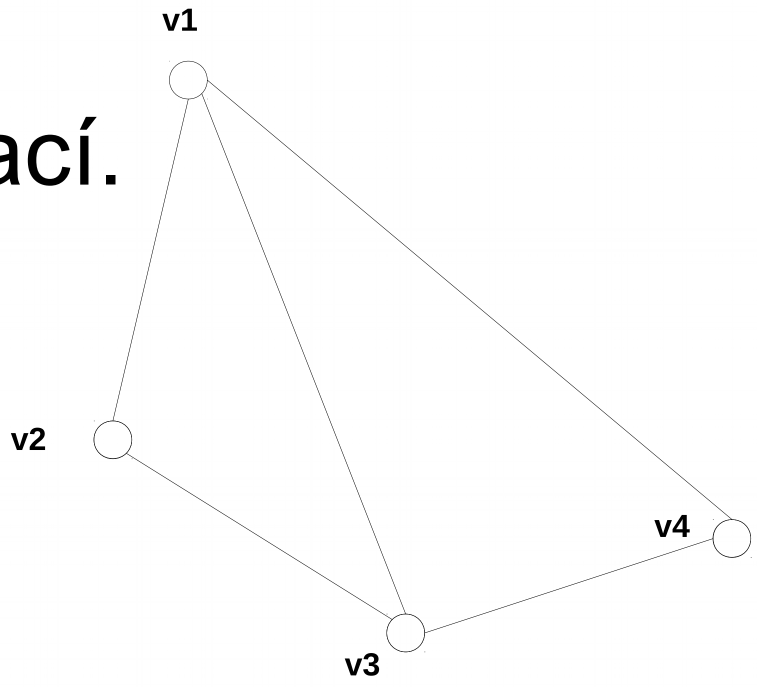
Úplným grafem rozumíme graf s *n* vrcholy, které jsou všechny navzájem propojeny hranami.

K_5



Reprezentace grafu

- Graficky,
- maticí incidence,
- maticí sousednosti,
- spojovou reprezentací.



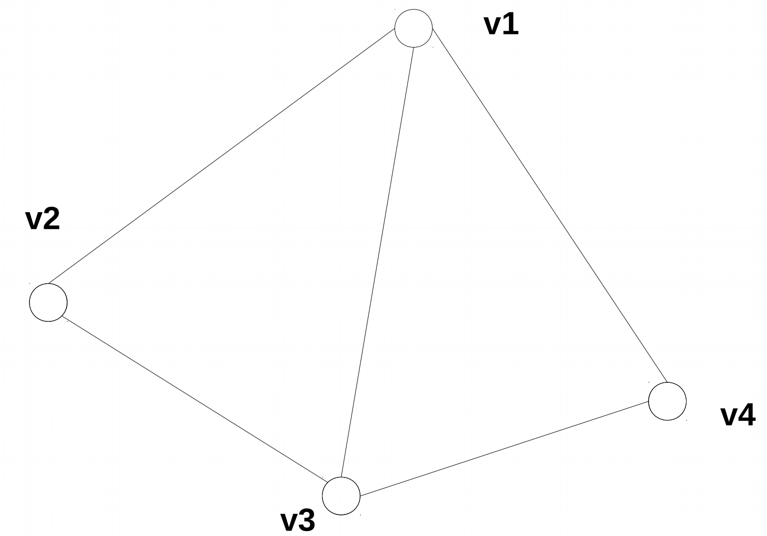
Matice sousednosti

Matice *sousednosti* NxN, řádky i sloupce představují uzly grafu.

Hodnota prvku *Aij* je jednotková právě tehdy když z uzlu *i* vede hrana do uzlu *j.* Jinak je hodnota nulová.

