Магьосник (анализ)

A5: 320

Решението на тази задача се състои от две части - геометрия и максимално двойкосъчетание в двуделен граф. Задачата като цяло е на най-много фигури да съпоставим на всяка фигура по една дупка като на дупка не можем да съпоставим повече от една фигура.

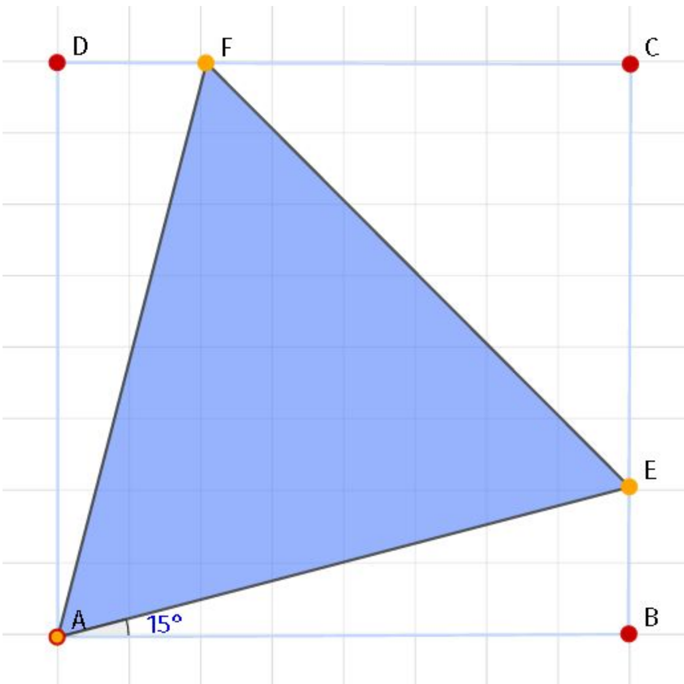
Част 1 - геометрия

С помощта на геометрията, естествено, можем да разберем кои фигури в кои дупки могат да се поберат. Очевидно е че ако формата на фигурата е същата като тази на дупката, трябва размерът на фигурата да е по-малък или равен на този на дупката.

Когато за различни, обаче, размерът на фигурата трябва да е по-малък или равен от този на фигура, вписана в дупката. Това се намира чрез тригонометрия. Нека *a* е страната/диаметърът на фигурата и *A* е страната/диаметърът на дупката. Тогава за да може да се намести фигура в дупка, трябва да е изпълнено:

- ако фигурата е триъгълник, а дупката - квадрат: $a \le \left(\sqrt{6} - \sqrt{2}\right)A$
- ако фигурата е триъгълник, а дупката - кръг: $2a \le \sqrt{3}A$
- ако фигурата е квадрат, а дупката - триъгълник: $a \le \left(2\sqrt{3} - 3\right)A$
- ако фигурата е квадрат, а дупката - кръг: $2a \le \sqrt{2}A$
- ако фигурата е кръг, а дупката - триъгълник: $\sqrt{3}a \le A$
- ако фигурата е кръг, а дупката - квадрат: $a \le A$

Следното изображение показва какъв е най-големият триъгълник, който може да се събере в квадрат:



Част 2 - максимално двойкосъчетание (max bipartite matching)

След като знаем кои фигури в кои дупки могат да се съберат, можем да направим съчетанието. Нашият двуделен граф ще бъде граф, където има ребро между фигура и дупка само ако фигурата може да се събере в нея и ако е в сила условието **големината на дупката е с най-много D по-голяма от големината на фигурата**.

Оттук задачата е тривиална - трябва да се напише алгоритъма на Хопкрофт-Карп (Hopcroft-Karp algorithm) - сравнително прост и бърз, но работи само за двойкосъчетание; има сложност $O\left(E\sqrt{V}\right)$, където V е броят на върховете (V = A+B+C+P+Q+R) и E е броят на ребрата (E <= (A+B+C)*(P+Q+R)).

Може да се напише и алгоритъм за намиране на най-голям поток (max-flow), например на Диниц(Dinic), но той е по-бавен, отнемайки $O\left(V^2E\right)$ време.