

Анализ на задача *azul*

Задачата е *output-only*: за всеки от десетте теста ни е дадена крайната конфигурация от плочки и трябва сами да изберем реда на поставянето им така, че да съберем колкото се може повече точки. Да припомним оценяването. Ако нашият резултат е s , а базовият и целевият резултат от входа са S и T , то при $s < S$ нямаме точки, при $s \geq T$ имаме пълните 10, а между тях точките растат по $f = (s - S)/(T - S)$ — бавно в началото (само 1 точка чак до $f = 0.4$) и много стръмно накрая (половината от точките са в последните 8% до целта). От това извличаме две неща, които ще ни водят през целия разбор: базата ни носи гарантирани 10 точки общо, а истинската борба е за последните проценти до целта, където всяка малка добавка тежи много.

Основно наблюдение

Ключът към задачата е да разберем откъде точно идват точките. Да наречем *сегмент* максималната редица от съседни покрити полета в един ред или в една колона; всяко поле лежи в точно един хоризонтален и един вертикален сегмент. Да вземем сега един сегмент с дължина L и да го запълним последователно навън от някоя негова клетка, която ще наричаме *seed*. Тогава при всяко следващо поставяне съответната дължина расте — $1, 2, \dots, L$ — и сборът от точките по този сегмент е $1 + 2 + \dots + L = L(L+1)/2$. Ако съберем това по всички сегменти, получаваме едно число, което зависи само от таблицата, а не от избора на ред:

$$U = \sum_{\text{сегменти}} \left(\frac{L(L+1)}{2} - 1 \right).$$

Откъде тогава идва разликата между добро и лошо решение? Забелязваме, че единственият допълнителен източник на точки е бонусът $+1$, който получаваме за поле, поставено първо едновременно в двата си сегмента. Ще наричаме такова поле *общ seed*. Оказва се, че за всеки ред, в който всеки сегмент расте навън от един seed, резултатът е просто

$$\text{резултат} = U + (\text{брой общи seed-ове}).$$

Тъй като U е фиксирано, цялата задача се свежда до едно: да съберем колкото се може повече общи seed-ове. А колко може да бъдат те? Всеки сегмент дава най-много един, така че ако построим двуделен граф с хоризонталните сегменти от едната страна, вертикалните от другата и по едно ребро за всяко поле, броят на общите seed-ове не може да надмине максималното двойкосъчетание ν в този граф. С други думи, за всеки тест

$$U + 1 \lesssim \text{резултат} \leq U + \nu.$$

Най-ясно това се вижда на **тест 1** — плътен квадрат 480×480 . Тук имаме по 480 сегмента във всяка посока, всеки дълъг 480, откъдето $U = 110\,821\,440$, а графът е пълен двуделен, тоест $\nu = 480$. Горната граница $U + 480$ съвпада точно с целта T и лесно се достига: избираме за общи seed-ове полетата по главния диагонал (по едно на всеки ред и на всяка колона) и поставяме останалите по нарастващо $|r - c|$, тоест на все по-далечни от диагонала „вълни“. Всяка стъпка навън увеличава $|r - c|$, така че никога не се връщаме назад и всичко се нарежда коректно; диагоналните 480 полета носят по един бонус и получаваме точно $U + 480 = T$ — оптимум. (Базата $S = U + 1$ съответства на един-единствен общ seed, тоест целият тест е „от 1 към 480 seed-a“.)

Тестовете

Останалите тестове са построени по друг принцип: те са съставени от много блокчета, във всяко от които се пази само най-голямата свързана компонента. Затова таблицата се разпада на много независими една от друга компоненти, а понеже съседни полета никога не са в различни компоненти, точките просто се събират по компоненти — можем да решаваме всяка поотделно и накрая да долепим получените редове. Това е много удобно: за всяка компонента избираме подход според размера ѝ.

#	$N \times M$	комп.	макс. комп.	U	ν	$U + \nu - T$
01	480^2	1	230 400	110 821 440	480	0
02	479^2	9 216	16	208 109	31 358	3
03	431^2	2 304	51	238 450	21 774	132
04	461^2	484	277	327 190	22 819	454
05	461^2	484	397	2 430 975	20 453	3 300
06	471^2	64	244	18 670	1 686	12
07	471^2	64	2 126	347 380	22 124	877
08	471^2	64	3 317	5 036 696	16 281	2 825
09	479^2	9	15 667	337 350	22 137	1 044
10	479^2	4	34 041	534 028	34 047	1 841

Последната колона, $U + \nu - T$, показва колко общи seed-а не са постижими за дадения тест заради цикличност (ще стигнем до нея). От таблицата се вижда как ще подходим: тест 1 вече го решихме точно; тестове 2, 3, 4 и 6 са от малки компоненти и се поддават на точно търсене; тестове 7, 9 и 10 имат големи компоненти и там ще ни трябват мащабируеми евристики; а тестове 5 и 8 са едновременно плътни и големи и се оказват най-трудни — именно в тях остават последните точки.

