Elixir NHF1 dokumentáció

Reiter Márton, VVUED2
marci.reiter501@gmail.com

Követelmények elemzése:

A programmal szemben támasztott követelmények megfelelnek a házi feladat követelményeinek. Egy megkapott kempinget leíró hármas alapján kell előállítania minden lehetséges megoldást a sátrak elhelyezésére, belátható időn belül. Fontos tehát, hogy gyorsnak kell lennie, és minden lehetséges megoldást meg kell találnia.

Tervezés és eljárások:

Két fő eljárása van a programnak. Először előállítja az összes olyan lehetséges elrendezést, amiben a sátrak pozíciója, nem ütközik másik sátorral, vagy fával, és egy sorban/oszlopban megfelelő mennyiségű fa van. Majd ezután, az így kapott lehetséges megoldásokat ellenőrzi, hogy azok valóban megfelelnek-e a követelményeknek a khf3 némileg módosított ellenőrzője szerint.

A program jó megoldást nem zár ki, ellenben rossz megoldást az első kiderülésnél elvet. Így nem vizsgálunk feleslegesen elhelyezéseket, ha tudjuk, hogy egy új sátor ezt elrontja.

Kódolás, algoritmusok:

A megoldásokat előállító algoritmus a következő:

- Vesszük a kapott fák közül az elsőt, és előállítjuk rá a lehetséges irányokat
    - A lehetséges irányoknál figyelembe vesszük, hogy az iránynak megfelelően elhelyezett sátor ne essen kívül a kemping mezőn, ne ütközzön másik fával, és az eddig elhelyezett sátrakkal sem.
- Minden így kapott lehetséges irányra rekurzívan ismételjük ezeket a lépéseket, viszont az eddig kiválasztott irányokhoz hozzáfűzzük az aktuális irányt, és csak azokat a fákat adjuk tovább, amiket még nem vizsgáltunk. Továbbá frissítjük az adott sorra/oszlopra vonatkozó korlátot, így sokkal rövidebb idő is elég ellenőrizni a mennyiséget adott sorban/oszlopban.

- Amennyiben már nem maradt fa, amit vizsgáljunk egy adott ágon, tehát rendelkezünk egy minden fához irányt tartalmazó listával akkor találtunk egy megoldást, és azt visszaadjuk.

A megoldások ellenőrzésének algoritmus a következő:

- Kiszámolom hány sátor van egy-egy sorban/oszlopban
- Ha ez a szám nem negatív korlát esetén több, mint a megadott korlát, akkor hibás a megoldás, egyébként pedig jó

Tesztelési megfontolások:

A programot időméréssel teszteltem nagy inputra. A cél az volt, hogy gyors legyen, így többször, több helyen is mértem az időt. Saját gépen egy több tízes nagyságrendben fákat tartalmazó bemenetre 500 milliszekundom alatt futott, ezzel már megelégedtem.

Ezen kívül további teszteket nem futtattam, az ETS-ben található tesztek kimenetére hagyatkoztam.

Az algoritmus magában biztosítja, hogy csak a jó megoldások álljanak elő.

Kipróbálási tapasztalatok:

Viszonylag effektíven fut a program és az algoritmus. Nagy inputra is gyorsan előállítja a megoldásokat. Tudom, hogy létezik gyorsabb, de megelégedtem ezzel.

Kezdetben csak 8 tesztre futott le időben a program az ETS-en, így érdekes volt megtapasztalni, hogy a jónak gondolt algoritmus még mennyi helyen lehet optimalizálatlan. Némi átgondolás után, még sok részén tudtam javítani, és némi plusz logikával durván csökkenteni a lehetséges ágak számát.