A 15-ös játék

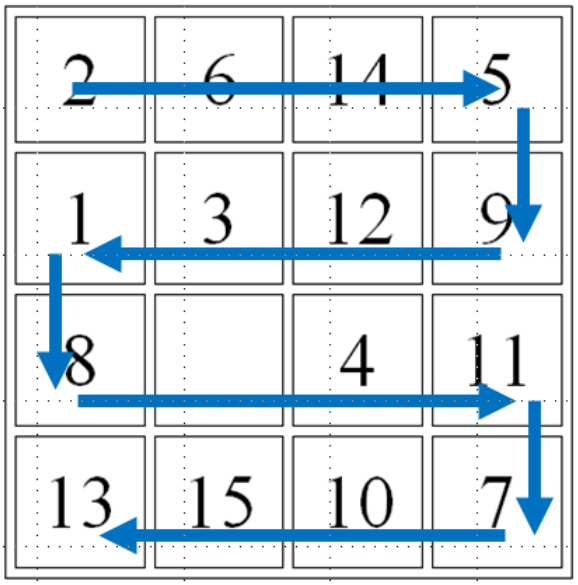
Szécsi Péter György (K7I0ET)

A probléma:

Egy 4*4-es táblán található 15 db számozott négyzet alapú lap. Ezek csúsztatásával érjük el, hogy a végén a számok sorfolytonosan növekvő sorban szerepeljenek a táblán (a bal alsó sarkot üresen hagyva).

Megoldhatóság:

Bizonyított, hogy ha vesszük az elemeket az ábrán látható sorrendben, akkor ennek a sorozatnak az inverziószáma minden lépés (csúsztatás hatására) páros számmal változik, tehát a paritása állandó. Így ki tudunk zárni már az elején lehetetlen helyzeteket, ám ami még fontosabb ez egy szükséges és elégséges feltétel. Tehát amennyiben a kezdőállapot és a végállapot ezen módon számított értéke, megegyezik, úgy létezik is a kezdőből a végállapotba vezető lépéssorozat.



Megoldás:

Amennyiben a probléma megoldható, úgy egy A* segítségével próbál a program egy megoldást találni. A heurisztikus *h()* függvény adott állapotra kiszámolja a mezők végső állapothoz képesti manhattan-távolságainak az összegét. A következő vizsgált állapot eldöntése a *h()* és a *cost()* függvény összegén alapszik, ahol a *cost()* az a kezdőállapotból tett lépések száma (az adott állapotba vezető lépések száma). Viszont a fenti heurisztika esetében sok állapotot vizsgálunk, kifejezetten lassú lesz az útvonalkeresés (cserébe relatív kevés lépésből álló megoldást keres). Ezen javíthatunk, ha a *h()* függvény dominánsabbá tesszük, a programban 30-al felszoroztam az értékét, és ez alapján választjuk az új állapotot. Ezáltal gyorsabban találunk megoldást, viszont több lépésben. A 30-as szorzó esetében 200-250 lépésből álló megoldások a leggyakoribbak.

Technikai optimalizáció:

Mivel 16 elemű a tábla és a táblán levő értékek a [0..15]-ből kerülnek ki, így egy adott állapot eltárolásához 16*4=64 bit elegendő. Emiatt egy 64 bites integer számként tárolja a program a tábla állapotát, ami 64 bites architektúrán könnyen, gyorsan ábrázolható/módosítható, illetve sok optimalizálhatósági lehetőséget ad a fordítónak. Ez által az állapotváltoztató műveletek is visszavezethetőek bitműveletekre.

Használati útmutató:

A program indításakor rákérdez, hogy a megoldandó játékállapotot véletlenszerűen generálja, vagy egy fájlból olvassa. Utóbbi esetben a fájl nevét is meg kell adnunk. Ezek után a megoldási lépések grafikusan is szemléltetésre kerülnek, a talált lépéssorozatot 30 mp alatt végigmutatja a program. Ezen felül a standard outputra is kiírásra kerül a kiindulási állapot, a megoldás megtalálásához szükséges idő, illetve a talált megoldás által tett lépések száma.

Mérési eredmények:

2000 elemű véletlenszerűen generált – de csak megoldható kiindulási állapotot tartalmazó – mintán futtatva a programot következő eredményeket kapjuk a futási idő és a lépések számára:

Futási idő (ms):

1. 98%-os kvantilis: 3973
2. 90%-os kvantilis: 1157
3. 70%-os kvantilis: 492
4. 50%-os kvantilis: 224

Lépések száma (db):

1. 98%-os kvantilis: 305
2. 90%-os kvantilis: 267
3. 70%-os kvantilis: 221
4. 50%-os kvantilis: 186