هوش مصنوعی

(جستجو-بخش سوم)

صادق اسکندری - دانشکده علوم ریاضی، گروه علوم کامپیوتر

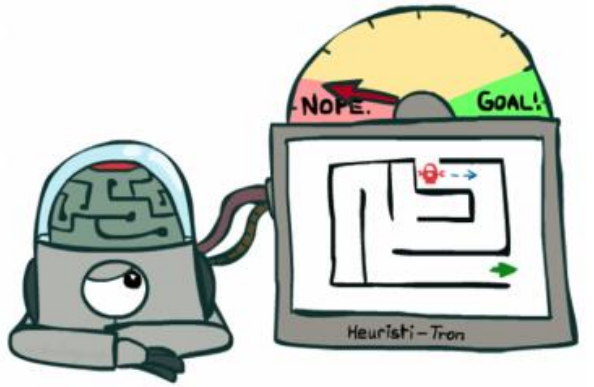
eskandari@guilan.ac.ir

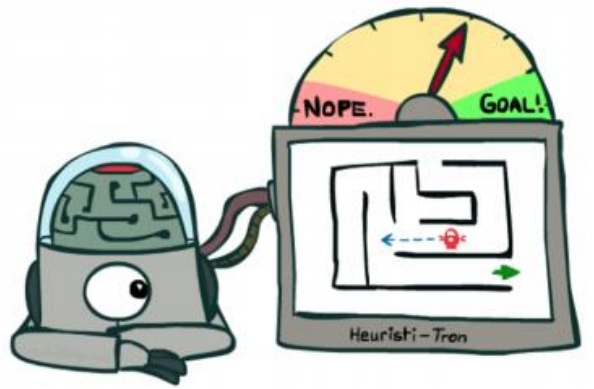
یادآوری ...

استراتژی	نوع	ترتیب بسط درخت	ترتیب گرهها در صف	کامل بودن	بهینگی	پیچیدگی زمانی	پیچیدگی فضایی
عمق اول (DFS)	کوکورانه (ناآگاهانه)	عمیق ترین گره ابتدا بسط داده می شود.	گرهها در صف بر اساس عمق به شکل نزولی مرتب هستند. فرزندان جدید به ابتدای صف اضافه می شوند.	تنها در صورتی که حداکثر عمق محدود باشد.	✗	$O(b^m)$ M: حداکثر عمق درخت	$O(bm)$
سطح اول (BFS)	کوکورانه (ناآگاهانه)	سطحی ترین گره ابتدا بسط داده می شود.	گرهها در صف بر اساس عمق به شکل صعودی مرتب هستند. فرزندان جدید به انتهای صف اضافه می شوند.	✓	در صورتی که هزینه مسیر یک تابع صعودی یکنواخت از عمق باشد،	$O(b^s)$ S: عمق سطحی ترین پاسخ	$O(b^s)$
هزینه یکنواخت (UCS)	کوکورانه (ناآگاهانه)	ارزان ترین گره ابتدا بسط داده می شود.	گرهها در صف بر اساس هزینه مسیر به شکل صعودی مرتب هستند.	✓	✓	$O(b^{\frac{C^*}{\epsilon}})$ $\frac{C^*}{\epsilon}$: عمق موثر	$O(b^{\frac{C^*}{\epsilon}})$
عمیق سازی تکراری (IDS)	کوکورانه (ناآگاهانه)	عمیق ترین گره ابتدا بسط داده می شود (ترکیبی از سطح اول و عمق اول)	گرهها در صف بر اساس عمق به شکل نزولی مرتب هستند. فرزندان جدید به ابتدای صف اضافه می شوند.	✓	در صورتی که هزینه مسیر یک تابع صعودی یکنواخت از عمق باشد،	$O(b^s)$ S: عمق سطحی ترین پاسخ	$O(bs)$
حریصانه	آگاهانه	همواره گرهی بسط داده می شود که کمترین فاصله تخمینی را با هدف داشته باشد.	فرزندان تولید شده از بسط گرهها، به گونه ای به صف اضافه می شوند که صف از نظر تابع ارزیاب (h) مرتب باشد.	✗	✗	در بدترین حالت مشابه DFS	در بدترین حالت مشابه DFS

یادآوری ...

یک *تابع ابتکاری* (Heuristic Function) میزان نزدیکی یک حالت به حالت هدف را **تخمین** می زند.

