АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА BUBBLES

Задачата беше нестандартна по няколко причини.
Първо, входът и изходът бяха сравнително нетипични. Това беше направено с цел състезателите да не бъдат ограничени откъм Time Limit – примерно авторското решение обработва всички тестове за цели 10 минути. Друг плюс от това беше, че така има full feedback на релативна задача – нещо, което иначе не се поддържа от системата.
Второ, задачата беше очевидно NP-пълна. Такива задачи се срещат почти всяка година на IOI в последно време, като в същото време са интересни и представят поле за изява на креативност.

Някои от състезателите, които миналата година са били в разширения отбор може би помнят задачата Tetris от третата контрола за национален отбор. Забавното е, че освен сравнително сходна постановка (някакви неща падат и изчезват – тук е наобратно, изчезват и после падат), авторското решение също е доста подобно. Миналата година никой не го измисли, като максималните точките по задачата бяха далеч от задоволителни (максимумът беше 53, но мнозинството бяха около 20). Тази задача може да послужи за проверка дали някой си е направил труда да прочете анализа :)

А сега решението! Първо ще разглеждаме дъската като "блокове" или "островчета" от еднакви букви. Избирайки коя да е клетка от дадено островче го премахва цялото.
В самото условие на задачата е намекнато, че по-големи блокове носят повече точки. Логично решение е всеки път да избираме най-големия блок! Всъщност това решение не е много лошо и би донесло около 50 точки (45 в случая на моята имплементация) на състезателите. Ако състезателите се усетят, че някои от тестовете са достатъчно малки за да направят с пълно изчерпване, а дори на ръка, биха могли да спечелят допълнителни 10-15 точки.

Условието намеква, че понякога е хубаво да създаваме по-големи блокове. Тъй като получените точки са големината на блока на квадрат, то можем да жертваме някой ход (в който махнем малко островче) за да може след това да получим по-голям блок.

Това вече е доста сложен проблем, който обикновено се решава с апроксимационни алгоритми. Един вариант е да правим бектрек, в който пробваме всички възможности, като така бихме намерили оптимално решение на задачата... за адски много време. Дори 5-те часа няма да са близо до достатъчни за да мине това решение. Така че трябва да направим компромис – не изцяло перфектно решение, но все пак гледащо известен брой ходове напред.

Чистият бектрек трудно може да гледа повече от 2-3 хода напред. Една оптимизация е Alpha-beta отсичане, която реже дървото и би могла да разреши още един или два хода напред.

Проблемът на бектрекът е, че броя възможности, които трябва да разгледаме, расте експоненциално с всяка следваща стъпка напред, която искаме да разгледаме. Точно това ще решим чрез използването на Beam Search: http://en.wikipedia.org/wiki/Beam_search.

Вместо на всяка стъпка да пускаме рекурсията от *всяка* възможна конфигурация, ще я пускаме само от най-обещаващите такива. Например на даден ход може да се получат стотици възможни продължения, но ние ще ги сортираме по някаква евристична оценка и ще продължим само от K-те най-добри (в тази задача K е удачно да бъде около 20, тъй като твърде малко K води до твърде greedy решение, докато голямо K не разрешава голяма дълбочина на търсенето).

Каква евристика да ползваме? Доста очевидно решение е просто да ползваме грийдито за евристика. Знаем, че то не е перфектно (нали точно него се опитваме да оправим!), но пък е сравнително бързо и дава относително задоволителни резултати. Това е един сравнително добър избор.

Моето решение обаче прави дори по-глупава евристика с цел тя да бъде по-бърза и да позволи малко по-дълбоко търсене (5-6-7 нива навътре). Вместо да симулирам как дъската се променя след премахването на най-големия остров (както бихме направили в грийдито), аз просто намирам големината на всички острови на дъската, сортирам ги по големина и сумирам най-големите K, ако са ми останали K хода. Това не е акуратно, тъй като може да даде резултат, който е непостижим на практика, но се оказва, че е достатъчно близо до истинския резултат за да работи добре.

Няма да описвам алгоритъма в големи детайли (ако искате, вижте задачата Tetris: http://www.math.bas.bg/infos/files/2012-06-14-A2.pdf и нейния анализ: http://www.math.bas.bg/infos/files/2012-06-14-sol-A2.pdf).

Автор: Александър Георгиев