Mesterséges Intelligenciák féléves feladat

JMDRGG - Könnyű

2022/23 I. félév

A feladat implementációját – azaz a forráskódot, és a lezáró dokumentumot az `aitflew@uni-miskolc.hu` e-mail címre kell elküldeni, majd azt a 13. és 14. héten a gyakorlatokon kell megvédeni. A kész feladat leadásához a GitHub javasolt, de nem kötelező. Az e-mail tárgya és a küldő jól beazonosítható legyen, ez vonatkozik a GitHub felhasználóra is.

Tetszőleges nyelv, keretrendszer, technológia választható. Az egyetlen megkötés, hogy nem lehet olyan könyvtárat használni, amely tartalmazza a feladat modelljét és/vagy a megoldó algoritmusokat.

A plusz feladatok elvégzésével magasabb érdemjegyet lehet elérni.

Probléma: Flow-shop ütemezés

A flow-shop feladat egy ún. egyutas, többoperációs gyártásütemezési feladat. A gyártásütemezési feladatok formális leírására az alfa, béta, gamma jelölést alkalmazzuk ($\alpha|\beta|\gamma$). E szerint a rendszer szerint az egyszerű flow-shop feladat leírása:

$$F||C_{max}$$

Jelentése:

Az erőforrások az ütemezési időszakban folyamatosan rendelkezésre állnak. Az erőforrások egyszerre csak egy munkán dolgoznak. A munkák legkorábbi indítási időpontja nulla (bármikor indíthatóak). Minden egyes munkához adott m számú operáció tartozik, melyeknek pontosan ismert a végrehajtási ideje: $p_{i,j}, i \in 1, 2, \ldots, n, j \in 1, 2, \ldots, m$. Az operációk végrehajtási sorrendje kötött és minden munka esetében azonos. Az operációk végrehajtása nem szakítható meg. A gépek között a munkák várakozhatnak, a műveletkőzi tárolók mérete nem korlátos. Az ütemezés célja az utolsóként elkészülő munka befejezési időpontjának minimalizálása.

Példa

$F|perm|C_{max}$ (nincs előzés)

Munkák száma: $n = 5$

Munkák halmaza: $J = J_1, J_2, J_3, J_4, J_5$

Gépek száma: $m = 3$

Gépek halmaza: $M = M_1, M_2, M_3$

A lehetséges megoldások száma: $5! = 120$

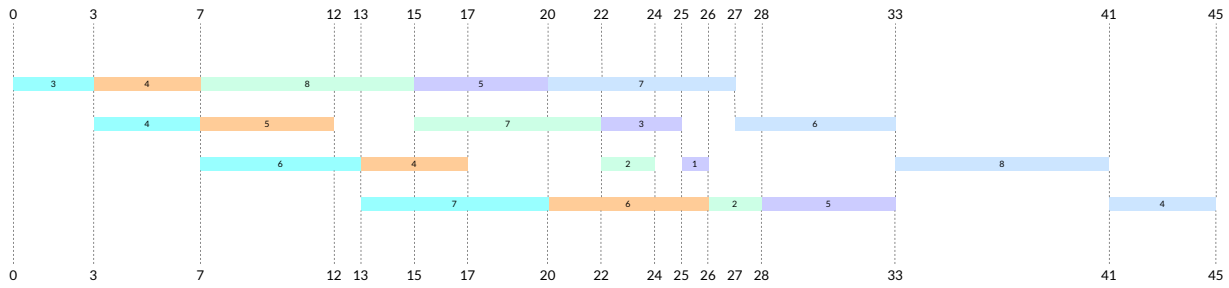
Egy lehetséges ütemterv: $[J_5, J_3, J_1, J_2, J_4]$, ekkor a végrehajtási sorrend minden gépen: $J_5 \rightarrow J_3 \rightarrow J_1 \rightarrow J_2 \rightarrow J_4$.

Az adott ütemterv szimulációja:
Adott egy feladat az alábbi időadatokkal:

$p_{i,j}$	M_1	M_2	M_3	M_4
J_1	3	4	6	7
J_2	4	5	4	6
J_3	8	7	2	2
J_4	5	3	1	5
J_5	7	6	8	4

Adott a következő ütemterv: $S = [J_5, J_3, J_1, J_2, J_4]$.