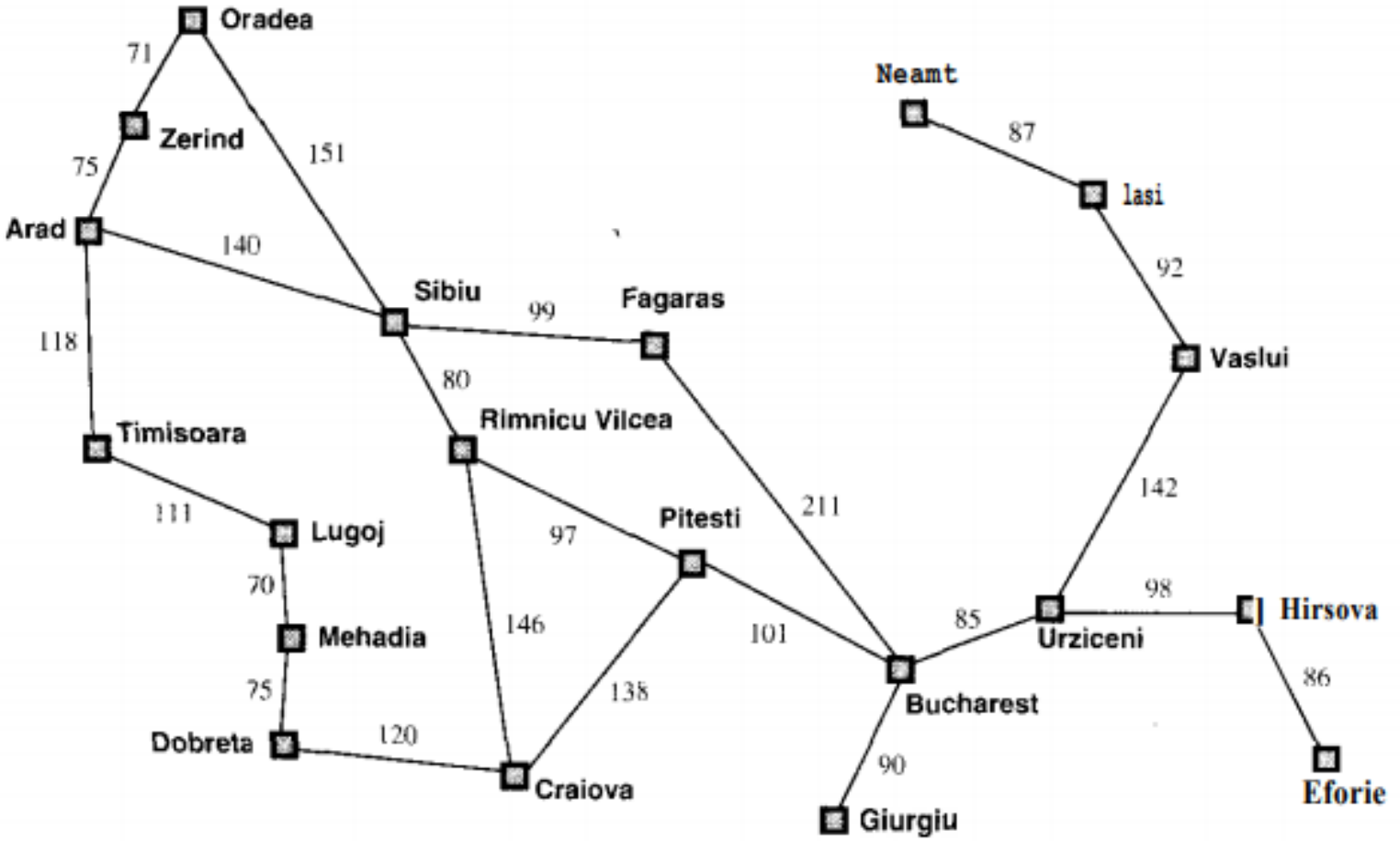




یک تابع ابتکاری کلی وجود ندارد و برای هر مسئله باید به شکل جدا طراحی شود.

یادآوری ...

مثال: تابع ابتکاری برای مسئله مسافرت در رومانی



Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

h(x)

الگوریتم جستجوی A*

این استراتژی، ترکیبی از استراتژی های جستجوی هزینه یکنواخت و حریصانه است.

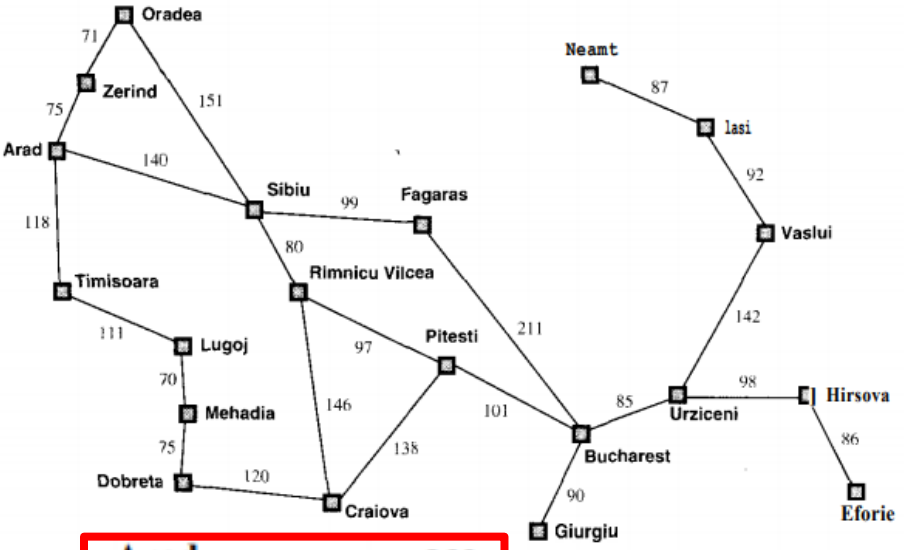
- $g(n)$: هزینه مسیر از ریشه تا گره n
- $h(n)$: هزینه تخمینی مسیر از گره n تا گره هدف

در این استراتژی، فرزندان تولید شده از بسط گرهها، به گونه ای به صف اضافه می شوند که صف از نظر تابع ارزیاب زیر مرتب باشد:

$$f(n) = g(n) + h(n)$$

استراتژی های جستجوی آگاهانه: جستجوی حریصانه

مثال: تابع ابتکاری برای مسئله مسافرت در رومانی



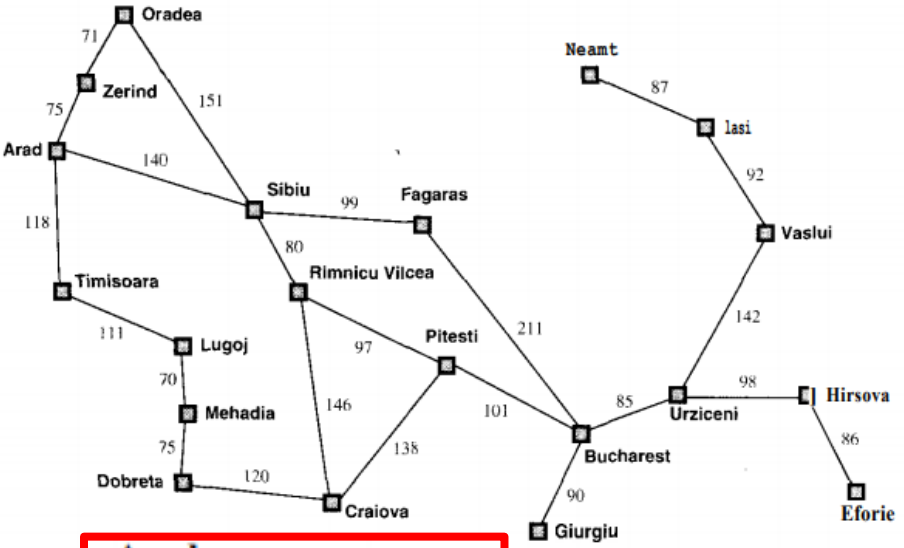
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Arad

