Tema1IA

Reprezentarea starilor si a restrictilor

Extragerea datelor:

un dictionar in care cheia reprezinta cate un curs. Pentru fiecare curs, am creat un dictionar in care cheia reprezita cate un profesor care preda materia. Iar pentru un profesor am creat un dictionar in care retin intervalele si zilele in care poate sa predea

un dictionar in care retin,pentru fiecare profesor zilele si intervale preferate

un dictionar in care retin ,pentru fiecare zi, numarul de profesori care prefera sa predea in ziua respectiva

un dictionar in care retin numarul de sali pentru fiecare materie

STARE:

M am folosit de scheletul din laborator ul Hill Climbing

Am creat o clasa state in care am pus la constructor orarul,

numarul de conflicte soft, un dicitonar in care retin numarul de ore

tinute de fiecare profesor si un dictionar in care retin numarul de studenti ramasi de alocat pentru fiecare curs

Functia “get_next_states” primeste un curs si intoarca o lista cu starile vecine,folosindu-se de dictionarul de materii mentionat mai sus. Numarul de stari intoarse este unul eficient, deoarece ,pentru cursul primit, caut doar prin salile acceptate,profesorii care predau materia respectiva si intervalele si zilele disponibile lor

In functia “apply_move”, verific sa vad daca s au incalcat constrangeri hard.(daca da,atunci returnez vid),iar daca nu

calculez constrangerile soft,actualizez parametrii clasei state,

si intorc un tuplu ce reprezinta starea,sala,ziua si profesorul

(Am implementat si bonusul: dupa ce actualizez noul orar,

verific sa vad daca exista o constargere legata de pauza(iterez

Prin orar,pentru ziua respectiva si adaug fiecare interval intr o lista.

La final,parcurg lista si daca exista o diferenta intre 2 elemente consecutive,mai mare decat pauza permisa, atunci actualizez numarul de constrangeri soft)

Functia “room_nr_courses” o folosesc la functia de evaluare a algoritmului HC pentru a vedea numarul de materii permise pentru fiecare sala

Functia “is_final” returneza true daca nu exista constrageri soft

Functia “check_final_state_course” o folosesc pentru a vedea

daca mai am de planificat studenti pentru materia primita ca si parametru

Algoritmi si optimizari:

Pentru HC, am folosit o euristica in care acopar mai intai materia cu cele mai putine sali. In caz de egalitate,acopar mai intai materia cu cei mai multi cursanti. Am folosit o coada pentru a adauga materile in functie de euristica

Hill_Climbing:

Iterez cat timp nu am depasit numarul de iteratii si mai am cursuri in coada

Preiau lista de vecini, pentru starea in care ma aflu la momentul curent

Pentru a extrage starea cea mai potrivita(starea cu cea mai buna functie de evaluare), ma folosesc de functia “min” prin care compar starile in ordine crescatoare, dupa urmatoarea structura:

1. dupa numarul de conflicte
2. dupa capacitatea sali alese
3. dupa numarul de materii pe care le poate sustine sala
4. suma dintre : ziua in care prefera cei mai putini profesori,

   numarul de zile preferate de catre profesor

   numarul de intervale preferate de catre profesor

Am folosit functia “min” si nu o functie de evaluare dupa modelul din laborator, deoarece nu am vrut sa tin cont de ponderile pe care le aduce fiecare restrictie in parte (cele 4 enumerate mai sus)

Astfel,aleg cea mai buna stare din lista primita.

Ma opresc atunci cand am terminat de completat fiecare materie

Algoritmul MCTS:

M am folosit de scheletul din labortor, preluand functia “init_node” si “select_action”, precum si functia de mcts pe care am modificat-o. Am folosit euristica mentionata la HC pentru ordinea in care acopar materiile

Optimizari:

In functia mcts ,in loc sa aleg o stare random, am preluat functia softmax din laboratorul de HillClimbing. Aleg o stare aleatoare cu o probabilitate corespunzatoare cu calitatea starii.

Calitatea starii este data de functia de evaluare din fiecare stare.

Functia de evaluarea este una asemanatoare cu cea de la HC, in care tin cont de structura prezentata mai sus.

Astfel, converg mai repede catre o solutie finala, fara restrictii hard si cu cat mai putine constrangeri soft. In functie de bugetul pe care il aloc, acesta poate sa obtina o solutie cu mai putine incalcari soft.

Functia de mcts respecta desfasurarea algoritmului din laborator:

1. Dacă algoritmul pornește cu un arbore gol (fără memorie), atunci se construiește un nod nou. Altfel se alege subarborele corespunzător ultimei acțiuni a adversarului.