Pro neorietované grafy je matice symetrická podle diagonály.

	v1	v2	v3	v4
v1	0	1	1	0
v2	1	0	1	0
v3	1	1	0	1
v4	0	0	1	0



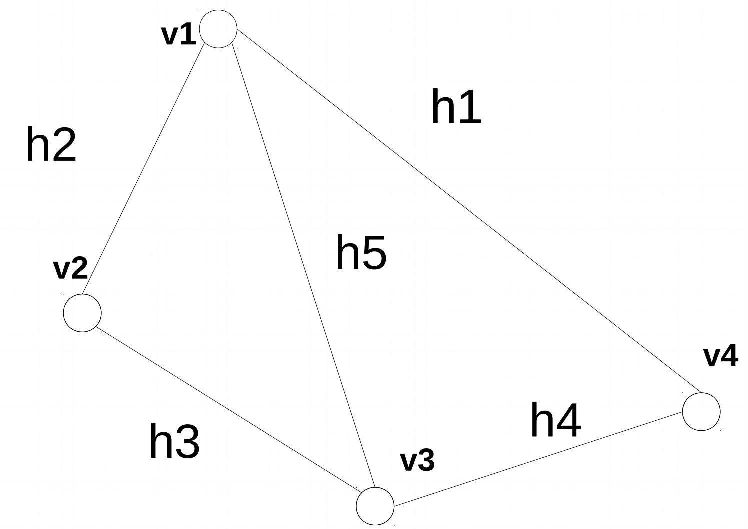
Matice incidence

Matice *incidence NxM,* řádky reprezentují uzly grafu, sloupce hrany.

Hodnota prvku *Aij* je jednotková právě tehdy když je uzel *i* incidenční s hranou *j.* Jinak je hodnota nulová.

U orientovaných grafů lze použít hodnotu 1, pokud je uzel *i* zdrojovým uzlem hrany *j*, -1 pokud cílovým uzlem hrany *j.*

	h1	h2	h3	h4	h5
v1	1	1	0	0	1
v2	0	1	1	0	0
v3	0	0	1	1	1
v4	1	0	0	1	0



Prohledávání grafu

Prohledávání grafu je velmi často používaná úloha v mnoha aplikacích. Můžeme si představit, že z výchozího uzlu se postupně přesouváme po hraně k dalšímu a zkoumáme, zda uzel vyhovuje našemu parametru.

Úloha končí buď nalezením hledaného uzlu, nebo prohledáním celého grafu bez kladného výsledku.

- Prohledávání do šířky (BFS – breadth-first search).
- Prohledávání do hloubky (DFS – depth-first search).

Prohledávání do šířky (BFS)

Postup prohledávání do šířky využívá frontu:

1. Do fronty zařadíme výchozí uzel.
2. Uzel odebereme z fronty a porovnáme jej.
3. Je-li to hledaný uzel,
   a) pak ukončíme prohledávání a vrátíme jej ve výsledku,
   b) jinak zařadíme do fronty dostupné uzly, které ještě nebyly porovnány.
4. Je-li fronta prázdná,
   a) pak byly porovnány všechny uzly. Prohledávání je ukončeno, vracíme výsledek „Hledaný uzel nebyl nalezen“,
   b) jinak opakujeme bod 2.

Prohledávání do hloubky (DFS)

Postup prohledávání do šířky využívá zásobník:

1. Do zásobníku zařadíme výchozí uzel.
2. Uzel odebereme ze zásobníku a porovnáme jej.
3. Je-li to hledaný uzel,
   a) pak ukončíme prohledávání a vrátíme uzel ve výsledku,
   b) jinak zařadíme do zásobníku následné uzly dostupné z tohoto uzlu, dosud neprozkoumané.
4. Je-li zásobník prázdný
   a) pak všechny uzly byly porovnány. Prohledávání je ukončeno, vracíme výsledek „Hledaný uzel nebyl nalezen“,
   b) jinak opakujeme bod 2.

