Dinamikus jármű útvonaltervezés

Önálló projekt, szakmai gyakorlat III.

Hatala Imre

Témavezető: Horváth Markó (SZTAKI)

Áttekintés

A projekt témája

- Dinamikus felvétel-lerakási probléma
  (Dynamic Pickup and Delivery Problem, DPDP)
  - Felvétel-lerakási probléma
    (Pickup and Delivery Problem, PDP)
  - Sztochasztikus-dinamikus jármű útvonaltervezési probléma
    (Stochastic Dynamic Vehicle Routing Problem, SDVRP)
    - Bizonytalanság (igények, logisztikai környezet, erőforrások)
    - Statikus tervezés helyett dinamikus tervezést igényel
- Konkrét feladat: ICAPS 2021 – Huawei

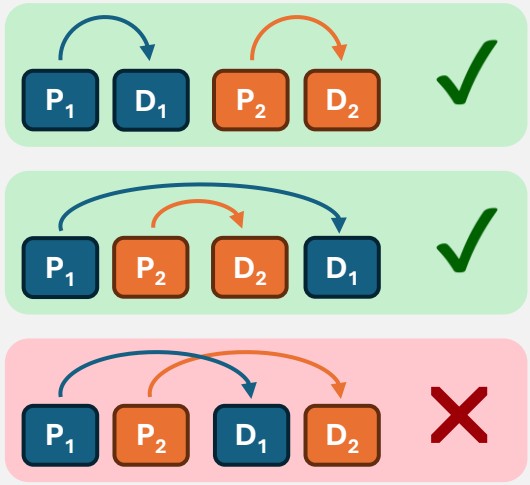
Az előadás menete

- A feladat bemutatása
- Algoritmusok
- Eredmények, konklúzió

A feladat – röviden

- A Huawei valós problémája: termékek és alkatrészek különböző gyárak közötti szállítása
- Dinamikus igények: megrendelések 10 percenként
- A statikus feladat (egyszerűsítve)
  - Input:
    - Gyárak és úthálózat (teljes irányított gráf, távolság, utazási idő)
    - Rendelések (felvételi és lerakási pont, méret, kézbesítési határidő)
    - Járművek (kapacitás)
  - Korlátok:
    - Szokásos korlátok (rendelések teljesítése, kapacitás)
    - Várakozás
    - LIFO (Last In, First Out) rakodási szabály
  - Output: Szállítási terv
- A dinamikus feladat (10 perces iterációk)
  - További input és korlátok:
    - Új igények iterációnként
    - Jármű információk (pozíció, útvonalterv, feladatlista)
    - Úton lévő jármű célállomását nem lehet megváltoztatni
  - Output: Szállítási terv iterációnként