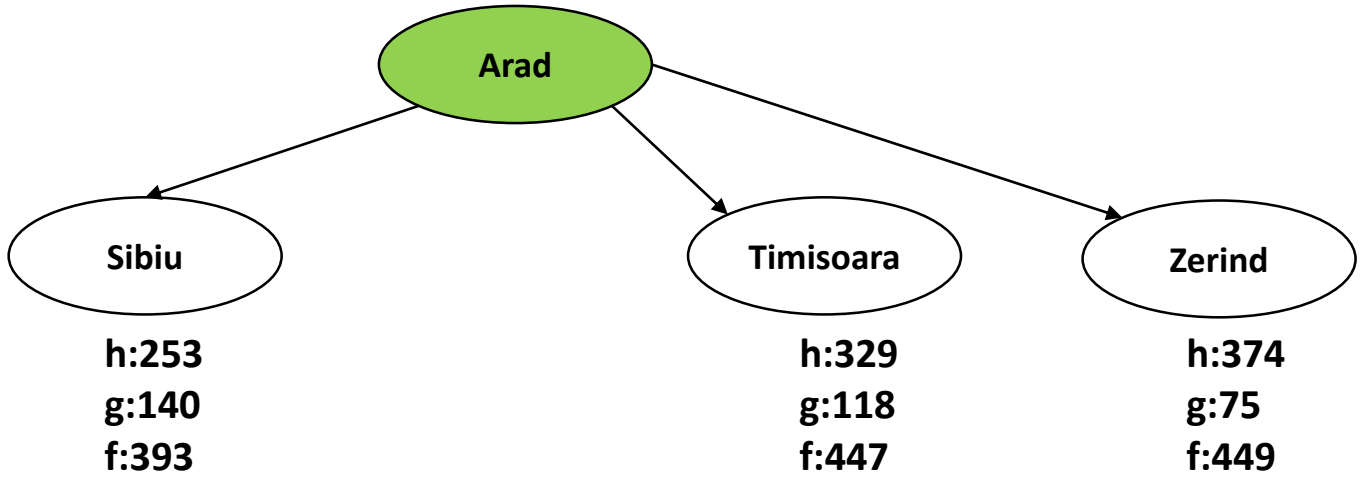
h:366
g:0
f:366

استراتژی های جستجوی آگاهانه: جستجوی حریصانه

مثال: تابع ابتکاری برای مسئله مسافرت در رومانی

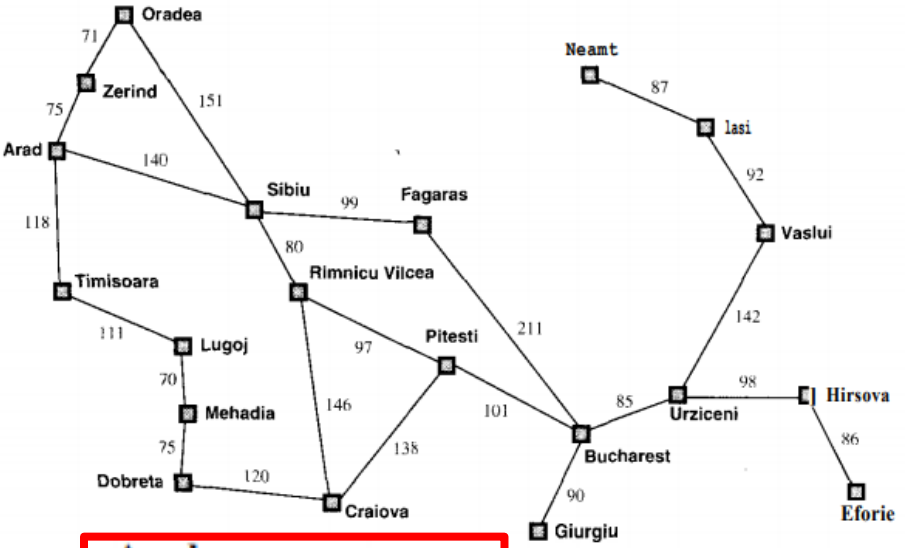


Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

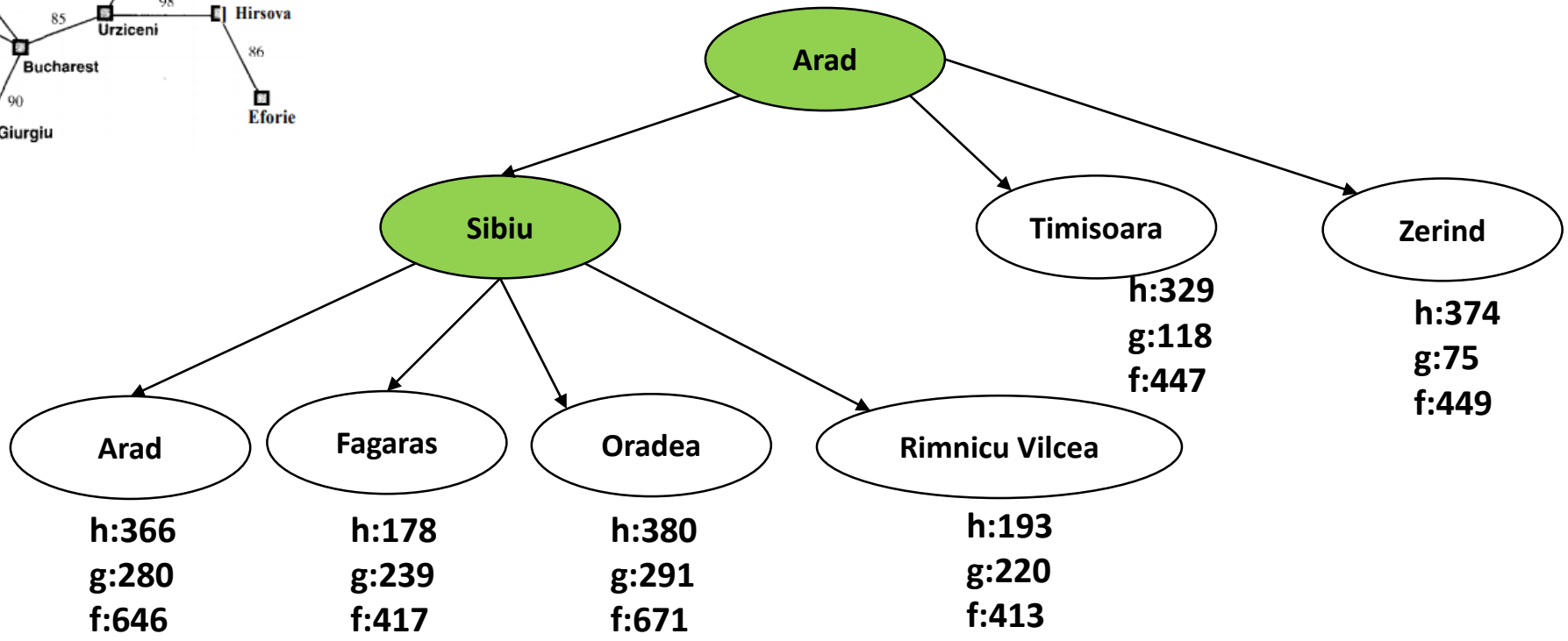


استراتژی های جستجوی آگاهانه: جستجوی حریصانه