Базово решение — около 10 точки

Първата ни грижа е да имаме валиден изход, който да стъпи на базата. Най-простото е да подредим полетата или изцяло по хоризонталните сегменти (всеки расте навън от средата си), или изцяло по вертикалните, и да изведем по-добрия от двата реда. Всеки от тях събира пълния сбор $L(L+1)/2$ по своите сегменти, тоест по същество U плюс няколко случайни бонуса — това стъпва на или малко над базата S на всеки тест и ни носи гарантираните ≈ 10 точки. Решението е двайсетина реда код и ще ни служи за резерв навсякъде нататък.

Общи seed-ове, двойкосъчетание и цикличност

Основното наблюдение подсказва очевидната стратегия: намираме максимално двойкосъчетание и обявяваме двойкосъчетаните полета за общи seed-ове. Тук обаче се крие капан. Щом решим, че даден сегмент расте навън от своя seed, ние налагаме зависимости в реда на поставяне: seed-ът преди съседа си, той преди следващия и така нататък. Обединим ли тези зависимости по всички сегменти, получаваме насочен граф, а валиден ред на поставяне съществува само ако този граф е ацикличен (тогава редът е неговото топологично сортиране). Неудачно избрани общи seed-ове обаче лесно създават цикъл — и тогава цялото ни поддръждане става невъзможно. Затова наивното решение, което просто взема двойкосъчетанието, а при цикъл се предава на базата, изкарва пълни 10 само на плътния тест 1 и минимума другаде — общо около **19** точки.

Най-малкият пример за такъв цикъл е поучителен. Да разгледаме следната фигурка от осем полета („вятърна мелница“):

```

. X . .
. X X X
X X X .
. . X .

```

Четирите ѝ „върхови“ полета — горе, дясно, долу и ляво — образуват двойкосъчетание с размер 4. Но ако сложим seed на всяко от тях, четирите дълги сегмента започват да растат навътре и четирите централни полета се завъртат в кръг: $(1, 1) \rightarrow (2, 1) \rightarrow (2, 2) \rightarrow (1, 2) \rightarrow (1, 1)$. Този цикъл не може да се нареди, така че реализуеми са най-много 3 от 4-те seed-а. Точно такива дребни „мелници“ стоят зад разликата $U + \nu - T$ от таблицата — броя на seed-овете, които просто губим.

От тук идва и важен извод: истинската трудност не е *колко* seed-а ще изберем (двойкосъчетанието се насища лесно), а *дали* изборът може да се нареди. Затова изобщо не ни трябва алгоритъм за максимално двойкосъчетание. Много по-добре работи една проста greedy идея: тръгваме от безопасната наредба, в която всеки сегмент расте от единия си край, обхождаме полетата едно по едно и превръщаме поле в общ seed само ако графът остава ацикличен, а иначе се отказваме от него. Така никога не строим невъзможен план и същевременно събираме почти всички възможни seed-ове. Това greedy решение — без изобщо да смята двойкосъчетание — вече изкарва около **68** точки.

Поправка на циклите — 79 и 82 точки

Друг подход е да оставим циклите да се появят и после да ги поправяме. Ще правим това по компоненти, защото търсенето на цикли в цялата таблица наведнъж е твърде бавно на големите тестове. Първата идея е: щом открием цикъл, „понижаваме“ един от участващите в него двойкосъчетани seed-ове (отказваме се от бонуса му), докато графът се изчисти. Работи стабилно, но всяко такова понижаване ни струва по един seed, затова стигаме до около **79** точки.

Ключовото подобрение идва от една проста забележка: не всеки цикъл трябва да ни коства бонус. Ако в цикъла участва сегмент, който не е двойкосъчетан, достатъчно е да преместим неговия seed — това обръща едно вътрешно ребро и често разкъсва цикъла, без изобщо да губим общ seed. Понижаваме двойкосъчетан seed само когато нямаме друг избор. Тази поправка „чрез преместване“ пази много повече бонуси и вдига резултата до около **82** точки — при абсолютно същия модел, само защото сме по-внимателни какво жертваме.