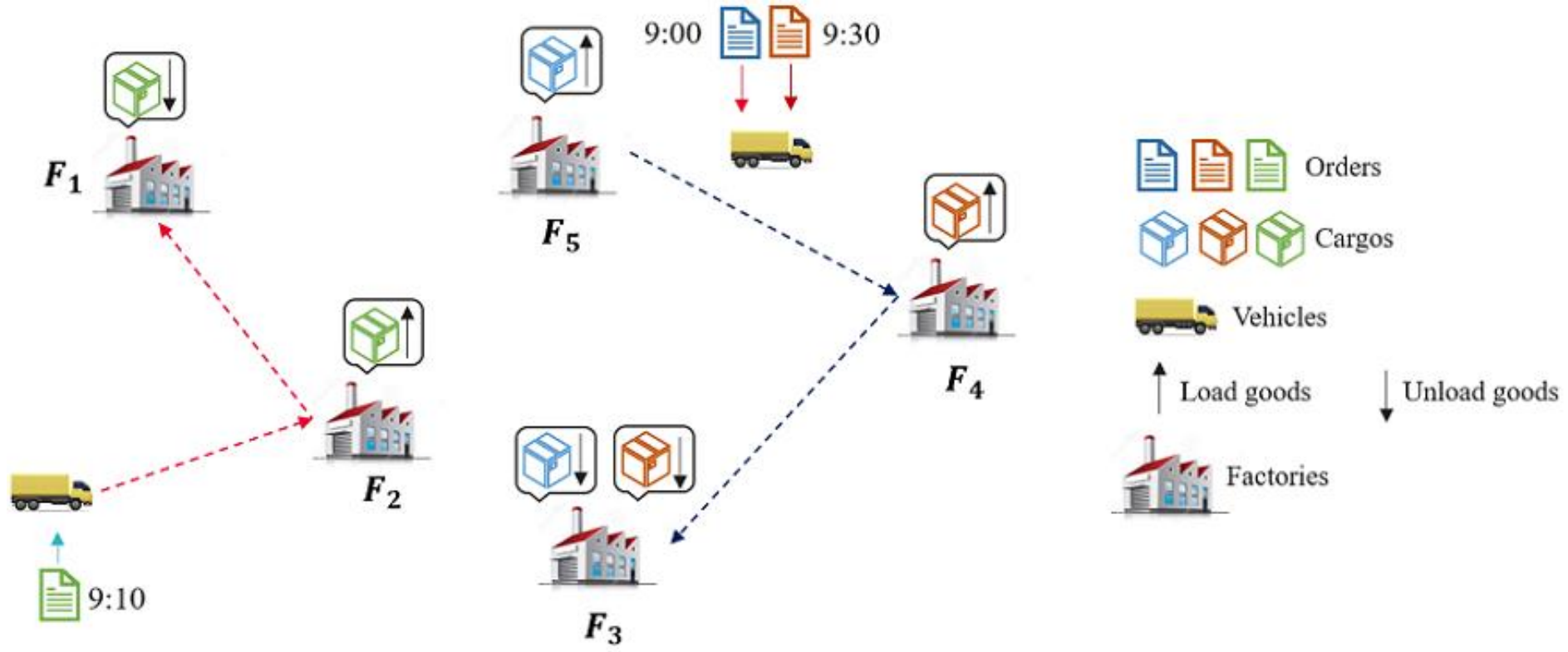
LIFO szabály



Célfüggvény

$$min\ f = \lambda \times f_1 + f_2$$

- f_1 : összesített késés
- f_2 : a járművek átlagosan megtett távolsága
- λ : ≥ 1 konstans ($\lambda = \frac{10000}{3600} \approx 2{,}78$)



A kiszállítás menete

Forrás: https://competition.huaweicloud.com/information/1000041411/circumstance

Algoritmusok

Egyszerűsített feladat

- Első két tesztcsoport példányai (2x8 instancia)
- Nincs várakozás

Demo algoritmus

- Felvétel-lerakási feladatok egyesével (egyszerre csak egy csomag)
- Új feladatok kiosztása: az i. feladat az $i \ mod \ K$ sorszámú járműhöz kerül (ahol K a járművek száma)

Naiv algoritmus

- Felvétel-lerakási feladatok egyesével (egyszerre csak egy csomag)
- Új feladatok kiosztása: az i. feladat ahhoz a járműhöz kerül, amelyik (a meglévő feladatai végeztével) a leghamarabb tud odaérni a felvételi pontra

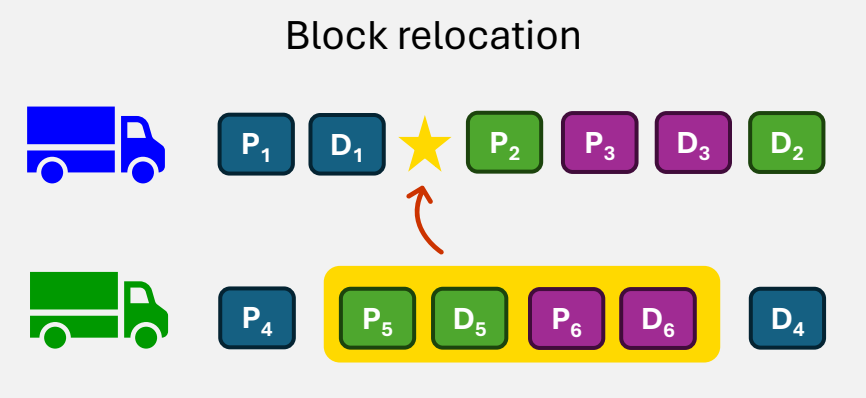
Best insert

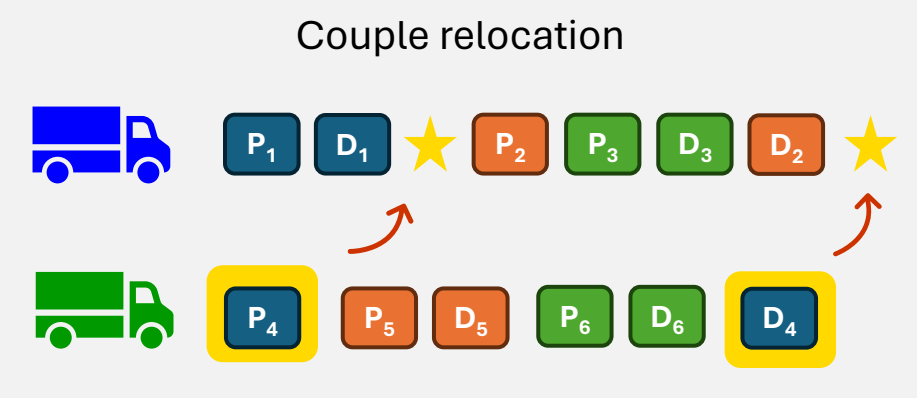
- Új feladatok kiosztása: az i. feladatot úgy illeszti be valamely jármű útvonaltervébe, hogy azzal minimalizálja a keletkező terv célfüggvényértékét
- Itt már több csomag is lehet egy járművön, ezért ellenőrizni kell a tervezési korlátokat (úton lévő jármű célállomása nem változhat, kapacitáskorlát, LIFO szabály)