مثال: تابع ابتکاری برای مسئله مسافرت در رومانی

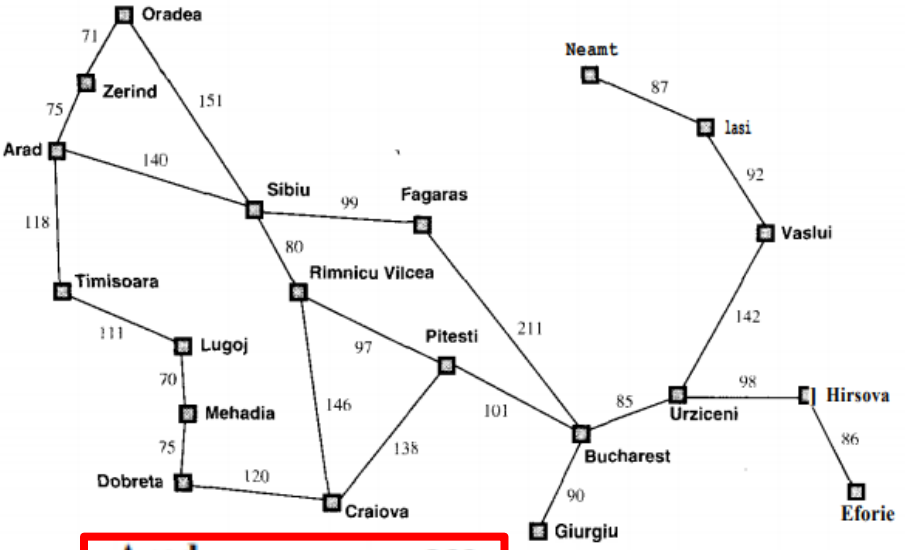


Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

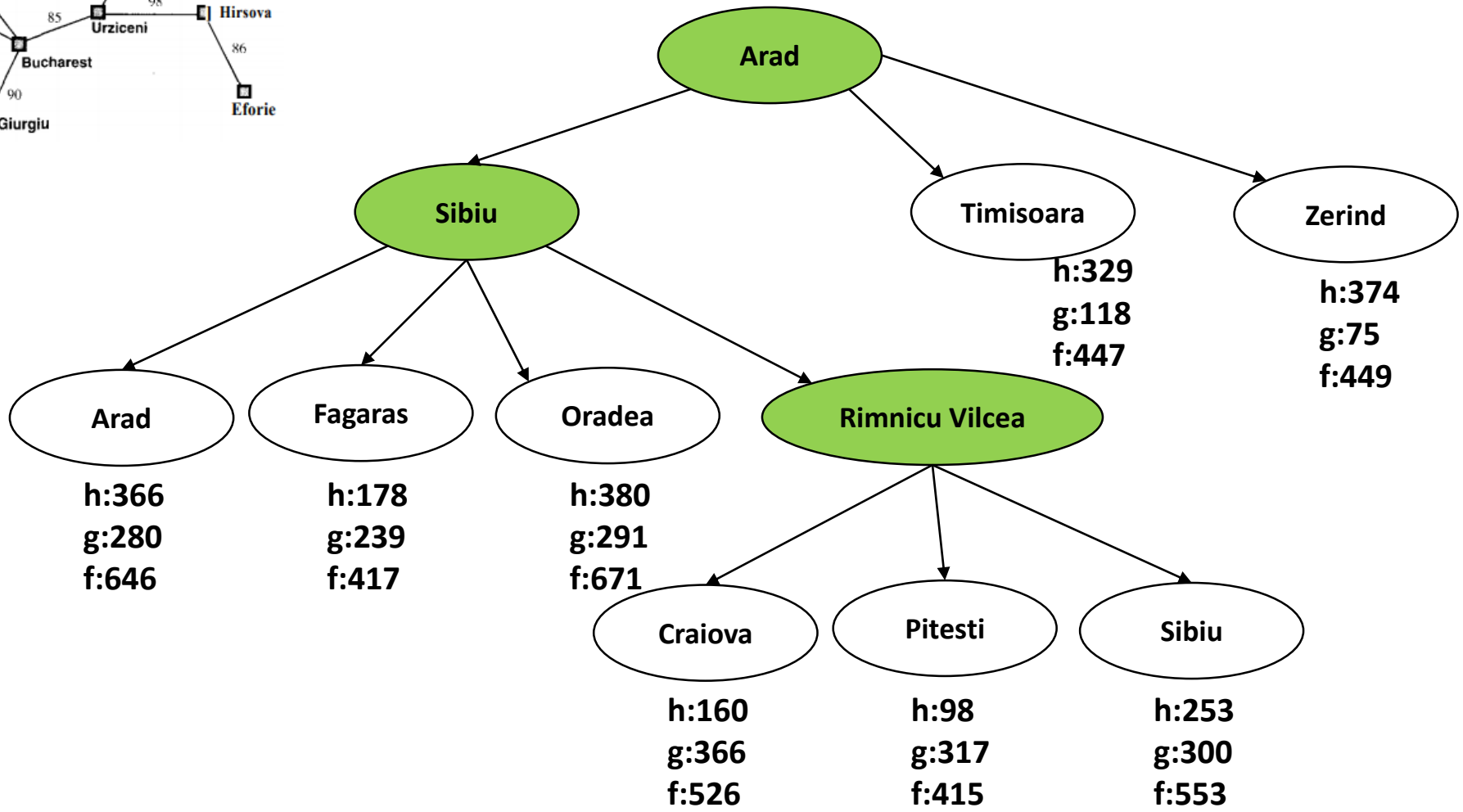


استراتژی های جستجوی آگاهانه: جستجوی حریصانه

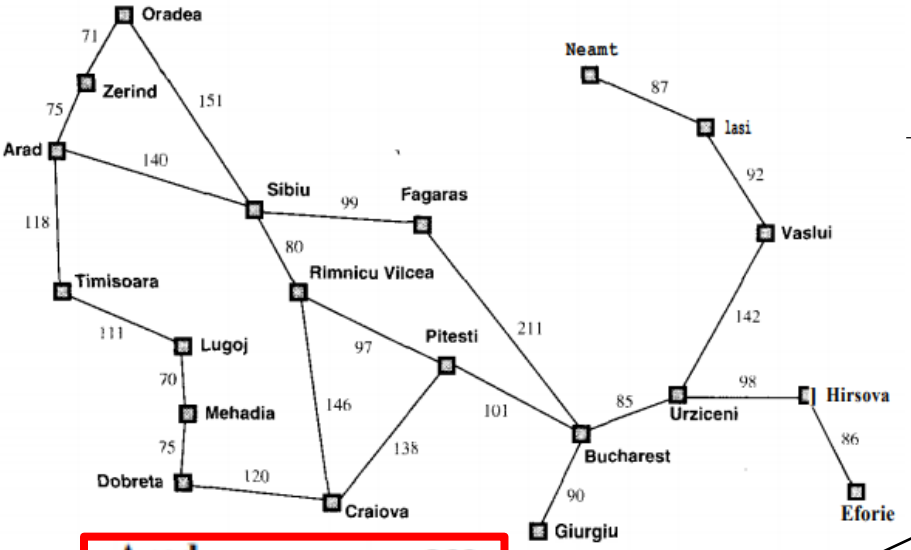
مثال: تابع ابتکاری برای مسئله مسافرت در رومانی



