Вказівки щодо розв'язання завдання № 26 відбірково-тренувальних зборів команди міста Києва

1. Шахи

Спочатку навчимося визначати кількість ігор при сталому D. Для цього побудуємо дводольний граф, вершини з однієї долі якого відповідатимуть учням з сільської команди, а з іншої — учням зі столиці. Якщо два учні можуть грати між собою при даному D, проведемо ребра між відповідними вершинами графа. Мовою графів задача звучить так: знайти максимальний за кількістю набір ребер, у якому жодні два ребра не мають спільної вершини. Інакше кажучи, потрібно знайти максимальну сполуку пар дводольного графа. Алгоритм такого пошуку див., наприклад, http://kievoi.narod.ru/lectures/pair.html.

Позначимо:

$l_{\min}$ — найменший з рівнів гри серед членів сільської команди;

$l_{\max}$ — найбільший з рівнів гри серед членів сільської команди;

$L_{\min}$ — найменший з рівнів гри серед членів міської команди;

$L_{\max}$ — найбільший з рівнів гри серед членів міської команди.

Доведемо, що при $D = A < L_{\min} + d - l_{max}$ відповідь буде не більшою, ніж при $D = B = L_{\min} + d - l_{max}$. Для цього достатньо довести таке: якщо ребро між двома вершинами присутнє у графі при $D = A$, то воно присутнє й у графі при $D = B$. Доведемо це від супротивного. Нехай ребро присутнє у графі при $D = A$, але відсутнє у графі при $D = B$. Нехай це ребро між вершинами i та j, які відповідають гравцям рівня a з сільської команди та b з міської, відповідно. Маємо: $|a + B - b| > d$ та $|a + A - b| \le d$. Перша нерівність рівносильна сукупності: $a + B - b > d$ або $a + B - b < -d$. Другу нерівність перепишемо так: $-d \le a + A - b \le d$. Якщо $a + B - b < -d$, то $a + B - b < a + A - b$. Звідси $B < A$, що суперечить нерівності: $A < B$. Отже, $a + B - b > d$. Маємо:

$$a + B - b = a + L_{\min} + d - l_{max} - b < (a - l_{max}) + d + (L_{\min} - b) \le d.$$

Отримали суперечність: $d < a + B - b \le d$. Отже, при $D < B$ максимальна сполука пар буде не більшою, ніж при $D = B$.

Доведемо, що відповідь буде не більшою при $D = A > L_{\max} - d - l_{\min}$, ніж при $D = B = L_{\max} - d - l_{\min}$. Для цього достатньо довести таке: якщо ребро між двома вершинами присутнє в графі при $D = A$, то воно присутнє і у графі при $D = B$. Доведемо це від супротивного. Нехай ребро присутнє у графі при $D = A$, але відсутнє у графі при $D = B$. Нехай це ребро між вершинами i та j, які відповідають гравцям рівня a з сільської команди та b з міської, відповідно. Маємо: $|a + B - b| > d$ та $|a + A - b| \le d$. Перша нерівність рівносильна сукупності: $a + B - b > d$ або $a + B - b < -d$. Другу нерівність перепишемо так:

$-d \le a + A - b \le d$. Якщо $a + B - b > d$, то $a + B - b > a + A - b$. Звідси $B > A$, що суперечить нерівності $A > B$. Отже, $a + B - b < -d$. Маємо:

$$a + B - b = a + L_{\max} - d - l_{\min} - b > (a - l_{\min}) - d + (L_{\max} - b) \ge -d.$$

Отримали суперечність $-d \le a + B - b < -d$. Отже, при $D > B$ максимальна сполука пар буде не більшою, ніж при $D = B$.

Найпростіше розв'язання таке. При $L_{\min} + d - l_{\max} \le L_{\max} - d - l_{\min}$ перебираємо усі значення D у межах від $L_{\min} + d - l_{\max}$ до $L_{\max} - d - l_{\min}$ включно, інакше розглянемо лише $D = L_{\min} + d - l_{\max}$. Для кожного такого D будуємо дводольний граф і знаходимо максимальну сполуку пар у ньому. Складність алгоритму — $O(C(N + M)^{2.5})$, де C — кількість різних можливих значень D.

Зауважимо, що не обов'язково перебирати усі можливі значення D, бо є такі, при яких граф залишається одним і тим самим.

Нехай $left[i]$ — рівень гравця з номером i з сільської команди, а $right[j]$ — рівень гравця з номером j зі столичної команди. Ребро між відповідними вершинами існує тоді й лише тоді, коли $|left[i] + D - right[j]| \le d$. Розкривши знак модуля, отримаємо: $right[j] - left[i] - d \le D \le right[j] - left[i] + d$.

