Piotr Stańczyk
Treść zadania, Opracowanie

Paweł Wolff
Program

Przeprawa

Drużyna Bajtołazów wybrała się na wycieczkę w Bajtogóry. Niestety Bajtołazi spowodowali lawinę i muszą przed nią uciekać. Na ich drodze znajduje się przepaść i stary most linowy. Muszą jak najszybciej przeprawić się przez most na drugą stronę. Bajtołazi są bardzo zgrani i postanowili, że albo wszyscy się uratują, albo nikt.

Most jest stary i zniszczony, więc nie wytrzyma zbyt dużego obciążenia. Łączna waga Bajtołazów znajdujących się równocześnie na moście nie może być większa od wytrzymałości mostu. Ponadto jest to most linowy. Bajtołazi muszą więc przechodzić przez niego grupami. Kolejna grupa może wejść na most dopiero wtedy, gdy poprzednia go opuści.

Dla każdego Bajtołaza wiadomo, ile czasu zajmie mu przeprawa przez most. Czas przeprawy przez most grupy Bajtołazów jest równy czasowi potrzebnemu na przeprawę przez most najwolniejszego członka grupy. Łączny czas przeprawy wszystkich Bajtołazów to suma czasów przeprawy wszystkich grup. Oczywiście zależy on od tego, jak się podzielą na grupy, przechodząc przez most.

Pomóż Bajtołazom uratować się! Oblicz, jaki jest minimalny czas przeprawy przez most wszystkich Bajtołazów.

Zadanie

Napisz program, który:

- *wczyta z wejścia opis mostu oraz Bajtołazów,*
- *wyznaczy najkrótszy czas przeprawy wszystkich Bajtołazów przez most,*
- *wypisze wyznaczony czas na wyjście.*

Wejście

Pierwsza linia standardowego wejścia zawiera dwie liczby całkowite oddzielone pojedynczym odstępem: w — określającą maksymalne dopuszczalne obciążenie mostu ($100 \leqslant w \leqslant 400$) oraz n — równą liczbie Bajtołazów ($1 \leqslant n \leqslant 16$). W kolejnych n wierszach znajdują się po dwie liczby całkowite oddzielone pojedynczym odstępem, opisujące kolejnych Bajtołazów: t — czas potrzebny na pokonanie mostu przez Bajtołaza ($1 \leqslant t \leqslant 50$) oraz w — waga Bajtołaza ($10 \leqslant w \leqslant 100$).

Wyjście

W pierwszym i jedynym wierszu standardowego wyjścia Twój program powinien wypisać jedną liczbę naturalną oznaczającą minimalny czas przeprawy wszystkich Bajtołazów przez most.

Przykład

Dla danych wejściowych:	*poprawnym wynikiem jest:*

`100 3`
`24 60`
`10 40`
`18 50`
	`42`

Rozwiązanie

Definicja 1 *Podział zbioru* — rodzina niepustych, parami rozłącznych zbiorów, których suma teoriomnogościowa daje dzielony zbiór.

Definicja 2 *Rozbicie zbioru* — podział zbioru składający się z dwóch podzbiorów.

Definicja 3 *Podzbiór maksymalny* — maksymalny podzbiór (w sensie zawierania) zbioru Bajtołazów nieprzeciążający mostu.

W zadaniu Przeprawa mamy do czynienia z problemem optymalizacyjnym. Prawdopodobnie nie istnieje algorytm rozwiązujący podane zadanie w czasie wielomianowym. W związku z tym należy skoncentrować się na konstruowaniu algorytmu przeszukującego zbiór możliwych rozwiązań w poszukiwaniu rozwiązania optymalnego. Z uwagi na fakt, iż możliwych podziałów zbioru n Bajtołazów jest B_n (B_n to n-ta liczb Bella — więcej informacji można znaleźć np. w „Kombinatoryce dla programistów" Witolda Lipskiego [23]), to rozpatrywanie wszystkich możliwych przypadków nie jest najlepszym rozwiązaniem. Pierwszych 15 liczb Bella zawarto w tabeli 1.

n	B_n
0	1
1	1
2	2
3	5
4	15
5	52
6	203
7	877
8	4140
9	21147
10	115975
11	678570
12	4213597
13	27644437
14	190899322
15	1382958545

Tabela 1: Kilka pierwszych liczb Bella

Rozwiązanie wzorcowe

Rozwiązanie wzorcowe nie generuje wszystkich możliwych podziałów zbioru Bajtołazów, lecz realizuje następującą strategię: dla danego zbioru Bajtołazów, program sprawdza, czy mogą oni wszyscy zostać przeprawieni przez most za jednym razem. Jeśli tak, to czas przeprawienia takiej grupy jest równy czasowi przeprawy najwolniejszego członka grupy. W przeciwnym przypadku dokonujemy serii rozbić na dwa rozłączne zbiory, a następnie rozwiązujemy podproblemy niezależnie. Po obliczeniu wyników częściowych musimy je scalić (czyli dodać do siebie czasy przeprawy obu podgrup), a następnie wybrać najlepszy czas spośród wszystkich rozpatrywanych rozbić.

W takim algorytmie wiele zbiorów Bajtołazów jest analizowanych wielokrotnie, więc dobrym pomysłem jest zastosowanie spamiętywania obliczonych już wyników. W ten sposób mamy do przeanalizowania 2^n różnych podzbiorów, co przy optymalnej implementacji generowania rozbić zbioru przekłada się na algorytm działający w czasie $O(3^n)$.

