Inteligenta artificiala: Tema 1

Mihail Feraru
mihail.feraru@s.unibuc.ro

May 19, 2021

Problema unui lacat

Cum contextul problemei este deja cunoscut cititorului voi sari peste o introducere a problemei si ma voi rezuma la a defini notatia (destul de informala) folosita mai jos.

Definim un lacat a fi o multime oronata $L = (x \mid x \in \mathbb{N})$, iar o cheie o multime ordonata $K = (x \mid \in \{-1, 0, +1\})$. Vom considera ca lungimea lacatului si cheilor este N. Vom folosi notatia specifica din informatica pentru a desemna a i-a pozitie a unei chei sau a unui lacat. (ie. $L[i]$ sau $K[i]$)

A descuia un lacat cu o cheie oarecare defineste urmatoarea operatie:

$$L + K = \Big(max(0, L[i] + K[i]) \;\Big|\; \forall\, 1 \leq i \leq N\Big)$$

Cu exceptia ca daca $L[i]$ este "cu truc spre j", iar $L[i] + K[i] < 0$ atunci $L[j]$ va creste cu 1.

Mai jos vom prezenta o serie de euristici.

Numarul de trucuri (inadmisibila)

Propunem urmatoarea euristica: orice nod nedescuiat va creste costul rezolvarii cu cel putin 1.
$\hat{h}_1(L) = \sum \Big\{1 \;\Big|\; L[i] > 0$ si are truc spre j$\Big\}$. Daca consideram un lacat cu un sigur truc de la i la j si o cheie cu $K[j] = -1$ stim din enuntul problemei ca nu se va aduna niciun cost pentru a anula trucul, deci $h(L) = 0$, dar $\hat{h}_1(L) = 1$, atunci $\hat{h}_1(L) > h(L)$ si euristica este inadmisibila.

0.1 Totalul de incuieri

Orice incuiere va creste costul rezolvarii cu cel putin 1. $\hat{h}_2(L) = \sum L[i]$ Admisibilitatea este evidenta din enuntul problemei, stim ca descuierea oricarei pozitii va produce un cost de 1, deci $\hat{h}_2(L) \leq h(L)$.

0.2 Totalul de incuieri si costul trucurilor

Vom combina cele doua euristici mentionate mai sus adaugand o exceptie ce va oferi admisibilitatea. Pentru fiecare truc nedescuiat (de la i la j) costul va creste cu 1, doar daca nu exista nicio cheie care are $K[j] = -1$ si $L[i] + K[i] < 0$.
$\hat{h}_3(L) = \hat{h}_2(L) + \sum \Big\{1 \;\Big|\; L[i] > 0$ si are truc spre j si $\nexists\, K[j] = -1$ si $L[i] + K[i] < 0\Big\}$. Avem de tratat doua cazuri:

1. Exista K cu $K[j] = -1$ si $L[i] + K[i] < 0$ pentru orice truc, atunci $\hat{h}_3(L) = \hat{h}_2(L)$.
2. Exista trucuri pentru care nu exista cheia dorita, deci descuierea pozitiei i va produce o incuiere pe pozitia j ce va trebui descuiata cu o alta cheie. Costul va fi minim 1.

Concluzionam ca euristica este admisibila.

0.3 Totalul de pozitii incuiate

Pentru fiecare pozitie nedescuiata costul rezolvarii este cel putin 1. $\hat{h}_4(L) = \sum \Big\{1 \;\Big|\; L[i] > 0\Big\}$.

In mod evident $\hat{h}_4(L) \leq \hat{h}_2(L) \leq h(L)$, deci este admisibila.

Comparatie performanta

Asa cum observam in tabelele de mai jos cea mai performanta euristica este euristica 1. Desi euristica 2 pare ca ar trebui sa aduca o imbunatatire, costul computational depaseste castigurile.

Algoritm	Euristica	Timp	Noduri in memorie	Noduri generate	Raport memorie/generate
A*	naiva	10.01836s	1330794	1525391	0.8724
A*	1	0.11262s	7490	8546	0.8764
A*	2	0.12605s	7490	8546	0.8764
A*	3	10.01834s	1194475	1375136	0.8686
A* open-closed	naiva	4.44912s	174871	399566	0.4377
A* open-closed	1	0.04718s	635	895	0.7095
A* open-closed	2	0.05117s	635	895	0.7095
A* open-closed	3	0.43301s	14507	22914	0.6331
IDA*	naiva	10.03571s	16	3054887	0
IDA*	1	0.08509s	9	9114	0.001
IDA*	2	0.07405s	9	9114	0.001
IDA*	3	10.01131s	14	2530360	0

Table 1: Comparatie pe exemplul de pe site

Algoritm	Euristica	Timp	Noduri in memorie	Noduri generate	Raport memorie/generate
A*	1	4.51217s	480057	580892	0.8264
A*	2	7.45s	480057	580892	0.8264
A* open-closed	1	0.21486s	1914	3744	0.5112
A* open-closed	2	0.2552s	1914	3744	0.5112
IDA*	1	3.66303s	14	837078	0
IDA*	2	4.9061s	14	837078	0

Table 2: Comparatie pe d.