Globálna navigácia mobilných robotov na báze topologickej mapy

František Duchoň, Ladislav Jurišica

Abstrakt
Článok sa zaoberá analýzou využitia topologických máp pre účely navigácie mobilných robotov. Samotným jadrom článku sú dve metódy z teórie grafov využiteľné pri hľadaní najkratšej cesty v topologických mapách.

Kľúčové slová: topologická mapa, Dijkstrov algoritmus, A* algoritmus

Úvod

Medzi základné typy reprezentácií konfiguračného priestoru, teda reprezentácie prostredia mobilným robotom, možno zaradiť geometrické mapy, bunkovú resp. mriežkovú dekompozíciu (metrické mapy) a topologické mapy (grafová reprezentácia prostredia) [4].

Topologické mapy reprezentujú prostredia štruktúrou grafu, kde vrcholy vyjadrujú jednoznačne určené miesta v prostredí a hrany vyjadrujú prechody medzi týmito vrcholmi. Hrany nebývajú ohodnotené, vyjadrujú len možnosť prechodu medzi vrcholmi. V poslednom období sa však často uplatňujú topologické mapy s metrickým ohodnotením hrán a hybridné mapy, ktoré kombinujú metrické a topologické mapy.

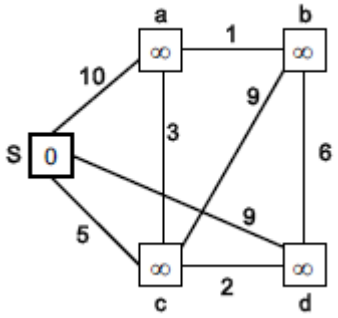
Výhodou využitia topologických máp pri globálnej navigácii mobilných robotov v prostredí je nízka náročnosť na výpočtový výkon a pamäť pri plánovaní cesty, jednoduché plánovanie cesty robota prostredím (možnosť využiť typické grafové metódy) a nenáročné rozširovanie takejto mapy o nové miesta, teda o vrcholy grafu. Medzi najväčšie nevýhody patria neschopnosť reprezentovať metrické vlastnosti prostredia a nutnosť neustále porovnávať a korigovať reálne väzby v mape, teda v grafe. Keďže topologická mapa je grafovou štuktúrou, pri plánovaní cesty v topologickej mape sa uplatňujú najmä algoritmy z teórie grafov ako sú Djikstra alebo A* algoritmus.

1. Dijkstrov algoritmus [1]

Tento algoritmus je schopný nájsť najkratšiu cestu z jedného vrcholu grafu do druhého. Jeho výpočtová náročnosť je $O(h+v^2)$, dá sa zredukovať na $O(h+v*\log v)$, pre h počet hrán a v počet uzlov. Vzdialenosť medzi dvoma vrcholmi je ohodnotená, teda hrany grafu sú pri aplikácii tejto metódy ohodnotené. Metóda vyžaduje, aby boli medzi spojenými uzlami definované pozitívne vzdialenosti. Algoritmus spočíva v týchto krokoch:

1. Inicializuj množinu *Pripravené* - vlož do nej štartovný vrchol.
2. Vyber vrchol s najmenšou vzdialenosťou k množine *Pripravené*, vypočítaj vzdialenosti od tohto vrcholu k všetkým susedom a ulož predchodcu.
3. Pridaj tento uzol do skupiny *Pripravené*.
4. Opakuj kroky 2 a 3 až kým nie sú v množine *Pripravené* všetky vrcholy grafu.

Uvažujme graf z obrázka (Obr. 1). Na začiatku je dosiahnuteľný len vrchol S so vzdialenosťou 0 (definícia). Vzdialenosti do ostatných vrcholov sú v tomto kroku nekonečné a nemajú uloženého predchodcu. Preto je do množiny *Pripravené* v ďalšom kroku vložený vrchol S.

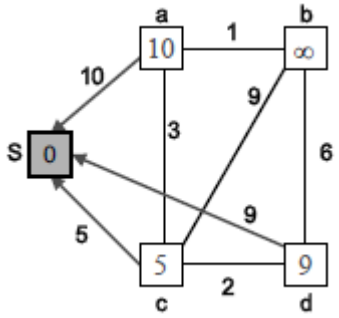


Z S do:	S	a	b	c	d
Vzdialenosť	0	∞	∞	∞	∞
Predchodca	-	-	-	-	-
Pripravené	{}				

Obr. 1 Krok 0 [1]

Fig. 1 Step 0

Vrchol S susedí s vrcholmi a, c a d, sú teda určené vzdialenosti k týmto vrcholom a ich predchodca je určený ako vrchol S (Obr. 2).

