Agents

Environment	CS_GO	Sudoku
Fully/Partially observable	Partially observable	Fully observable
Deterministic/ Stochastic	Strategic	Deterministic
Episodic/ Sequential	Sequential	Sequential
Static/Dynamic /Semi-Dynamic	Dynamic	Static
Discrete/ Continuous	Continuous	Discrete
Single-agent /Multi-agent	Multi-agent	Single-agent

81010408 ابک حسینی‌منش

search

① الف) BFS: چون برای گرفتن جواب بهینه باید هزینه مسیر تا هی غیر نزولی از عمق باشد پس تنها اجرای BFS از گره I و حداکثر تا عمق دو یعنی یکی از رئوس G و F و D و C به جواب بهینه می رسیم به ازای هیچ دو رأس دیگری به پاسخ بهینه نمی رسیم.

DFS: چون این الگوریتم اولین پاسخی که یافت را برمی گرداند پس به احتمال بالایی به پاسخ بهینه نمی رسیم.

اگر هزینه مسیر تا هی از عمق بود، BFS پاسخ بهینه را می داد ولی DFS با هم پاسخ بهینه نمی داد و از طرفی هزینه حافظه DFS در $O(bm)$ بسیار بهتر از هزینه حافظه BFS در $O(b^m)$ است. همچنین کامل بودن BFS و کامل نبودن DFS در tree-search از مزیت های دیگر BFS است.

ب)

رأس	مسیر	هزینه	Explored	Frontier
S	S	0	S	A, C, B
A	SA	5	S, A	C, B, G
B	SB	10	S, A, B	C, G, I, D, F
C	SC	12	S, A, B, C	G, I, D, F, E
I	SBI	12	S, A, B, C, I	G, D, F, E

رأس	مسیر	هزینه	Explored	Frontier
E	SCE	14	S, A, B, C, I, E	G, D, F, H
D	SBD	15	S, A, B, C, I, E, D	G, F, H
F	SBF	17	S, A, B, C, I, E, D, F	G, H
H	SBFH	25	S, A, B, C, I, E, D, F, H	G
G	SBFHG	37	S, A, B, C, I, E, D, F, H, G	

در نتیجه حداقل هزینه برای رسیدن از S به G می شود 37.

رأس	مسیر	هزینه	$f_{(n)}$	Explored	Frontiers (2)
S	S	0	36	S	A, C, B
B	SB	10	35	S, B	A, C, I, D, F
D	SBD	15	30	S, B, D	A, C, I, F, E, H
I	SBI	12	35	S, B, D, I	A, C, F, E, H
F	SBF	17	36	S, B, D, I, F	A, C, E, H, G
H	SBFH	25	35	S, B, D, I, F, H	A, C, E, G
G	SBFHG	37	37	S, B, D, I, F, H, G	A, E, C

الف)

ب) چون به ازای هر رأس $h_{(n)}$ آن کوچکتر مساوی هزینه واقعی رأس تا گره ی است پس هیوریستیک Admissible است

این هیوریستیک Consistent نیست به دلیل وجود مثال نقض $h_{(S)} - h_{(B)} = 36 - 25 = 11 \not\geq C_{(SB)} = 10$. S B

(3) الف) زمان هایی که حافظه مورد استفاده اهمیت دارد یا گرفتن یک پاسخ نسبتاً مناسب در زمان و حافظه کمتر از سایر الگوریتم ها می توانیم از local search استفاده کنیم. همچنین وقت هایی که امکان planing وجود نداشته باشد هم از local search استفاده می کنیم.

ب) یکی از روش ها simulated annealing است. در این روش احتمال غیر صفری به رفتن به پاسخ های بدتر می دهیم به امید این که بتوانیم در نهایت به پاسخ بهتری برسیم. در این روش با افزایش پاسخ های بررسی شده احتمال رفتن به پاسخ های نامناسب را کاهش می دهیم. روش دیگر می تواند الگوریتم جست‌وجوی ممنوعه باشد. در این الگوریتم بهترین حرکت را انجام می دهیم البته در صورتی که آن پاسخ در لیست ممنوعه نباشد و سپس از هر حرکت، آن پاسخ را در لیست ممنوعه ذخیره می کنیم.