#	Demo	Naiv	Best insert
1	157 938.2	198.1	142.5
2	89 812.9	9 206.1	103.8
3	33 834.0	157.1	102.6
4	41 754.2	158.3	102.9
5	147 364.0	4 403.6	5 448.5
6	52 385.6	190.2	120.6
7	95 747.5	6 451.3	3 988.1
8	38 768.0	124.3	63.1
9	2 121 982.7	1 823 806.3	227.3
10	5 415 769.6	5 234 380.2	225 918.2
11	1 919 632.2	1 168 255.7	9 094.5
12	3 249 158.9	2 556 355.2	27 872.0
13	2 495 984.3	1 896 729.5	24 634.9
14	2 614 084.7	2 257 233.7	6 841.1
15	3 362 476.0	2 771 743.4	48 309.0
16	3 239 131.3	1 825 784.3	42 837.9

Lokális keresés

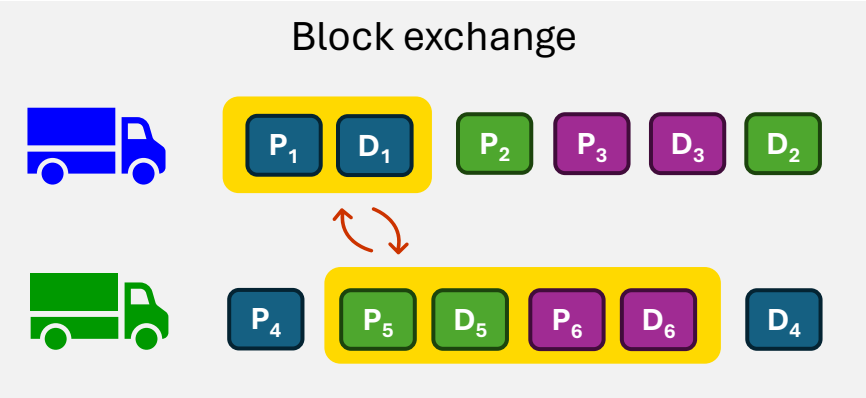
- Általános optimalizálási módszer (metaheurisztika)
- Operátorok → keresési tér
- Felhasznált operátorok:
    1. **Block relocation**: egy jármű útvonaltervének egy szakaszát (egy ún. blokkot) áthelyezzük egy másik helyre (ami lehet egy másik jármű útvonalterve is).
    2. **Couple relocation**: egy jármű útvonaltervéből egy felvétel-lerakási feladatpárt kiemelünk és egyenként beillesztjük őket egy-egy másik helyre (amelyek lehetnek egy másik jármű útvonaltervében is).

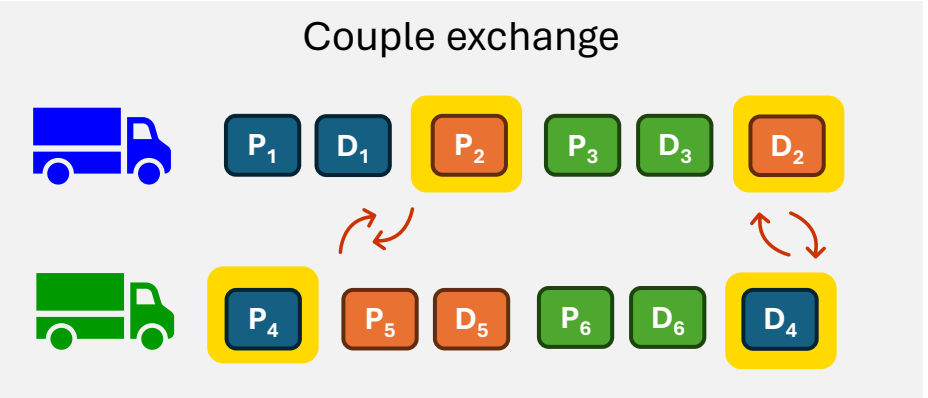




Lokális keresés

- Felhasznált operátorok:
  3. **Block exchange**: két diszjunkt blokkot (amelyek lehetnek azonos vagy különböző útvonalterv részei) megcserélünk.
  4. **Couple exchange**: két felvétel-lerakási feladatpárt (amelyek lehetnek azonos vagy különböző útvonalterv részei) megcserélünk, azaz megcseréljük a két felvételi feladatot és megcseréljük a két lerakási feladatot is.





