

Jak nám heuristiky usnadňují řešení problémů?

Jan Bradáč¹, Tereza Kyselová², Alexander Košťál³

¹G Boskovice; bradacjan777@gmail.com

²G a SOŠPg Nová Paka; terezakyselova48@gmail.com

³G Brno, třída Kapitána Jaroše; sascha.kostal@gmail.com

Garant: Ing. Vladimír Jarý, Ph.D. (KSI)

Abstrakt

V tomto článku si představíme pojem heuristika a vysvětlíme, jak nám pomáhá zlepšit různé algoritmy, které poté porovnáme. Přitom použijeme pojmy stavový prostor a inteligentní agent. Nakonec ukážeme algoritmus A^* .

1 Teoretický základ

Nejdříve si zadefinujeme pojmy, na které budeme později odkazovat.

Pod „grafem“ si každý představí něco jiného, my budeme používat graf jako množinu vrcholů V , spojenými hranami E .

Stavový prostor je množina všech možných stavů v uzavřeném systému, také mnoho problémů lze do tohoto prostoru zjednodušit. Při práci s těmito prostory používáme grafy. Pro objasnění si uvedeme příklad: uvažujme levý a pravý pokoj, v pokojích může či nemusí být smetí a v jednom z pokojů se nachází robot. Jednoduchým výpočtem ukážeme, že existuje 8 disjunktivních stavů. $2 \sum_{n=0}^2 \binom{2}{n} = 8$.

Ve stavovém prostoru se může pohybovat tzv. inteligentní agent, který může v systému interagovat s prostředím, což může způsobit přechody mezi stavy. Obecně se používá i mimo stavové prostory, např. umělá inteligence.

Teď si představíme pojem heuristika, což je odhad, který pomáhá snížit část stavového prostoru, který je potřeba prozkoumat. Heuristiky jednoho problému mohou mít více různých forem. My budeme používat heuristiku, která odhaduje vzdálenost k cíli.

2 Možnosti algoritmů

Budeme používat algoritmus, který hledá cestu k cíli ve stavovém prostoru. Algoritmus vždy vybere jedno sousední pole, zjistí, jestli došel k cíli, a pokud ne tak jede dál. Podle toho, jak si algoritmus vybírá, co bude dále prohledávat rozdělujeme algoritmy do několika skupin.

1. UCS

Hledá nejkratší možnou cestu do dalšího bodu, avšak má velkou paměťovou náročnost, protože každou cestu si ukládá.

2. Breadth First Search (prohledávání do šířky)

Graf prochází a při každém rozvětvení jde všemi cestami stejně rychle. Tudíž při najití cíle okamžitě ví nejkratší cestu, protože šel všemi cestami současně. Stejně jako UCS má velkou paměťovou náročnost.

3. Depth First Search (prohledávání do hloubky)

Vybere si jednu cestu a pokračuje po ní dokud nedojde do slepé uličky, nebo neskončí, poté se vrátí k minulému rozcestí a prohledává další cestu. Ne vždy najde nejlepší cestu.

4. Greedy Best First Search (hladový algoritmus)

Algoritmus vyhodnocuje vzdálenost do cíle (což se označuje heuristika). Jedná se o rychlý algoritmus, ale ne vždy najde nejlepší cestu.

5. Algoritmus A*

Jde o kombinaci UCS a hladového algoritmu. Cestu vybírá podle funkce f , která je určena $f = g + h$, kde g je cena cesty ze startu a h je odhad vzdálenosti do cíle. Při správně zvolené heuristice zároveň najde nejlepší cestu.

3 Porovnání algoritmů

Mapu jsme rozdělili na čtverečky tzv. diskretizovali, vzdálenost mezi sousedními políčky je a a políčky, které leží na diagonále, je $a\sqrt{2}$. Pro jednoduchost jsme použili tento výraz: $\lfloor a\sqrt{2} \rfloor$. Pro odhad vzdálenosti h používáme tzv. manhattanskou vzdálenost, která je definována $h = |x_1 - x_2| + |y_1 - y_2|$, kde proměnné označují část souřadnic bodů.

28	24	20	24	28	38	48	58	68	78	88	98	
24	14	10	14	24	34	44	54	64	74	84	94	
20	10	0	10	20	30	40	50	60	70	80	90	
24	14	10	14				54	64	74	84	94	
28	24	20	24			78	68	64	68	78	88	98
38	34	30	34			82	78	74	78	82	92	102
48	44	40	44			92	88	84		92	96	
58	54	50	54				98	94				
68	64	60	64									
78	74	70	74	78	88							
88	84	80	84	88	92							
98	94	90	94	98	102							

Obrázek 1: Do šířky

	100	90	80									
	90	80	70	60								
	80	70	60									
	70	60	50					30	40	50		
70	60	50	40				10	20	30	40	50	
60	50	40	30				0	10		30	40	
70	60	50	40				10	20		40	50	
	70	60	50							50	60	
		70	60	50	40	30	40	50	60	70		
			70	60	50	40	50	60	70			
				70	60	50	60	70				

Obrázek 2: Hladový algoritmus

Na obrázku 1 si můžeme všimnout, že program do šířky nejprve prohledá velkou část mapy a teprve poté na cestu.

		128	114	108	108	108	128					
	114	100	94	94	94	94	114					
128	100	80	80	80	80	80	100	128				
114	94	80	74				94	114				
108	94	80	74		108	88	94	108				
108	94	80	74		102	88	102					
108	94	80	74		102	88	102					
128	114	100	94									
		128	114									

Obrázek 3: A*

Hladový algoritmus sice najde cestu velice rychle, ale překážky ho zmatou a nemusí najít optimální cestu.

A* algoritmus sice jde některými zbytečnými cestami, ale zdaleka ne tak moc jako prohledávání do šířky.

4 Shrnutí

Naprogramovali jsme program, ve kterém jsme porovnávali různé typy algoritmů. Došli jsme k závěru, že program A* si dokáže nejlépe pomoci s překážkami díky heuristikám. Heuristika nám pomáhá tím, že zužuje výběr možností pouze na pravděpodobnější možnosti.

Poděkování

Děkujeme všem organizátorům Týdne vědy na Jaderce za uspořádání této akce a také děkujeme panu Ing. Vladimíru Jarému, Ph.D. za možnost vypracování tohoto miniprojektu a vysvětlení problematiky heuristik.

Odkazy

1. PEARL, J. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. 7th. New York: Addison-Wesley, 1983.
2. TONDA. Teorie grafů. *BRKOS Team*. 2024. Dostupné také z: <https://brkos.math.muni.cz/files/povidani/povidani303.pdf>. [cit. 2024-06-18].