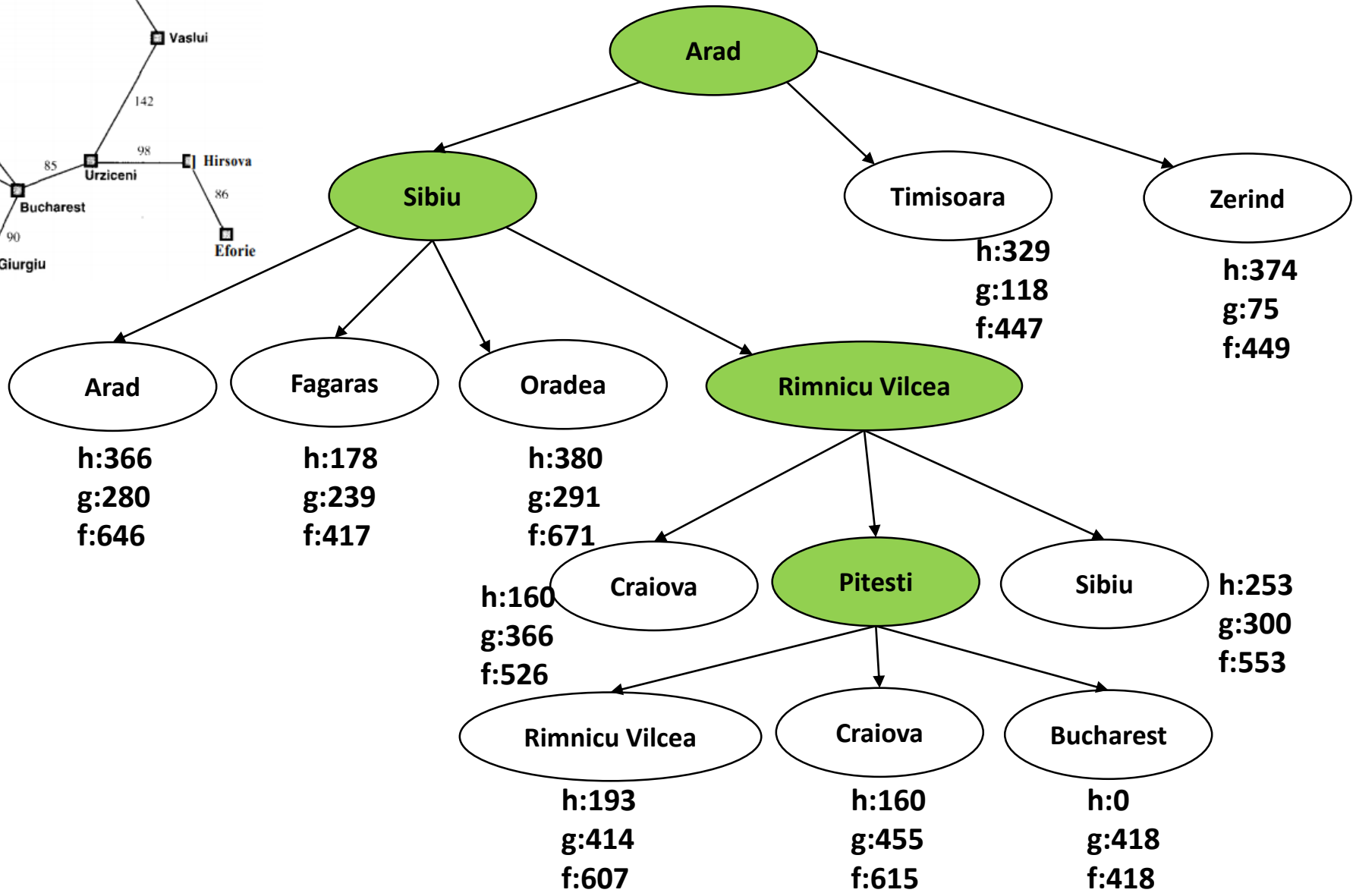
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374



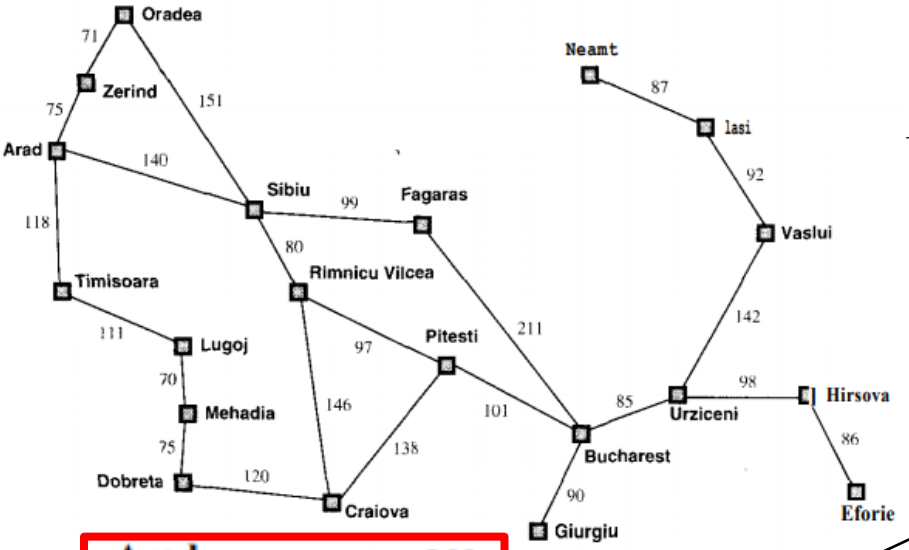
استراتژی های جستجوی آگاهانه: جستجوی حریصانه



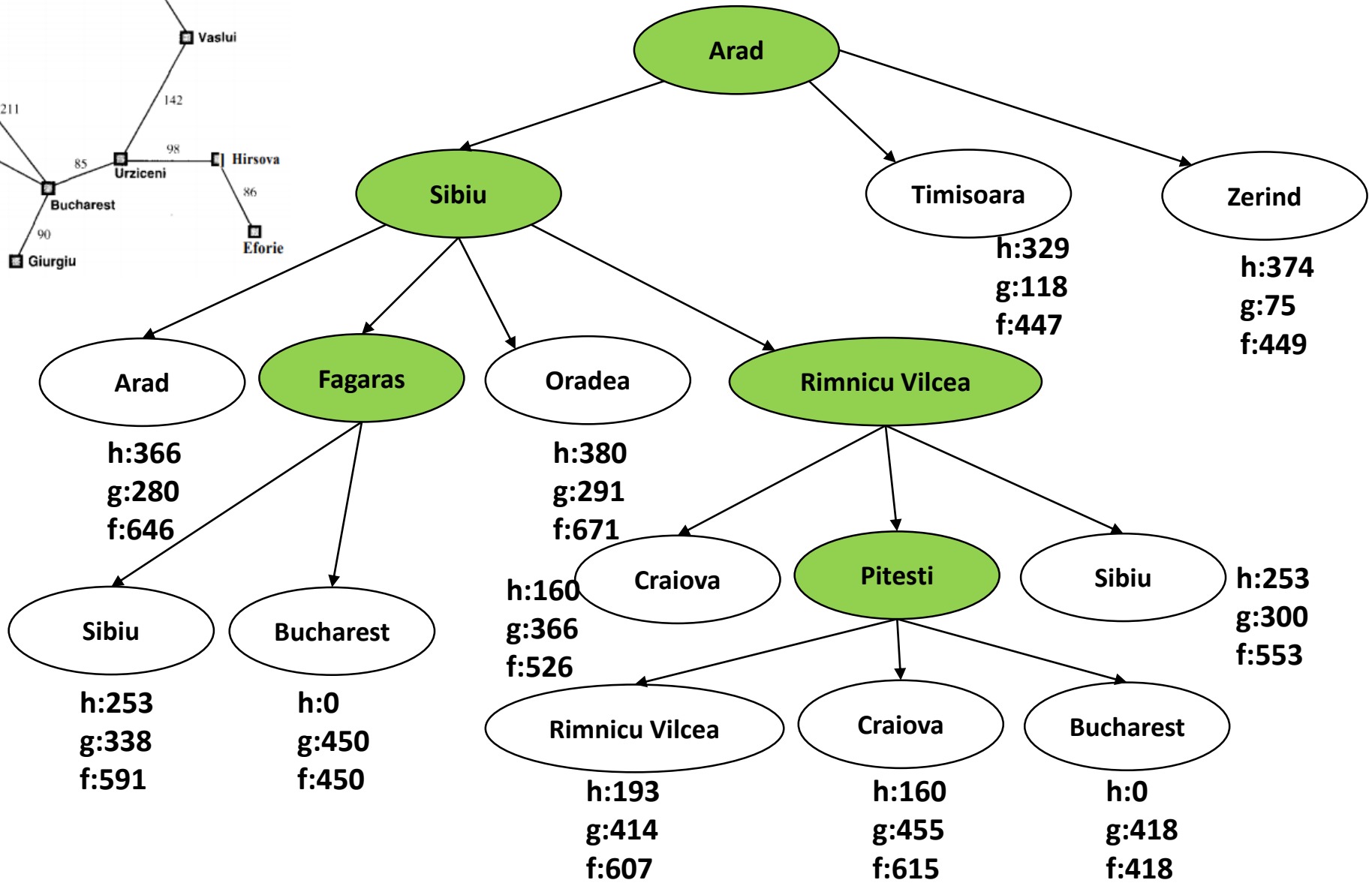
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374



