## 3. Gyakorlat

**F1.** Egy hosszú utazást tervezünk autóval. Az autó tankjában $n$ liter benzin fér el. Van egy térképünk, amely ábrázolja az úton levő benzinkutakat. Tervezzünk hatékony algoritmust, amely megadja azon benzinkutakat, amelyeknél megállva, minimális számú megállással megoldható az utazás!

*Megold:* Legyenek a $K_{n-1}, \ldots, K_0$ a benzinkutak az út során elérhető sorrendben, és $K_n$ a kiindulási pontunk. A $P_i$ részprobléma az a feladat, hogy az $K_i$ pontból, ahol tele a tank jussunk el a célba minimális számú megállással. A $P_i$ részprobléma esetén a mohó választás az, hogy a lehető legutolsó benzinkútnál állunk meg, azaz az utolsó olyan kútnál amely nincs messzebb, mint $n$. Tegyük fel, hogy ez az út a $K_i, \ldots, K_j$ pontokat fedi le, és a maradék útra az optimális megoldást vesszük. Ekkor a $P_i$ probléma így kapott megoldásának a költsége $1 + OPT(P_j)$. Másrészt ez valóban $OPT(P_i)$ hiszen $P_i$ minden megoldásában tankolni kell a $K_{i+1}, \ldots, K_j$ kutak valamelyikénél, és bárhogy választunk kutat legalább a $K_j$-től kezdődő utat meg kell még tennünk. Tehát az algoritmus mindig az utolsó kútnál áll meg, ahol még nem fogy el a benzin.

**F2.** Vegyük azt az ütemezési problémát, ahol egy gép van, minden munkának van egy végrehajtási ideje és egy határideje. A cél a maximális késedelem (befejezési időből kivont határidő) minimalizálása. Igazoljuk, hogy a mohó algoritmus, amely mindig a legkisebb határidővel rendelkező még nem végrehajtott munkát hajtja végre optimális.

*Megold:* Vegyünk egy részproblémát, ami ez esetben a hátralevő munkák ütemezése. Tekintsünk egy olyan ütemezést legyen $S$, amely nem egy legkisebb határidővel rendelkező munkát ütemez elsőnek. Legyen a legkisebb határidővel rendelkező munka $j$. Tegyük fel, hogy $S$ valamilyen $i_1, i_2, \ldots, i_k, j$ munkasorrenddel kezdődik. Vegyük $S'$-t, amelyet úgy kapunk $S$-ből, hogy a munkák ezen kezdeti sorrendjét kicseréljük a $j, i_1, i_2, \ldots, i_k$ sorrendre. Az új ütemezésre az $i_1, \ldots, i_k$ munkák később fejeződnek be, viszont a befejezési idejük nem lesz nagyobb, mint $j$-nek az $S$ ütemezésben, ezért a késedelmük sem, hiszen $j$ határideje minimális volt. Következésképpen az így kapott megoldás is optimális lesz, amivel igazoltuk, hogy a mohó algoritmus optimális.

**F3.** Vegyük a pénzváltási probléma azon változatát, ahol a lehető legkevesebb érmére akarunk felváltani $n$ forintot.

a) Tervezzünk mohó algoritmust, ha a felhasználható érmék $c^0, c^1, \ldots c^k$ forintok, valamely $c > 1, k > 1$ egészekre. Igazoljuk, hogy az algoritmus optimális!

b) Adjunk meg olyan érmesorozatot, amelyre a mohó algoritmus nem optimális!

c) Határozzuk meg milyen $a < b$ számokra optimális az $(1, a, b)$ pénzérmékre a pénzváltási probléma?

*Megold* a) Azt kell igazolnunk, hogy minden felváltandó összegre van olyan optimális felváltás, amely használja a legnagyobb a felváltandónál nem nagyobb váltópénzt. Az állításnál többet igazolhatunk. Az optimális felváltás, a szám $c$ alapú számrendszerbe felírt formája, ahol a $c^k$-nál nagyobb helyiértéken levő

számokat $c^k$-s értékekkel fejezzük ki. Ez az állítás úgy igazolható, hogy tetszőleges felbontásból kiindulunk, és az egyesektől kezdve minden $c^k$-nál kisebb érmére, amennyiben $c$-nél több van helyettesítünk $c$ darabot az eggyel értékesebb érmével, ezzel rendre jobb megoldáshoz jutva.

b) Legyenek az érmék $1, 4, 5$. Ekkor $n = 8$-ra a mohó eredmény $8 = 5 + 1 + 1 + 1$, az optimális $8 = 4 + 4$.

c) Ha a $a$ osztója $b$-nek egyből adódik, hogy a mohó algoritmus optimális, tetszőleges $b$-nél nem kisebb szám felváltása esetén, lesz az érméknek olyan halmaza, amelyben az összeg $b$, és ezt kicserélve $b$-vel egy nem rosszabb megoldást kapunk. Ha $a$ nem osztója $b$-nek, akkor írjuk $b$-t fel $k \cdot a + m$ alakban, ahol $0 < m < a$. Ekkor amennyiben a $(k+1)a$ szám mohó felírása ($b$ és $a - m$ darab 1) több érmét tartalmaz, mint $k + 1$, akkor a mohó nem optimális. Ellenkező esetben a mohó algoritmus optimális, mert a $k + 1$ darab $a$-t ki tudjuk cserélni egy $b$-re és $a - m$ darab 1-re.