Induljunk az első gép első munkájától. Ábrázoljuk a munka végrehajtását időarányos szimbólummal (vonal vagy téglalap). Amikor az első munkát befejezte az első gép azonnal indul a második gép az első munka valamint az első gépen a második munka. Az első gép folyamatosan dolgozik, nem várakozik. Az első munka sem várakozik a gépek előtt. Egy közbenső gépen egy közbenső munka akkor indítható, ha a munka végrehajtása befejeződött az előző gépen és az aktuális gép befejezte az előző munkát. Ha a munka a korábbi gépen korábbban készült el mint ahogy az ektuális gép felszabadul, akkor a munka várakozik. Ha az aktuális gép szabadul fel korábbban, mint ahogy a munka megérkezik, akkor a gép várakozik.



1. ábra: A megoldás Gantt diagrammja

A befejezési időpontot az utolsó gép utolsó munkájának időpontja adja meg: $C_{max} = C_5 = 45$.

Ezeket a feladatokat inicializálja valamilyen véletlen szám generátorral! Legyen különböző méretűek:

- a munkák száma legyen 10, 20, 50, 100, 200, 500,
- a gépek száma legyen 5, 10, 20.

A legenerált feladatok legyenek perzisztensen tárolva (vagy seedhez kötéssel procedurálisan generálva).

Algoritmus: Tabu keresés

A feladatot a Tabu kereséssel kell megoldani. Ez az algoritmus az egyszerű szomszédsági keresésre épül, azzal a különbséggel,

hogy a keresés során bevezetünk egy tabu listát, melyben tároljuk, hogy mik azok a bázisok, ahol már jártunk. Ezt determinisztikus elfogadási kritériumnak nevezzük. Ez a lista változásokat, vagy eredményeket is tarthat nyilván.

Ehhez a tabulistához általában rendelűnk egy maximális elemszámot. Ha ez az elemszám nagyon nagy, akkor a keresés túlságosan korlátozottá válik, ha túl kicsi, akkor ismétlődő keresési útvonal alakulhat ki.

Persze ezt nem csak mérettel lehet korlátozni. Lehetséges, hogy az egyes tiltott megoldásokhoz egy látogatási számot rendelünk. Ha a látogatások száma elér egy korlátot, akkor kivesszük a tabu listából.

Az elemek reprezentációja a listában lehet sokféle, mely a feladattól függ. Lehet egy kiindulási pont és a hozzá tartozó tiltott átalakítás, vagy átalakítások. Lehetnek a tiltott eredmények, vagy a tiltott eredmények valamilyen kódolt formája (akár hash is).

Plusz feladat

Vizsgálja meg, hogy tabu lista méretének változtatásával hogyan változik az eredmény, a futási idő, és a memóriaigény!

Vizsgálja meg, hogy mi történik az egyes kikerülési kritériumoknál! Ha fix a tabulista, ha csak azok vannak a tabulistában, amikre már sokszor sor került, ha az egyes elemek csak t ideig vannak a listában, ha n látogatás után kikerülnek az elemek.

A tabu listából a kikerülés kritériumainak paraméterei változhatnak a keresés során. Próbáljon ki különböző stratégiákat arra, hogy ezek hogyan változzanak a keresés során. Ehhez érdemes tanulmányozni a szimulált hűtés hűtési stratégiáit.

Az algoritmus végét egy ún. leállási feltétel vizsgálatával érjük el. Ez lehet egy adott iterációszám elérése, egy előre meghatározott hőmérséklet elérése, valamennyi iteráci eltelte úgy, hogy nem javult az eredmény, vagy ezeknek valamilyen kombinációja. Vessen össze ezekből legalább kettőt, vagy valamilyen kombinációikat. Hogyan hat ez az eredményre és a futási időre?

Készítsen egy alkalmazást, ahol a feladat megadásával elkészíti autómatikusan a megoldást és ezt valahogy megjeleníti (térkép, Gantt diagram).

A feladat lezárása

A féléves feladat lezárásaként egy dokumentumot kell elkészíteni, melyben szerepelnek a megoldás lépései, a választott nyelv, technológia, könyvtárak, keretrendszerek. Esetlegesen, a külön irodalomkutatás eredményeit is tartalmazza.

Minden plusz munka javítja a kapható érdemjegyet. A könnyű feladat csak kivételes esetben érhet el hármasnál jobb jegyet.