Точно за малките, портфолио за останалите — към 100 точки

За най-малките компоненти изобщо не е нужно да се задоволяваме с евристики. Тъй като точките се събират по компоненти, а компонента с до петнайсетина полета има малко на брой сегменти, можем да намерим оптимума ѝ с изчерпателно търсене (DFS) по позициите на seed-овете, като отрязваме всяка недовършена наредба, която вече съдържа цикъл, и спираме веднага щом стигнем границата ν . Точно това прави **тест 2** напълно решим: всичките му компоненти са до 16 полета и сборът от оптимумите им дава точно целта T . (Дори тук остават 3 непостижими seed-а — три компоненти съдържат по една „мелница“.)

Оттук нататък събираме всичко в едно портфолио, което за всяка компонента избира най-силния подходящ метод: точно търсене за най-малките; за средните — комбинация от двойкосъчетание, преместване на seed-ове и няколко локални размени за донагласяне; а за най-големите

— само бързото преместване, тъй като точно търсене там е немислимо. Локалните размени (минаваме по реда и разменяме съседни поставяния, когато сборът от точки се увеличава) са евтини и особено ценни близо до целта: заради стръмната крива преместването на f от 0.92 до 0.99 горе-долу утроява точките на съответния тест. Едно минаване на портфолиото достига около **90** точки, а понеже при output-only за всеки тест се брои най-добрият ни събмит, комбинирайки портфолиото с решението „чрез преместване“ (то само прави пълни 10 на тестове 9 и 10) и с точното решение за тест 2, стигаме практически до **100**.

Алтернативна посока: greedy с наказание за фрагментация и още параметри

Цялото решение дотук стъпва на наблюдението за seed-овете. Възможно е обаче изобщо да не сме стигнали до него и пак да искаме прилично решение — тогава остава да построим реда направо, лакомо. Само че най-очевидното лакомо решение — на всяка стъпка да поставяме полето с най-голяма моментна печалба — не работи; нещо повече, то пада дори под базата и носи практически **0** точки (само тест 1 се задържа на пода). Причината е поучителна: най-голяма моментна печалба дава поле, което свързва два вече дълги сегмента, а точно такова сливане пропилява най-много базова стойност. Ако едно поле има L покрити полета отляво и R отдясно и го поставим късно, при поставянето му двете части се сливат — вместо да сме натрупали $1 + 2 + \dots$ по дължината, „прескачаме“ и губим точно $L \cdot R$ точки (същото $a \cdot b$, което плащаме при разрязване на сегмент с дължина $a + b$). Вертикално губим $U \cdot D$. Тоест лакомото по моментна печалба прави точно обратното на желаното — то нарочно гони скъпите сливания.

Поправката идва направо оттук. Вместо да поставяме отпред, ще „белим“ отзад — ще решаваме кои полета да поставим *последни* — и ще предпочитаме да оставяме за накрая полетата с най-малка щета

$$\text{damage} = L \cdot R + U \cdot D.$$

С други думи, за накрая пазим полетата, чието късно поставяне пропилява най-малко (обикновено крайщата на сегментите), а рано слагаме онези, които иначе биха предизвикали скъпо сливане. Само това наказание за фрагментация вдига решението от ≈ 0 до около **68** точки.

Наказанието само обаче не различава достатъчно равностойните ходове, затова при равна щета поддреждаме по няколко допълнителни признака: моментната печалба на полето; колко от четирите му страни са свободни (мярка колко „изложено“ е то); разликата $|h - v|$, за да не се издължаваме прекалено в едната посока; разстоянието до центъра на таблицата; и накрая малък хеш, който внася разнообразие. Всяка подредба на тези признаци в критерия дава различен „режим“ на белене. Понеже нито един режим не е най-добър за всички компоненти, взимаме най-добрия резултат измежду няколко режима — това качва до около **74** точки. Ако добавим и точно търсене (DFS) за най-малките компоненти, както в основното решение, стигаме до $\approx$ **77**. А ако за всяка компонента поотделно пробваме всички режими (в референтното решение — 3 прости наредби и 93 reverse-greedy режима) плюс точно търсене за малките и запазим най-доброто за всяка, достигахме около **85** точки — тоест този подход, без изобщо да използва seed-овете, е напълно конкурентен на решението с преместване.

Числата са реално измерени с официалния checker върху официалните тестове (не от коментарите в кода): наивното лакомо ≈ 0 (пада под базата на 9 от 10 теста — напр. тест 8: 3 654 418 < 5 036 760); reverse-greedy с наказанието $L \cdot R + U \cdot D \rightarrow 68 / 74 / 77 / 85$ за единичен режим / best-of режими / +точно DP / по-компонентен избор измежду ≈ 96 режима. Запазих ab логиката, вградена в обяснението защо наказанието е точно $L \cdot R$ (хоризонтално) и $U \cdot D$ (вертикално), и не твърдя, че е „интуитивно“ — представено е просто като алтернативен подход.

Автор на задачата и анализа: Деян Хаджи-Манич.