Hledání nejkratší cesty v grafu

Z algoritmů pro hledání nejkratší cesty v grafu si předvedeme

- Dijkstrův algoritmus (1958, Dijkstra)
- A* (postupně vylepšován 1964 – 1968, P. Hart, N. Nilsson, B. Raphael)

Dijkstrův algoritmus nelze použít pro grafy s hranou záporné hodnoty.

A* je závislá na použité heuristické funkci.

Dijkstrův algoritmus

Hledá nejkratší cesty z počátečního uzlu do všech ostatních uzlů ohodnoceného grafu.

- Pro implementaci se využívá *prioritní fronta*.
- Každému uzlu je přiřazena hodnota – součet vzdálenosti od aktuálního uzlu + hodnota hrany vedoucí z aktuálního uzlu do hodnoceného.
- V opakujícím se kroku přehodnocování uzlů se tato hodnota přepočte a do výsledku se vždy zařadí uzel s nejmenší hodnotou.
- Asymptotická složitost O(E+V.log(V))

Dijkstrův algoritmus

1. Vytvoříme pole, kde prvky budou tvořit struktury – uzel, hodnota aktuální vzdálenosti od předchozího uzlu. Výsledek - množina uzlů X s nalezenou nejkratší cestou.
2. Inicializace:
   a) Aktuálním uzlem je počáteční uzel A.
   b) Množina X obsahuje uzel A.
   c) Pro pole bude platit - hodnota uzlu A = 0, pro sousední uzly U hodnota = hodnota hrany (A, U), pro nesousední uzly hodnota = nekonečno.
3. Dokud množina X neobsahuje všechny uzly grafu, opakujeme:
   a) Vybereme uzel V s nejmenší hodnotou.
   b) Uzel V zařadíme do množiny X.
   c) Pro každý uzel U, který sousedí s uzlem V a není prvkem množiny X, přepočteme jeho hodnotu. Pro každý uzel U vybereme minimum [stávající hodnota, hodnota uzlu V + hodnota hrany (U, V)].

Algoritmus A*

Grafový vyhledávací algoritmus – hledá optimální (nejkratší, nejlevnější) cestu mezi dvěma uzly v kladně ohodnoceném grafu.

Vývoj: 1964-1968, P. Hart, N. Nilsson a B. Raphael, pojmenování podle UNIX syntaxe A*

Vstup: ohodnocený graf, počáteční uzel a koncový uzel.

Výstup: nejkratší cesta z počátečního uzlu do koncového, případně neexistence cesty.

Algoritmus A*

- Princip jako prohledávání do šířky, používá se ohodnocení uzlů pomocí speciální funkce f.
- Tato funkce určuje pořadí, v kterém se mají uzly procházet.
- Pro každou cestu *p* je funkce f definovánana takto:
- f(x)=h(x)+g(x) kde
  - f(x) – předpokládaná délka cesty x,
  - h(x) – heuristická funkce pro koncový uzel cesty x (odhad hodnoty cesty z uzlu x k cíli)
  - g(x) – vzdálenost mezi počátečním a daným uzlem x.

Algoritmus A*

- Volí se cesta p, která má v uzlu x hodnotu funkce f(p) nejnižší (tato cesta p má nejvyšší prioritu).
- Předpokladem je, že cesty jsou bez kružnic, pro cílový uzel se v grafu nachází nanejvýš jedna cesta, a to ta nejkratší doposud nalezená.
- Funkce h musí být přípustná (nesmí nadhodnocovat vzdálenost k cíli - lze si představit heuristiku ve formě vzdálenosti vzdušnou čarou, což je v prostoru nejkratší možná cesta).
- Je-li funkce h monotónní, pak je každý uzel navštíven maximálně jednou (nedochází k návratu do již prošlého).

