# Dokumentacija: Parkiranje vozila u spremištu

## Opis problema

U ovom projektu pokušali smo riješiti problem parkiranja vozila u spremištu. Problem bi se mogao jednostavno opisati na sljedeći način: "Cilj je smjestiti vozila u parkirne trake imajući na umu vrijeme polaska vozila iz spremišta, kako bi svako vozilo ujutro moglo što lakše izaći iz spremišta." Na prvu se problem ne čini veoma kompliciran, međutim uz brojna dodatna ograničenja problem brzo postaje složen. Ograničenja su sljedeća:

* Određena vozila moraju se smjestiti na trake sa specifičnom opremom: tramvaj se može parkirati isključivo na traku s tračnicama, trolejbus isključivo na traku s električnom kontaktnom mrežom, itd.
* Vozilo ne može napustiti traku ukoliko je neko vozilo ispred, odnosno vrijeme izlaska prvog vozila u traci iz spremišta mora prethoditi vremenu izlaska drugog vozila. Također, u određenim slučajevima vozila u jednoj traci mogu blokirati vozila u drugoj traci, tj. postoje slučajevi u kojima sva vozila jedne trake moraju napustiti spremište prije ijednog vozila druge trake.
* Vrijeme napuštanja spremišta mora se strogo poštivati za sva vozila.
* Određena vozila mogu imati specifičan raspored. Za neku grupu vozila može postojati pojačan raspored u vrijeme gužvi ili pak poseban raspored održavanja.
* itd.

## Opis algoritma

Algoritam koji smo koristili za rješavanje ovog problema je sljedeći: sortiramo vozila po vremenu polaska, zatim tražimo sve moguće kombinacije vozilo-traka gdje je vozilo moguće dodati u traku, i pokušamo izvesti svaki od tih poteza te računamo omjer fja g2/g1 za svaki potez pritom pamteći najveći omjer g2/g1 i najbolji potez. Nakon što smo prošli kroz sve kombinacije imat ćemo najbolji potez i njega izvršimo. Algoritam nakon toga nastavlja postupak, ali sada s jednim vozilom manje i tako radi sve dok ima neraspoređenih vozila. Algoritam se pokreće onoliko puta koliko je to moguće unutar zadanog vremena.

## Pseudokod algoritma

sortiraj vozila po vremenu polaska

dok ima neraspoređenih vozila radi

inicijaliziraj best\_ratio, best\_track i best\_vehicle

promiješaj trake nasumično

za svako vozilo iz liste sortiranih vozila radi

za svaku traku iz promijesanih traka radi

pokusaj dodati vozilo u traku

ako uspjesno dodavanje

izracunaj omjer g2/g1

ako omjer veci od best\_ratio

azuriraj best\_ratio, best\_track i best\_vehicle

makni vozilo iz trake

ako postoji best\_track i best\_vehicle

dodaj vozilo best\_vehicle u traku best\_track

makni vozilo iz liste sortiranih vozila

else

vrati False

## Opis dobivenih rezultata i diskusija

Implementirani algoritam nam daje dobar rezultat za 1.instancu, međutim za 2. i 3. instancu algoritam ne daje valjano rješenje unutar razumnog vremena (do 20ak min). Iz ovog bi se dalo zaključiti da je algoritam prespor (zbog provjere svih kombinacija) i zato na kompliciranijim instancama (2. i 3.) ne uspije dati rješenje u razumnom vremenu.

## Kratki zaključak

Kao što je već spomenuto, čini se da je naš algoritam prespor i da zato radi samo za 1.instancu. Razlog tomu je (vrlo vjerojatno) veliki broj kombinacija koje se provjeravaju i pritom se svaki put racunaju fje cilja. Poboljšanje koje bi se dalo napraviti je smisliti neki malo jednostavniji algoritam koji ne provjerava sve moguće kombinacije i koji ne zove fje cilja pri svakoj mogućoj kombinaciji.

## Studenti koji su radili na projektu:

* Davor Češljaš 0036486648
* Krešimir Dodig 0036483971