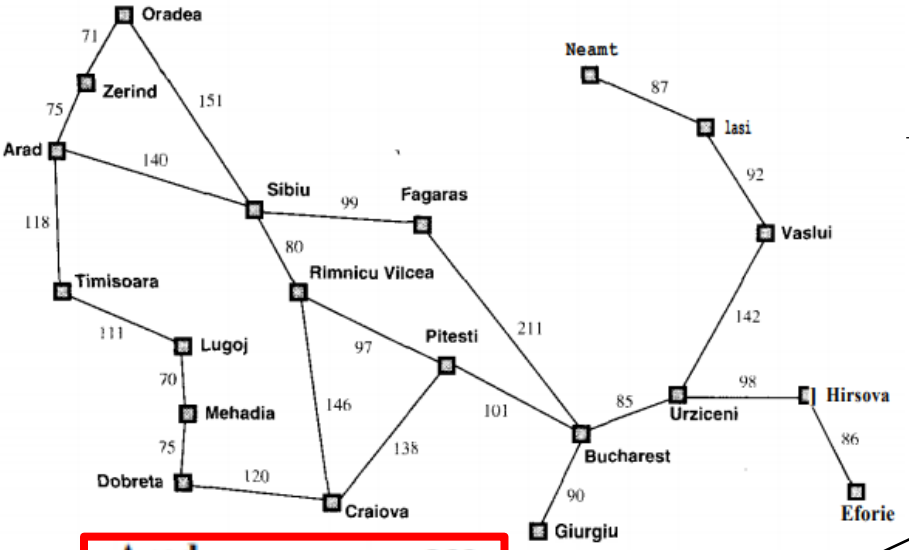
استراتژی های جستجوی آگاهانه: جستجوی حریصانه



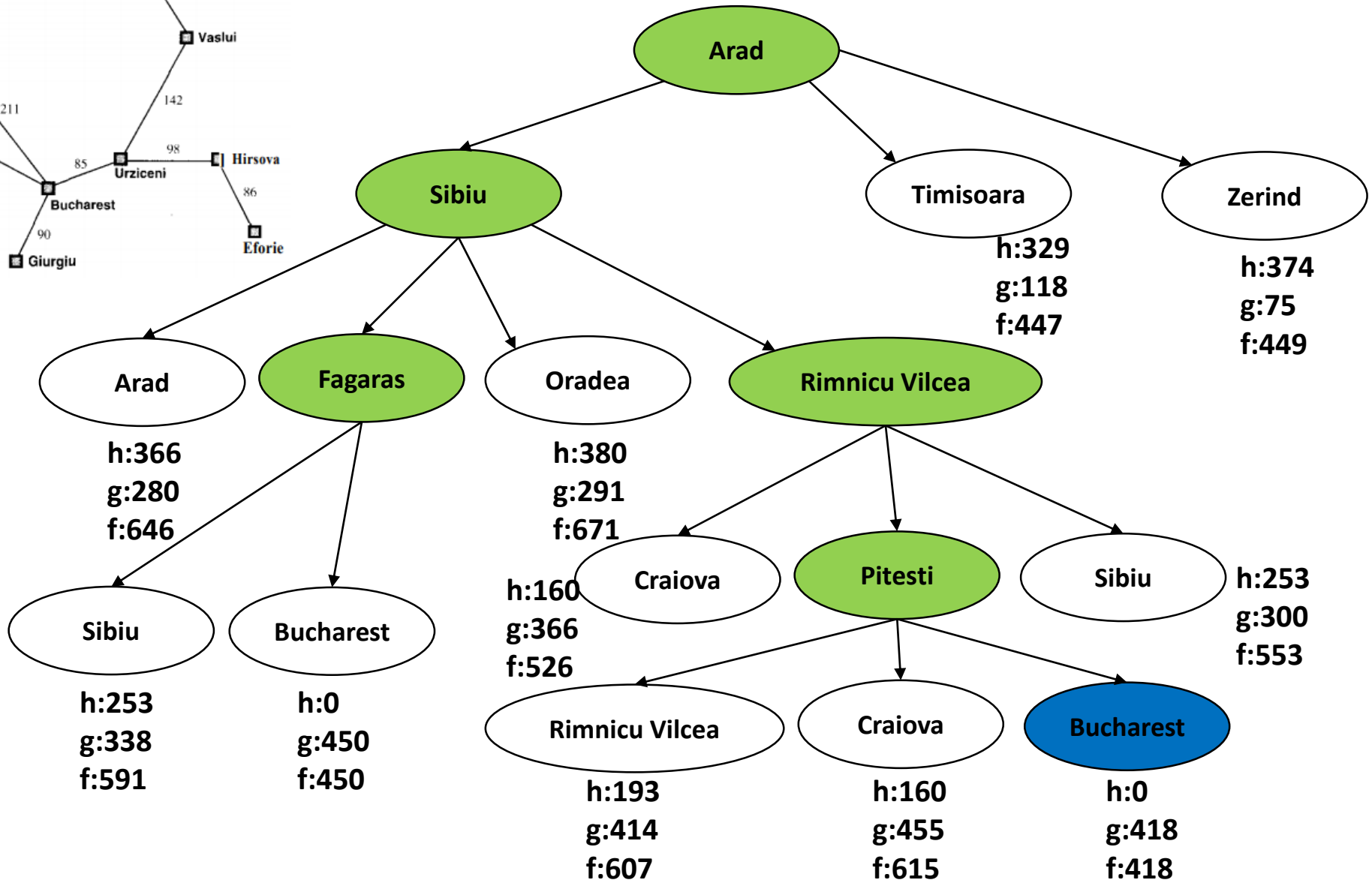
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374



استراتژی های جستجوی آگاهانه: جستجوی حریصانه



Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374



توابع ابتکاری تصدیق پذیر

تابع ابتکاری h را تصدیق پذیر (Admissible) می گوییم اگر هیچگاه از مقدار واقعی بزرگتر نباشد.