Запровадимо такі позначення:

$$a_{ij} = right[j] - left[i] - d;$$
$$b_{ij} = right[j] - left[i] + d.$$

Умова існування ребра має вигляд: $D \in [a_{ij} ; b_{ij}]$. Зафіксуємо значення D та розглянемо усі такі відрізки $[a_{ij} ; b_{ij}]$, яким воно належить. Тоді граф містить лише ті ребра, які відповідають цим відрізкам. Нехай D не належить краю жодного з відрізків. Тоді ми можемо його зменшувати або збільшувати доти, доки не досягнемо кінця якогось відрізка. Граф при цьому не змінюється. Тому доцільно розглядати лише ті значення D, які дорівнюють a_{ij} або b_{ij} для певних i та j.

Отримуємо розв'язання: перебиратимемо усі ті значення D, що є кінцями відрізків $[a_{ij}; b_{ij}]$. Для кожного такого значення будуємо дводольний граф і шукаємо максимальну сполуку пар у ньому. Складність алгоритму — $O(NM(N+M)^{2.5})$.

Таке розв'язання значно швидше за попереднє, але при максимальних величинах параметрів задачі не вкладеться в обмеження за часом. Щоб прискорити його, будемо перебирати усі значення D у порядку їх зростання, зробивши кілька зауважень.

Що станеться, якщо ми збільшимо D? Для деяких відрізків D стане більшим за координату їхнього правого кінця, а для деяких D стане не меншим за координату лівого кінця. Інакше кажучи, з графа видалиться деяка (можливо, нульова) кількість ребер, а також деяка (можливо, нульова) кількість ребер додасться.

Що станеться з максимальною сполукою пар при додаванні одного ребра? Вона або не зміниться, або збільшиться на 1. Аналогічно, при видаленні ребра кількість пар або не зміниться, або зменшиться на 1.

Зауважимо: не обов'язково після додавання (видалення) кожного ребра одразу змінювати максимальну сполуку пар. При збільшенні D може статися, що ми кілька разів підряд долучаємо або вилучаємо ребро. Можна виділити усі такі послідовні однотипні зміни, назвавши їх блоками, та шукати максимальну сполуку пар лише по завершенню блока. Більше того, доцільно шукати максимальну сполуку пар лише у кінці кожного блоку, у якому додають ребра, бо після видалення ребер кількість пар не збільшується.

Остаточне розв'язання таке.

1. Знайдемо усі можливі значення a_{ij} та b_{ij} і впорядкуємо їх за неспаданням. При збігу двох величин спочатку поставимо ліві кінці відрізків a_{ij}, потім — праві кінці відрізків b_{ij}. Для кожної такої величини запам'ятаємо величини i, j та чи є дана величина лівим кінцем.
2. Виділимо ті значення, які завершують блоки послідовних лівих кінців.
3. Розглянемо впорядковану послідовність як послідовність значень D. При переході до нового значення змінюємо граф: долучаємо або вилучаємо ребро. Кожен раз після опрацювання останнього елемента блоку лівих кінців шукаємо максимальну сполуку пар та порівнюємо її з уже знайденою відповіддю. Для її пошуку використовуємо вже знайдену максимальну сполуку пар, що залишилася після розглядання попереднього блоку, та за допомогою серії пошуків в глибину покращуємо її.

Складність такого алгоритму становить $O(NM \log(NM))$ для впорядкування та $O(N^2M^2)$ для основного алгоритму. При найбільших значеннях N і M маємо: $NM \log(NM)+N^2M^2 \approx 1{,}6\cdot10^9$, але за рахунок малого коефіцієнта пропорційності розв'язання вкладається в обмеження за часом.

2. Сервери

Рівень 1. Оптимізований перебір перестановок.

Рівень 2. Використання рекурентних співвідношень. Нехай S — множина усіх серверів, Q — довільна підмножина S. Позначимо через $f(Q)$ найменшу загальну довжину кабелю при розташуванні усіх серверів з Q на місцях $1, 2, \ldots, |Q|$, а серверів з $S \setminus Q$ на місці $|Q|$. Тобто на місці $|Q|$ стоятиме $|S \setminus Q| + 1$ сервер. Це суперечить умові при всіх $Q \neq S$. Проте $f(S)$ відповідатиме абсолютно правильному розташуванню серверів, при якому загальна довжина використаного кабелю найменша. Маємо таке рекурентне співвідношення:

$$f(Q) = \min\{f(Q\setminus\{x\}) + T(Q\setminus\{x\}) \mid x \in Q\},$$

де $T(A)$ — кількість з'єднань між серверами з множини A і серверами з множини $S \setminus A$.