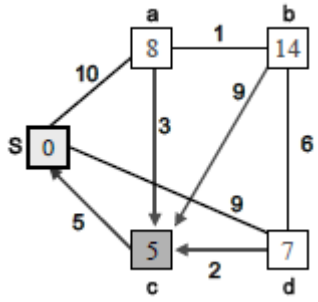


Z S do:	S	a	b	c	d
Vzdialenosť	0	10	∞	5	9
Predchodca	-	S	-	S	S
Pripravené	{S}				

Obr. 2 Krok 1 [1]

Fig. 2 Step 1

V ďalšom kroku je potrebné nájsť najbližší vrchol k vrcholu S a vložiť ho do množiny *Pripravené*. V tomto prípade je to vrchol c. Tabuľka je potom aktualizovaná pre všetkých susedov vrcholu c. Takto je nájdená nová kratšia cesta do vrcholov a a d, pre tieto vrcholy je uložený predchodca c a vrchol c je vložený do množiny *Pripravené* (Obr. 3).

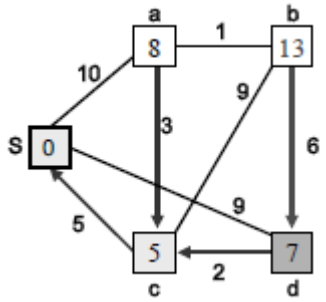


Z S do:	S	a	b	c	d
Vzdialenosť	0	8	14	5	7
Predchodca	-	c	c	S	C
Pripravené	{S, c}				

Obr. 3 Krok 2 [1]

Fig. 3 Step 2

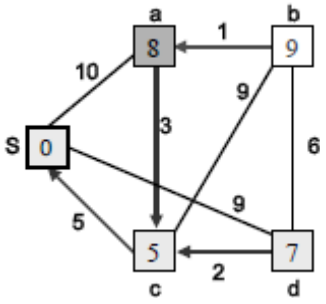
V ďalšom kroku je potrebné opäť nájsť najbližší vrchol k množine *Pripravené* a opakovať postup, kým táto množina neobsahuje všetky vrcholy grafu (Obr. 4 až Obr. 6).



Z S do:	S	a	b	c	d
Vzdialenosť	0	8	13	5	7
Predchodca	-	c	d	S	C
Pripravené	{S, c, d}				

Obr. 4 Krok 3 [1]

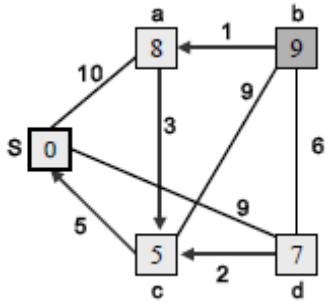
Fig. 4 Step 3



Z S do:	S	a	b	c	d
Vzdialenosť	0	8	9	5	7
Predchodca	-	c	a	S	C
Pripravené	{S, a, c, d}				

Obr. 5 Krok 4 [1]

Fig. 5 Step 4



Z S do:	S	a	b	c	d
Vzdialenosť	0	8	9	5	7
Predchodca	-	c	a	S	C
Pripravené	{S, a, b, c, d}				

Obr. 6 Krok 5 [1]

Fig. 6 Step 5

Najkratšiu cestu zo štartovacieho vrcholu do cieľového vrcholu potom možno skonštruovať spiatočnou propagáciou pomocou predchodcov jednotlivých vrcholov, v prípade cesty z vrcholu S do vrcholu b je to:

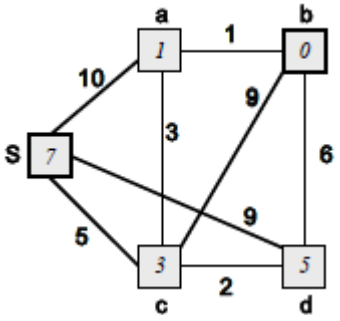
$$pre[b] = a;\ pre[a] = c,\ pre[c] = S$$