به عبارت دیگر، با فرض اینکه $h^*(n)$ نشان دهنده هزینه واقعی از گره n به گره هدف باشد، آنگاه تابع h تصدیق پذیر است اگر:

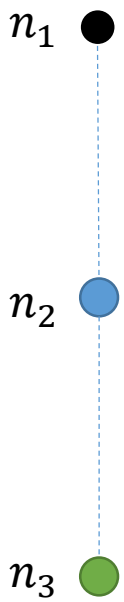
$$0 \leq h(n) \leq h^*(n)$$

مثال: فاصله مستقیم در مسئله مسافرت در رومانی

نکته: یافتن یک تابع ابتکاری تصدیق پذیر، مهمترین بخش از الگوریتم جستجوی A* است.

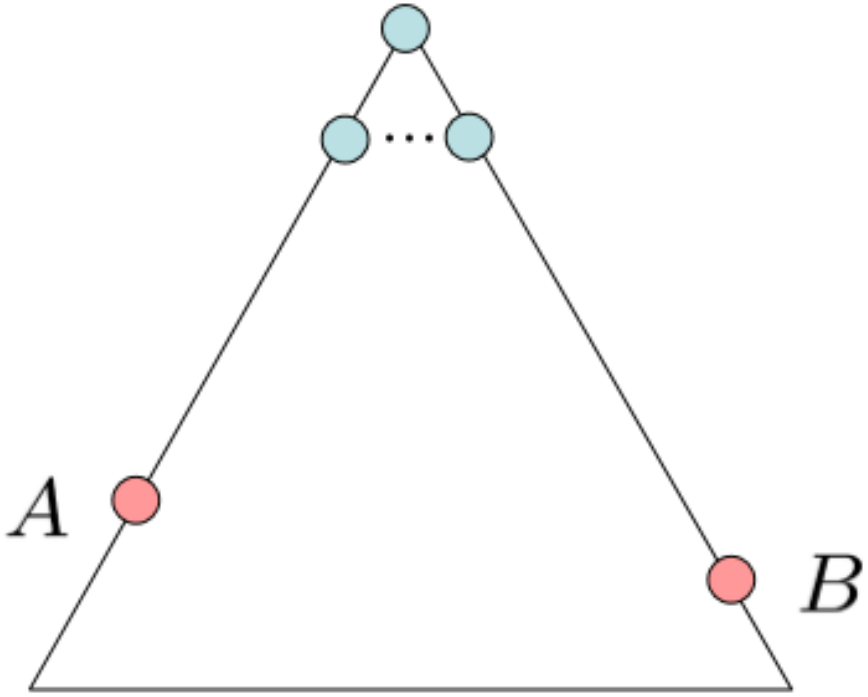
توابع ابتکاری تصدیق پذیر

اگر تابع h تصدیق پذیر باشد، آنگاه $f(n) = g(n) + h(n)$ در طول مسیر از ریشه تا هدف، یکنوا (monotonic) خواهد بود.



$$f(n_1) \leq f(n_2) \leq f(n_3)$$

بهینگی الگوریتم A*



فرض:

A یک گره هدف بهینه است.

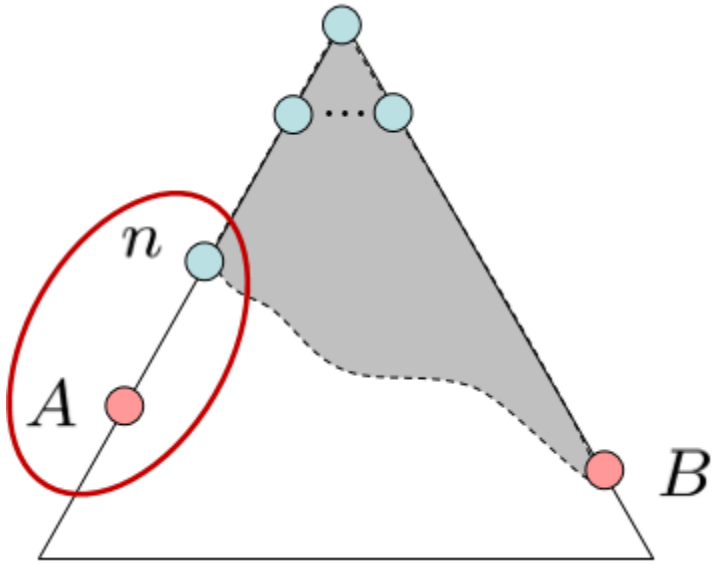
B یک گره هدف غیربهینه است.

h یک تابع تصدیق پذیر است.

ادعا:

گره A قبل از گره B از صف خارج خواهد شد.

بهینگی الگوریتم A*



اثبات:

فرض کنید B در صف است.

علاوه براین فرض کنید یکی از اجداد A نیز در صف است. این گره را n می نامیم. (n می تواند خود A باشد)

ادعا:

n قبل از B بسط داده خواهد شد (از صف بیرون خواهد آمد)

۱) با توجه به اینکه تابع f صعودی یکنوا است، داریم $f(n) \leq f(A)$

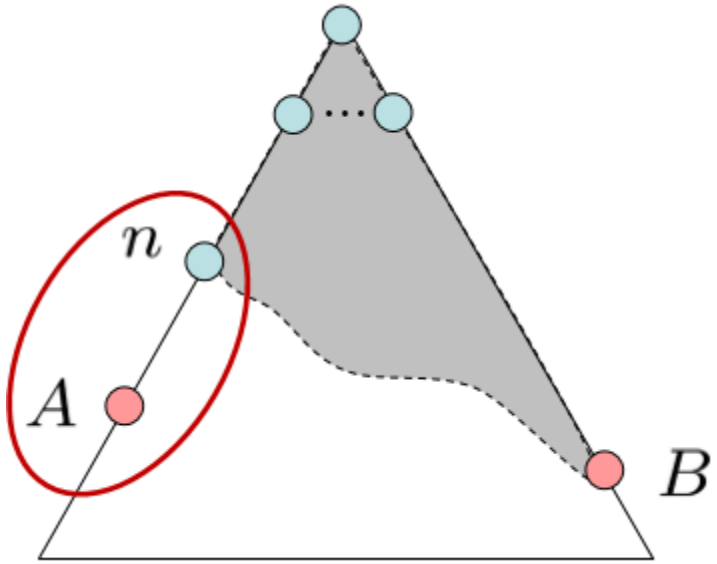
۲) با توجه به اینکه A گره هدف بهینه است، داریم $f(A) \leq f(B)$

با توجه به اینکه A بهینه است، هزینه مسیر آن کمتر است. $g(A) \leq g(B)$

زیرا هر دو گره، هدف هستند. $h(A) = h(B) = 0$

$f(A) = g(A), f(B) = g(B)$

بهینگی الگوریتم A*



اثبات:

فرض کنید B در صف است.