Кожну підмножину Q множини S можна подати числом x у межах від 0 до $2^N - 1$ включно: сервер j належить до підмножини тоді й лише тоді, коли j-та цифра двійкового запису x дорівнює 1. Тому рекурентну формулу можна записати й використовувати для цілих чисел x у межах від 0 до $2^N - 1$ включно.

Таким чином було створено перше авторське розв'язання servers-deterministic.cpp з оцінкою ефективності $O(2^N N^2)$. Ефективність алгоритму можна легко покращити до $O(2^N M)$. Це розв'язання використано для перевірки коректності тестів.

Рівень 3. Стохастичний пошук: починаючи з певної перестановки a її послідовно покращують:

(1) розглядають усі ті перестановки, які відрізняються від a лише розташуванням *двох* елементів;

(2) з усіх таких перестановок обирають одну з тих, у яких найменша загальна довжина кабелю.

Дії (1) – (2) повторюють доти, поки перестановку a неможна буде покращити переставлянням двох елементів.

Таке розв'язання називають *градієнтним спуском*. Воно гарантує, що знайдений розв'язок є лише локально найкращим, і не гарантує, що він буде глобальним мінімумом, тобто шуканим розв'язком задачі. Перевагою цього методу є те, що він діє швидко і просто програмується.

Для того, щоб побороти проблему локальності мінімуму, можна виконувати таке послідовне покращення велику кількість разів з різними початковими перестановками, вибраними *випадковим* чином. Саме тому таке розв'язання називають *стохастичним*.

Було розглянуто дуже багато варіантів вхідних даних (усі неізоморфні графи з кількістю вершин, що не перевищує 8 і кілька тисяч графів з більшою кількістю вершин і ребер). Для кожного варіанту було запущено описану вище процедуру покращення 10000 разів з випадково вибраною початковою перестановкою. Було визначено, у скількох наближеннях вдалось досягнути глобального мінімуму. Не було знайдено жодного варіанту, при якому *емпірична* ймовірність *не* прийти до глобального мінімуму, перевищувала б 99%. Тоді $0{,}99^{10000} \approx 2{,}24877 \cdot 10^{-44}$ — обмеження згори на ймовірність не

отримати глобальний мінімум *у жодному* з 10000 *незалежних* випробувань з випадково вибраною початковою перестановкою.

При калібруванні системи тестування правильний результат досягався в усіх тестах не більше ніж за 0,50 сек у результаті 650 *незалежних* випробувань з випадково вибраною початковою перестановкою. У цьому випадку обмеження згори на ймовірність не отримати глобальний мінімум *у жодному* з розглянутих випадків складає $0{,}99^{650} \approx 0{,}00146$. Насправді учасники мали значний «запас міцності»: для розгляду 1250 незалежних випробувань авторському розв'язанню було потрібно не більше 0,94 сек при обмеженні згори на ймовірність не отримати глобальний мінімум $0{,}99^{1250} \approx 3{,}5 \cdot 10^{-6}$.

Опис тестових файлів подано таблицею, в якій

$L_{\min}$ — шукана найменша загальна довжина кабелю;

P — *емпірична* ймовірність (у %) отримати *правильний* розв'язок при випадковому виборі початкової перестановки (у чисельнику — кількість перестановок, що перетворено в кінці на глобальний мінімум, у знаменнику — 10^4, тобто кількість усіх розглянутих випадкових початкових перестановок).

№	*N*	*M*	$L_{\min}$	*P* (у %)
1	5	5	7	79,47
2	6	8	13	41,89
3	7	8	12	27,84
4	8	14	27	18,42
5	9	12	20	18,77
6	10	12	20	10,89
7	11	22	55	6,89
8	12	30	86	7,65
9	13	40	140	6,35
10	14	60	241	8,97
11	15	50	182	9,66
12	15	30	93	4,22
13	16	50	189	6,20
14	16	60	233	3,75
15	18	60	257	1,63
16	18	100	508	3,23
17	19	80	386	2,56
18	19	90	450	3,85
19	19	100	518	2,99
20	19	120	676	4,35
21	20	120	661	2,15
22	20	180	1186	24,56
23	20	140	824	4,54
24	20	100	524	1,75
25	20	50	194	1,12