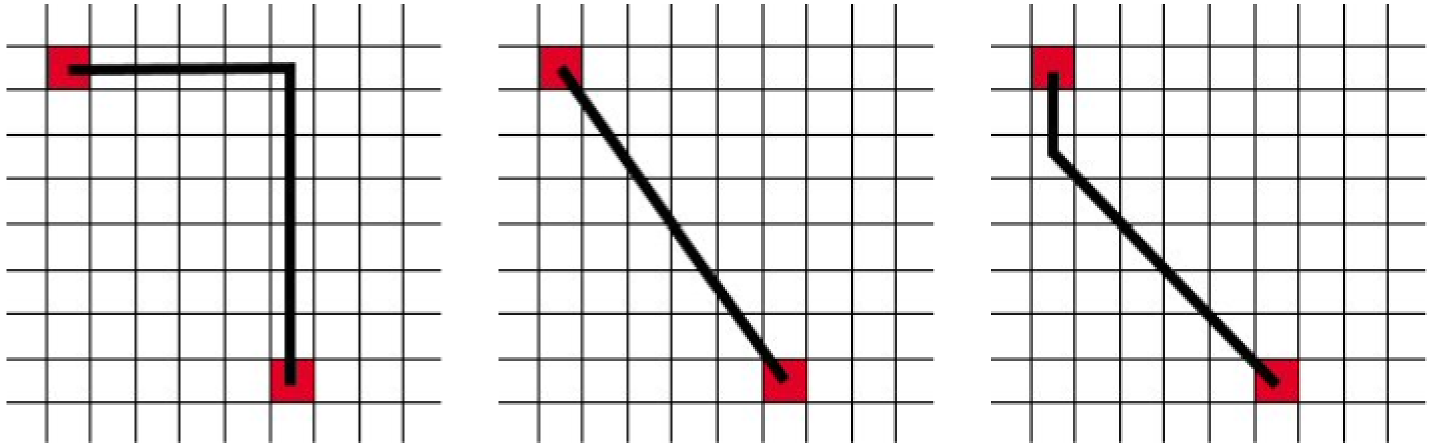
Algoritmus A*

1. Vytvoříme prázdnou množinu cest F.
2. Do množiny F vložíme cestu nulové délky obsahující počáteční uzel s.
3. Dokud není množina F prázdná, opakujeme:
   a) Z množiny F vybereme nejkratší cestu p (s nejnižší hodnotou f(p)) a odebereme ji.
   b) Je-li cesta v cílovém uzlu, vrátíme ji a ukončíme výpočet.
   c) Vytvoříme nové cesty použitím všech možných operátorů na koncový uzel cesty p, které neobsahují smyčky.
   d) Mají-li dvě cesty konec ve stejném uzlu, odstraníme všechny, ponecháme pouze nejkratší (s nejnižší hodnotou f(x)).
   e) Přidáme cestu p do množiny F.
4. Je-li množina F prázdná, pak na výstupu vypíšeme: “Neexistuje cesta z počátečního uzlu s do cílového uzlu.”

Algoritmus A*

Pro výpočty vzdálenosti se používá

- Manhattanská vzdálenost
- Diagonální vzdálenost
- Euklidovská vzdálenost



https://www.cs.auckland.ac.nz/~burkhard/Reports/2003_SS_DanielWichmann2.pdf

Složitost A*

Závisí na odhadu vzdálenosti do cíle.

Složitost

- v nejhorším případě je exponenciální,
- logaritmická pro exaktní odhady vzdálenosti.

Literatura

- Hliněný, P.: Základy Teorie Grafů, FI MU, 2012
- Wichmann, D. R.: Automated route finding on digital terrains https://www.cs.auckland.ac.nz/~burkhard/Reports/2003_SS_DanielWichmann2.pdf
- http://www.algoritmy.net/
- http://voho.cz/wiki/grafovy-algoritmus/
- http://theory.stanford.edu/~amitp/GameProgramming/
- http://modoogle.com/a-star/

evropský sociální fond v ČR

EVROPSKÁ UNIE

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

OP Vzdělávání pro konkurenceschopnost

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