Najkratšia cesta z S do b je teda:

$$S \to c \to a \to b$$

2. A* algoritmus [1] [2] [3]

Tento algoritmus hľadá najkratšiu cestu medzi štartovným vrcholom a cieľovým vrcholom, pričom jeho výpočtová náročnosť je $O(k*\log_k v)$, pre v uzlov a pre vetviaci faktor k. Táto metóda vyžaduje znalosť relatívnej vzdialenosti medzi vrcholmi a definíciu euklidovskej vzdialenosti vrcholov od cieľového vrcholu (inak povedané vzdialenosť vzdušnou čiarou). Algoritmus prehľadáva všetky cesty a vyberie tu s minimálnou vzdialenosťou. Táto minimálna vzdialenosť je určená ako súčet ohodnotenej hrany z jedného vrcholu do druhého a euklidovskej vzdialenosti od cieľového vrcholu.



Obr. 7 Príklad pre aplikáciu A* algoritmu [1]
Fig. 7 Example of A* algorithm

Pre príklad z obrázku (Obr. 7) v prvom kroku platí:

$$\{S,a\}=10+1=11$$
$$\{S,c\}=5+3=8$$
$$\{S,d\}=9+5=14$$

Aplikáciou výberu najkratšej cesty vyberáme cestu z vrchola c:

$$\{S,c,a\}=8+3+1=12$$
$$\{S,c,b\}=8+9+0=17$$
$$\{S,c,d\}=8+2+5=15$$

Najkratšia cesta je vo vrchole a, čo vedie na výber cesty do cieľového vrcholu b :

$$\{S,c,a,b\}=12+1+0=13$$

Algoritmus vyzerá zložito vzhľadom na to, že potrebuje ukladať neúplné cesty a ich dĺžky. Avšak aplikáciou rekurzívneho výberu najkratšej cesty nie je potrebné ukladať všetky možnosti výberu cesty. Zavedením kritéria euklidovskej vzdialenosti od cieľového uzla sa významne zmenšuje výpočtová náročnosť celého algoritmu.

Záver

Aj keď aplikácie typických grafových metód na vyhľadávanie najkratšej cesty medzi dvoma vrcholmi aplikovaných na topologické mapy v mobilnej robotike sú výpočtovo jednoduché, treba pripomenúť fakt, že zostrojiť účinnú a dostatočne reprezentatívnu topologickú mapu je veľmi výpočtovo náročné. Preto sa výskum v tejto oblasti zameriava skôr na možnosti tvorby, úpravy a korekcií samotných topologických máp ako na ich vyhodnotenie. Napriek tomuto fakt je potrebné sa oboznámiť s metódami vyhodnotenia týchto máp, lebo už len z takéhoto poznatku vyplýva návrh samotnej tvorby topologickej mapy. Neposledným faktorom ovplyvňujúcim tvorbu samotnej topologickej mapy je samozrejme robot a jeho schopnosti (senzorové vybavenie, dynamika apod.), ako aj prostredie, v ktorom sa pohybuje.

Poďakovanie

Tento článok vznikol pri riešení projektu VEGA 1/0690/09 a KEGA 3/7307/09.

Literatúra

[1] BRÄUNL T.: Embedded Robotics (Mobile Robot Design and Applications with Embedded Systems). Springer-Verlag Berlin Heidelberg, 2006. ISBN-10 3-540-34318-0.

[2] MURPHY R. R.: Introduction to AI Robotics. Massachusetts Institute of Technology, 2000. ISBN 0-262-13383-0.

[3] PATNAIK S.: Robot Cognition and Navigation (An Experiment with Mobile Robots). Springer-Verlag Berlin Heidelberg, 2007. ISSN 1611-2482.

[4] HANZEL, J.: Mapovanie prostredia na báze meraní ultrazvukom. Bratislava, STU v Bratislave FEI, 2007, 138 s.

Abstract

The paper deals with analysis of topological map usage in mobile robots navigation. Core of the paper deals with two methods from graph theory, which can be used for shortest path searching in topological map.

prof. Ing. Ladislav Jurišica, PhD.

Ing. František Duchoň, PhD.

Fakulta elektrotechniky a informatiky
Ústav riadenia a priemyselnej informatiky
Ilkovičova 3
812 19 Bratislava
ladislav.jurisica@stuba.sk, frantisek.duchon@stuba.sk