علاوه براین فرض کنید یکی از اجداد A نیز در صف است. این گره را n می نامیم. (n می تواند خود A باشد)

ادعا:

n قبل از B بسط داده خواهد شد (از صف بیرون خواهد آمد)

۱) با توجه به اینکه تابع f صعودی یکنوا است، داریم $f(n) \leq f(A)$

۲) با توجه به اینکه A گره هدف بهینه است، داریم $f(A) \leq f(B)$

۳) در نتیجه $f(n) \leq f(B)$ و بنابراین گره n قبل از B بسط داده خواهد شد.

بنابراین تمامی اجداد A قبل از B بسط داده خواهند شد.

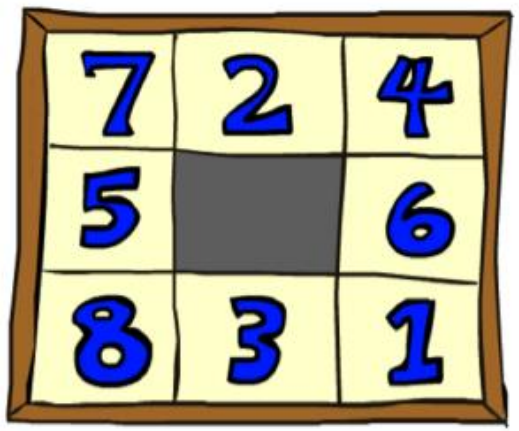
A نیز قبل از B بسط داده خواهد شد.

نتیجه: الگوریتم A* یک الگوریتم بهینه است.

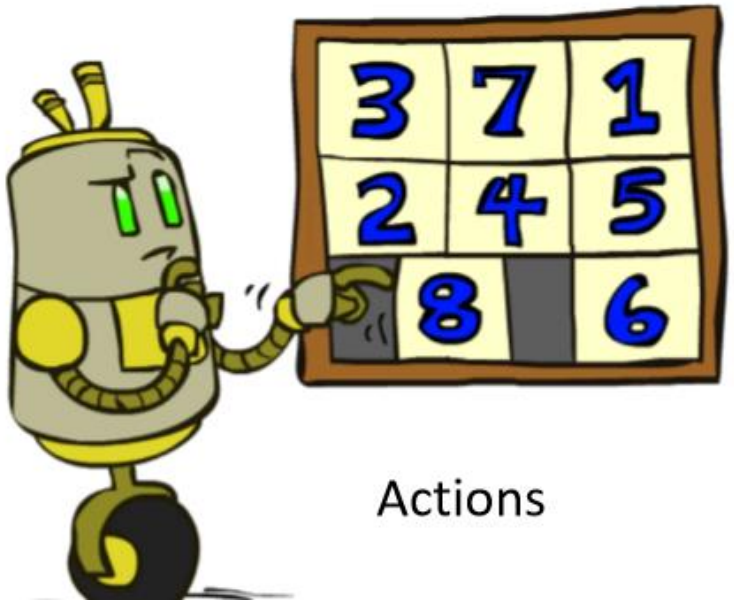
طراحی توابع ابتکاری تصدیق پذیر

طراحی توابع ابتکاری تصدیق پذیر، مهمترین بخش در استفاده از الگوریتم A* می باشد.

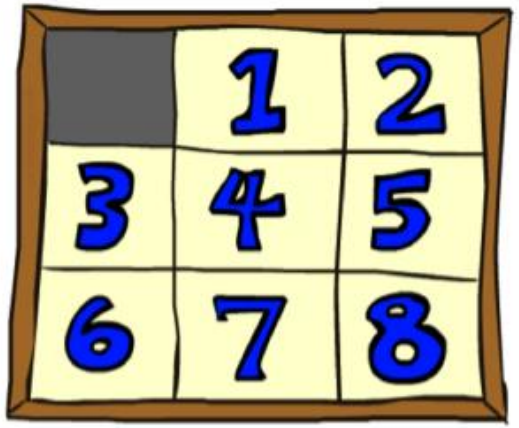
سوال: برای مسئله 8-puzzle چه توابع ابتکاری می توان استفاده کرد؟



Start State



Actions



Goal State

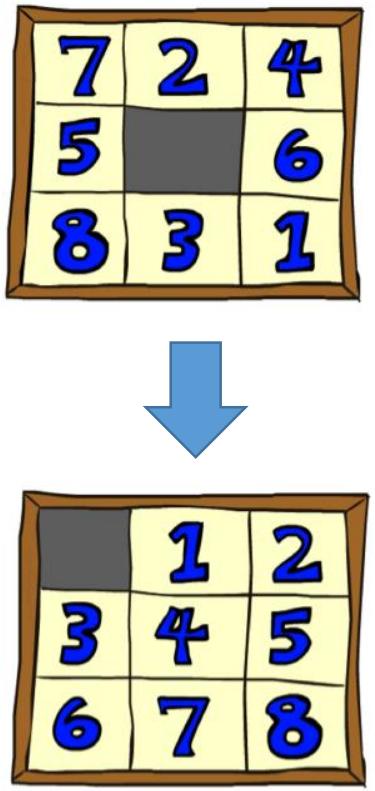
طراحی توابع ابتکاری تصدیق پذیر

طراحی توابع ابتکاری تصدیق پذیر، مهمترین بخش در استفاده از الگوریتم A* می باشد.

سوال: برای مسئله 8-puzzle چه توابع ابتکاری می توان استفاده کرد؟

h_1: تعداد خانه هایی که در جای درست خود قرار نگرفته اند.

h_2: مجموع فاصله های هر کره تا جایگاه واقعی



$$h_1 = 8$$

$$h_2 = 3 + 1 + 2 + 2 + 2 + 3 + 3 + 2 = 18$$

h_1 و h_2 هر دو تصدیق پذیر هستند.

سوال: h_1 بهتر است یا h_2؟

طراحی توابع ابتکاری تصدیق پذیر

سوال: h_1 بهتر است یا h_2؟

	Search Cost		
d	IDS	A*(h_1)	A*(h_2)
2	10	6	6
4	112	13	12
6	680	20	18
8	6384	39	25
10	47127	93	39
12	364404	227	73
14	3473941	539	113
16	–	1301	211
18	–	3056	363
20	–	7276	676
22	–	18094	1219
24	–	39135	1641

قاعده کلی: تابع تصدیق پذیری بهتر است که بیشترین مقدار را داشته باشد. زیرا به مقدار واقعی نزدیکتر است.

تمرین: برای مسئله 8-puzzle توابع تصدیق پذیر بهتری طراحی کنید.

جلسه آینده: مسائل ارضای محدودیت