**Camp**
CodeIT – Сезон 2014/2015 – III Маратон
Решение на Енчо Мишинев, Ямбол

Задачата изисква да намерим общ подграф между два графа с максимално тегло, но в добавка имаме и Wi – тегла на върховете.

Първа стъпка от решението е да премахнем параметъра **Wi**. Разглеждаме формулата за оценяване:

F = (сумата от Cij за всички съседи) + (сумата от произведението (Wi\*(броя съседи на i)), за всеки ученик от отбора на Иванчо)

Забелязва се, че за всяко ребро, излизащо от i в графа-решение, ще добавяме Wi към оценката. Следователно, ако предварително увеличим теглото на всички ребра излизащи от i с Wi, то оценката ще се запази и можем да търсим просто общ подграф с максимално тегло.

Следващата стъпка е да помислим за каквото и да е решение. Дори самата задача за проверка дали два произволни графа са изоморфни е нерешима в полиномиално време. Затова е естествено голяма част от алгоритъма ни да разчита на рандомизация и множество итерации.

Едно решение е да пуснем DFS в двата графа едновременно по следният начин:

Избираме на случен принцип едно бунгало и един ученик и настаняваме избраният ученик в избраното бунгало. След това правим два списъка – със съседите на текущия ученик от графа с приятелствата и със съседите на текущото бунгало от графа с бунгала. Избираме случайно неизбирано до сега бунгало от списъка.

Избирането на приятел има два различни подхода и води до две различни решения. Единият вариант е да изберем случен неизбран приятел от списъка, а другият - да изберем неизбиран приятел с най-голяма стойност на реброто сочещо към него.

След избирането на приятел от списъка настаняваме избрания приятел в избраното бунгало и продължаваме рекурсивно за тази двойка. Повтаряме избирането на двойки от списъците докато вече няма валидни неизбрани двойки.

През работата на алгоритъма внимаваме никой връх да не надхвърли лимита си **Di**.

Алгоритъмът наподобява DFS и реално след завършването му ние сме намерили някакво общо поддърво. След това правим списък с всички ребра, които имат избрани ученици и в двата края, но самите те не са избрани. Сортираме ги по намаляваща цена и пробваме да ги добавяме един по един, като следим ограничението **Di**.

Този greedy подход, комбиниран с множество итерации на горните два варианта на DFS, води до добри резултати и над 80% от точките за примерите, в които нито G1, нито G2 са дървета.

Когато G1 и/или G2 са дървета има много по-добри подходи.

Да разгледаме случая когато G1 е дърво. В горния алгоритъм спазването на ограничението **Di** е чрез greedy подход. Ако G1 е дърво, обаче, можем да направим това по оптимален начин.

В момента, в който изберем начален ученик с начално бунгало, можем да коренуваме G1 със корен този ученик. След което искаме да решим задачата „кои ребра да се отрежат, така че в дървото, коренувано на G1, да са спазени ограниченията **Di** и да има максимална обща тежест”.

Да дефинираме:

*F[k]=оптималната тежест на поддървото на k при спазени ограничения D.*

Очевидно отговорът, който търсим, е F[root], като *root* е избрания корен. Лесно се вижда как можем да пресметнем тези стойности. Ако искаме да сметнем F[k] и p1, p2, p3 ... pm са децата на k, то просто взимаме **D[k]-1** (понеже е свързано и с баща си) деца с максимални стойности на {F[pi]+ребро към pi} освен, ако сме в корена – тогава взимаме **D[k]** стойности.

След това „подрязване” на дървото, нормалните алгоритми, използващи DFS подхода, стават много по-ефективни, понеже не се съобразяват с нарушаването на ограничението за **Di**. Това решение взима 100% от точките за примери, в които G1 е дърво, а G2 не е.

Ако G1 и G2 са дървета, то е възможно и по-добро решение.

Като за начало, отново избираме по един връх от всеки граф случайно и „подрязваме” G1, за да не се съобразяваме с **Di**. След това коренуваме дърветата на избраните върхове и за всеки връх пресмятаме големината на поддървото му (брой върхове). След това пускаме подхода чрез DFS в двата графа, но вместо да избираме случайни/максимално тежки ребра, избираме ребрата към върховете с най-големи поддървета. Така настаняваме ученик с голямо поддърво в бунгало с голямо поддърво, което позволява настаняването на още много ученици.

Очевидно това решение се опитва да максимизира броя ученици, а не тежестта на ребрата, но понеже успява да настани много повече ученици от другите решения, то е много по-добро на практика. Решението взима 100% от точките за примери, в които G1 и G2 са дървета.