Rozwiązanie wzorcowe jest dodatkowo przyspieszone poprzez zastosowanie pewnych obserwacji, które pozwalają na zmniejszenie ilości rozpatrywanych podproblemów:

- Generowanie rozbicia zbioru Bajtołazów na dwa rozłączne podzbiory można dokonywać w taki sposób, że pierwszy generowany podzbiór składa się z maksymalnej grupy Bajtołazów nie przeciążającej mostu (grupa taka od razu może zostać przepuszczona przez most). Druga grupa składa się z reszty Bajtołazów, dla której wykonujemy kolejne rozbicia.

- Liczba generowanych rozbić może zostać dodatkowo zmniejszona poprzez dołączanie najwolniejszego Bajtołaza na stałe do grupy maksymalnej. Poprawność takiej operacji wynika z faktu, że dla dowolnego zbioru Bajtołazów, zawsze istnieje podział prowadzący do rozwiązania optymalnego, w którym najwolniejszy Bajtołaz znajduje się w grupie maksymalnej. Chciałbym w tym miejscu zauważyć, że dołączanie dowolnego Bajtołaza do grupy maksymalnej nie prowadzi do algorytmu poprawnego.

Lemat 1 (O najwolniejszym Bajtołazie) *Dla każdego zbioru Bajtołazów, istnieje podział prowadzący do rozwiązania optymalnego, w którym najwolniejszy Bajtołaz znajduje się w grupie maksymalnej.*

Dowód Załóżmy odwrotnie, że istnieje taki zbiór Bajtołazów, dla którego nie istnieje opisany w lemacie podział. Wybierzmy dla niego dowolny podział, prowadzący do rozwiązania optymalnego i oznaczmy przez *x* podzbiór zawierający najwolniejszego Bajtołaza. Z założenia wiemy, że *x* nie jest maksymalny, więc istnieje pewna grupa Bajtołazów, których przeniesienie do *x* robi z niego podzbiór maksymalny. Najwolniejszy Bajtołaz znajduje się w *x*, zatem czas przeprawy rozszerzonego zbioru *x* nie pogorszy się, więc nowo uzyskane rozwiązanie jest optymalne. Doszliśmy do sprzeczności z założeniem, zatem dowodzony lemat jest prawdziwy. ■

Pesymistyczna złożoność rozwiązania wzorcowego wynosi $O(3^n)$. Zastosowane optymalizacje nie zmieniają złożoności asymptotycznej, jednak w znacznym stopniu zmniejszają pracę wykonywaną przez program. Zaimplementowane rozwiązanie alternatywne o tej samej złożoności co rozwiązanie wzorcowe, lecz nie uwzględniające powyższych obserwacji, na dużych testach działało ponad 120 razy wolniej.

Rozwiązania nieoptymalne

Jednym z nieoptymalnych rozwiązań jest algorytm, który nie uwzględnia usprawnienia polegającego na rozpatrywaniu podziałów grupy Bajtołazów na grupy maksymalne. Podczas pomiarów na dużych testach takie rozwiązanie było około 10 razy wolniejsze od wzorcowego.

Kolejnym rozwiązaniem nieoptymalnym jest program, który nie dokonuje spamiętywania wyników podproblemów już obliczonych. Podczas testów jego osiągi czasowe były 5–10 razy gorsze od rozwiązania wzorcowego.

Kolejne rozwiązanie nie uwzględnia żadnych optymalizacji — nie tylko nie rozpatruje tylko grup maksymalnych, ale również nie spamiętuje wyników dla podproblemów. Taki algorytm okazuje się niezmiernie wolny w porównaniu z poprzednimi wersjami.

Inne możliwe nieoptymalne rozwiązania to stosowanie jedynie metody spamiętywania lub generowanie wszystkich możliwych podziałów zbioru Bajtołazów, które nie przeciążają mostu.

Rozwiązania niepoprawne

Podczas rozwiązywania tego zadania mogą nasuwać się pewne naturalne, lecz niepoprawne pomysły jego rozwiązania. Programy tego typu można podzielić na dwa rodzaje: algorytmy zachłanne oraz różnego rodzaju heurystyki. Oto przykłady:

- Algorytm zachłanny, który wydziela ze zbioru Bajtołazów kolejno przeprawiane grupy w następujący sposób — dla każdego nieprzeprawionego Bajtołaza (w kolejności od najwolniejszego do najszybszego) wykonuj: jeśli nie przeciąży on tworzonej grupy, to dołącz go do grupy.
- Jeszcze bardziej „naiwna” strategia zachłanna — dla wszystkich Bajtołazów (w kolejności od najwolniejszego do najszybszego) wykonuj: jeśli nie przeciąży on aktualnie tworzonej grupy, to dołącz go do tej grupy; w przeciwnym przypadku przepraw aktualną grupę i zacznij tworzyć nową, dołączając do niej przetwarzanego Bajtołaza.
- Heurystyka probabilistyczna, która generuje pewną liczbę podziałów i wybiera najlepszy z nich.

Testy

Zadanie było sprawdzane na 15 testach. Podczas ich układania duży nacisk położono na to, aby rozwiązania niepoprawne nie uzyskały zbyt wielu punktów. Testy 1–7 to testy poprawnościowe, a 8–15 wydajnościowe. Poniżej przedstawiono rozmiary poszczególnych testów i odpowiadające im wyniki.

nr testu	n	w
1	4	109
2	4	100
3	5	140
4	5	150
5	6	200
6	7	250
7	8	394
8	12	129
9	13	200
10	12	297
11	12	150
12	12	129
13	15	297
14	16	140
15	16	140