- Az algoritmus
  1. Meghatározzuk a felhasználható operátorok halmazát
  2. Kezdeti megoldás: *best insert* algoritmus
  3. Javítás, amíg tudunk érdemben (amíg a javítás mértéke $> \varepsilon$)

#	Demo	Naiv	Best insert	LS – block relocation	LS – couple relocation	LS – block exchange	LS – couple exchange	LS – all operators	Gold	Silver	Bronze
1	157 938.2	198.1	142.5	141.8	138.3	142.4	141.6	132.6	134.0	2 300.0	130.0
2	89 812.9	9 206.1	103.8	89.7	92.1	97.6	96.8	92.1	95.6	30 500.0	91.4
3	33 834.0	157.1	102.6	98.7	104.6	101.0	101.3	102.5	96.8	35 800.0	96.5
4	41 754.2	158.3	102.9	108.1	104.7	101.3	102.7	100.8	94.6	5 490.0	104.0
5	147 364.0	4 403.6	5 448.5	5 451.7	5 451.6	5 448.5	5 448.7	5 451.9	3 310.0	16 900.0	5 450.0
6	52 385.6	190.2	120.6	128.5	105.6	120.8	120.8	120.6	105.0	4 760.0	118.0
7	95 747.5	6 451.3	3 988.1	3 752.6	3 752.6	3 987.7	3 988.1	3 788.6	4 390.0	12 800.0	7 360.0
8	38 768.0	124.3	63.1	63.9	66.7	61.6	61.6	64.3	68.8	797.0	769.0
9	2 121 982.7	1 823 806.3	227.3	170.7	172.2	166.2	3 742.8	6 553.1	152.0	181 000.0	8 480.0
10	5 415 769.6	5 234 380.2	225 918.2	208 801.2	201 300.0	344 534.1	236 255.2	250 489.7	194 000.0	1 770 000.0	187 000.0
11	1 919 632.2	1 168 255.7	9 094.5	2 305.9	2696.9	4 036.4	173.2	5 750.4	198.0	181 000.0	3 040.0
12	3 249 158.9	2 556 355.2	27 872.0	83 011.8	78 175.3	39 146.9	30 224.2	38 807.4	52 900.0	567 000.0	84 200.0
13	2 495 984.3	1 896 729.5	24 634.9	3 817.0	185.0	7 440.5	6 450.3	4 572.0	7 180.0	220 000.0	277.0
14	2 614 084.7	2 257 233.7	6 841.1	145.5	13 570.9	4 546.1	177.9	2 101.5	9 390.0	237 000.0	7 820.0
15	3 362 476.0	2 771 743.4	48 309.0	25 320.9	13 490.5	40 674.2	21 979.3	22 782.8	13 500.0	793 000.0	149 000.0
16	3 239 131.3	1 825 784.3	42 837.9	32 369.4	22 064.2	49 447.1	49 452.8	20 994.7	16 800.0	854 000.0	57 800.0

Eredmények első 16 teszt instancián

Zöld kiemelés: a bemutatott algoritmusok közül a legjobb

Sárga kiemelés: a versenyen dobogós megoldások közül a legjobb

Konklúzió

- Az első két tesztcsoport példányain jól teljesítenek az algoritmusok (vö. dobogós helyezések megoldásai), azonban itt nem volt várakozás
- Az algoritmusok „mohó” elven működnek, ez hátulütőkkel jár
  - Az erőforrásokat potenciálisan pazarló módon használja fel
  - Nem elég flexibilis, a dinamikus tervezés ennél több rugalmasságot igényel
  - Például: sporadikusan előforduló, a többi megoldáshoz képest szignifikánsan jobb eredmények
  - Például: a lokális keresést használó algoritmusok sokszor rosszabb eredményt adnak, mint a *best insert*
- Továbbfejlesztési lehetőségek
  - Rugalmasság – erőforrás-tartalék képzése
    - A távoli határidővel rendelkező megrendelések teljesítésének késleltetése
    - Az alacsony töltöttségű járművek várakoztatása, illetve kerülőútra küldése
  - Az instanciák karakterisztikájának vizsgálata
    - Vannak-e sűrűn látogatott lokációk, amelyek közelében megéri szabad kapacitást állomásoztatni
  - A lokális keresés továbbfejlesztés
  - Szakdolgozat
    - Továbbfejlesztési lehetőségek
    - Várakozás
    - Teszt és összehasonlítás az összes teszt példányon

Köszönöm a figyelmet!