2. Până când se atinge limita bugetului de calcul:

3. pornind din rădăcină, se alege succesiv un nod următor până când se atinge o stare finală (check_final_state_course) sau un nod din care nu s-au explorat toate acțiunile posibile

4. pentru un nod care nu este final și din care nu s-au explorat toate acțiunile, se construiește un nod-copil pentru una dintre acțiunile neexplorate

5. se simulează un orar pornind din nodul nou până într-o stare finală

6. se evaluează starea finală și se calculează o recompensă

recompensa este calculata in felul urmator:

Daca ajung intr o stare finala primesc:

1- numarul de constrangeri soft/20

Am folosit o fractie pentru a avea o recompensa intre 0 si 1

Daca starea nu este finala,primesc 0

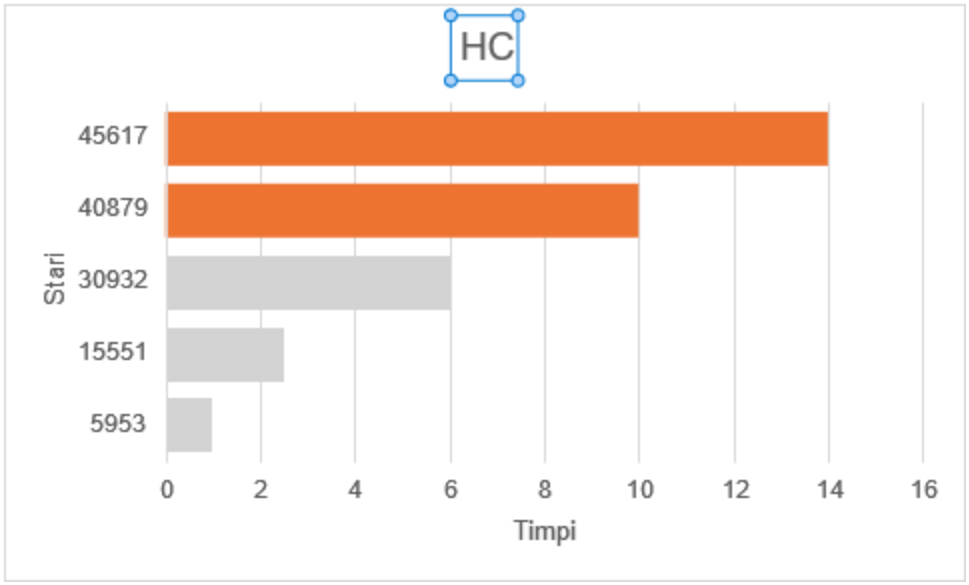
7. se propagă înapoi acea recompensă, actualizându-se și statisticile (numărul de vizite) pentru fiecare nod până la rădăcină

Pentru a verifica cat mai rapid, daca s au explorat toate actiunile posibile unei stari, retin in dicitonarul de acitons indexul unei stari

Exemplu: Eu stiu ca, pentru acceasi stare, lista de vecini pe care o sa o generez este mereu acceasi. Atunci cand se alege o stare noua , eu adaug in lista de actions a parintelui indexul la care se afla starea aleasa, pe post de cheie. Astfel, cand vreau sa verific daca am o anumita stare in lista de actiuni, o sa ii iau indexul din lista si o sa l compar prin cheile dictionarului meu.

Comparatie intre HillClimbing si MCTS:

HC	MIC	MEDIU	MARE	CONSTRANS	BONUS
Timpi	1s	6s	10s	2.5s	14 sec
Stari	5953	30932	40879	15551	45617
Conflicte	0	0	0	0	4 soft,0 pauze incalcate
Iteratii	33	49	51	56	104



Constrangeri soft la bonus:

HC creat alege cea mai buna stare locala la momentul respectiv de timp, folosind functia de evaluare pe care o dau. Astfel,desi ,la momentul respectiv de timp, ea pare cea mai ok, nu o sa convearge catre solutia minima globala. Constrangerile soft primite sunt de la ultima materie acoperita,deoarece nu mai are sloturi disponibile din care poate sa aleaga cu cost 0.

Pentru datele din tabel,s a folosit un buget de 5

MCTS	MIC	MEDIU	MARE	CONSTRANS	BONUS
Timpi	36s	3-4min	3-4min	2 min	14 min
Stari	33455	206612	152165	108960	510334

Conflicte	0	0	0	0	7-9soft,0/1 pauze incalcate
Iteratii	165	245	255	280	520

Constrangerile soft o sa fluctueze,deoarece se alege random o stare,folosind softmax, din lista de stari vecine . Functia de evaluare folosita o sa convearga catre un minim local,nu catre unul global.

De asemenea, constrangerile soft primite sunt de la ultima materie acoperita,deoarece nu mai are sloturi disponibile din care poate sa aleaga cu cost 0.