Анализ на задача `signs`

Тагове: стек, офлайн заявки

Задачата описва път с N табели една след друга от два вида - за ново ограничение на скоростта и за отмяна на последното активно ограничение, които работят като *undo* - възстановяват ограничението на скоростта, което е важало преди появата на текущото активно ограничение. Тези табели за отмяна са само за целите на задачата, на практика те обикновено отменят ограниченията за скоростта и възстановяват максималното възможно. Интересуваме се от най-краткото време, за което можем да изминем пътя от началната до крайната точка, като е ясно, че за тази цел ще се движим с най-голямата допустима скорост във всеки момент. Интересната част е наличието на заявки, които са независими - само в рамките на заявката се променя някоя табела и трябва да се намери най-краткото време с тези табели.

Решение на първа подзадача - 23 точки

Тази подзадача е с ниски ограничения и е предвидена за запознаване със задачата и осмисляне на табелите за промяна. Примерът покрива голяма част от случаите и би трябвало решение, което тръгне на него, да минава подзадачата. Както по-рано описахме, табелите за отмяна работят на принципа на *undo*. Можем да си мислим, че когато се появи нова табела за максимално ограничение, я слагаме в стек, а като се появи табела за отмяна, махаме най-горния елемент на стека. За улеснение можем в началото да сложим фиктивна табела за ограничение на скоростта от 120, с което да покрием случая при липса на ограничителна табела. В условието е описано, че няма опасност да "отменим" такава табела, така че винаги ще имаме елемент във върха на стека. Така ако направим чиста симулация, поддържайки стека, можем във всеки момент да знаем максималната допустима скорост и да сметнем най-бързото време, за което можем да изминем всяка част от пътя. Само трябва да сумираме тези времена за крайния отговор на всяка заявка.

Сложност: $O(QN)$.

Решение на втора подзадача - 37 точки (=23+14)

Подзадачата има няколко идеи. Едната идея е да могат се изкарат още точки по задачата дори и без особени идеи за пълното решението. Такива подзадачи често има и в задачите на международните състезания. Другата идея е да се насочи мисленето към това, заявките да се обработват офлайн. Записваме заявките, така че за всяка позиция да имаме заявките, които променят табела на тази позиция. Така можем да направим симулация на преминаването през табелите чрез стека и когато видим дадена заявка за позиция да пуснем нова симулацията от този момент до края (за да изчислим отговора). След това можем да се върнем обратно до състоянието от старата симулация, връщайки се по табелите от завършването на новата симулация в обратен ред и възстановявайки стека. Като имаме предвид допълнителното ограничение, че заявките променят табели само в края на редицата, то е гарантирано, че този подход ще е достатъчно бърз.

По принцип такова решение може да е идея и за чийт решение. При слаби тестове е възможно да се държи бързо и на по-големи ограничения.

Сложност: $O(N + Q.M)$, където M е максималната разлика на N и позициите в заявките.

Решение на трета подзадача - 10 точки

По-нататък, по аналогия с отварящите и затварящите скоби, ще наричаме табелите за максимално ограничение на скоростта отварящи табели, а тези за отмяна - затварящи табели. Този начин ни дава по-добро онагледяване, чрез което е по-лесно да правим нататъшните разсъждения какво става като сменим табела. Също преди да започнем с табелите имаме фиктивна отваряща скоба, съответстваща на ограничението по подразбиране 120. В задачата не е нужно полученият израз от скоби да е балансиран (например първата скоба, никога не се затваря).

Погледнато като видове табели имаме четири случая на смяна, от които един не е релевантен - да сменяме затваряща табела със затваряща. Ограничението в тази подзадача е, че сме в най-лесния от останалите 3 случая - сменяме отваряща табела с друга отваряща. За да обработваме тези заявки бързо е достатъчно преди заявките да намерим всяка табела на какво разстояние от пътя участва като активното ограничение (това в общия случай включва множество участъци от пътя, както 120 в примера). Това можем да направим по време на едно преминаване през табелите и поддържане на отворените табели в стек. Преди обработването на нова табела в стека, добавяме текущото разстояние (между новата табелата и предната в редицата) към разстоянието, в което участва отварящата табела на върха на стека. Разбира се, тези разстояния пазим в масив по позициите. Сега като се появи заявка за променя на скоростта на отваряща табела, просто трябва да изчислим разликата между времето при старата скорост и времето при новата скорост.

Сложност: $O(N + Q)$.

Пълно решение - 100 точки

Няма да обясняваме отделно четвърта и пета подзадача, които задават останалите два случая. Те така или иначе са част от пълното решение (както и случая в трета подзадача). Целта на тези подзадачи е по-скоро да е по-лесно писането на пълното решение, така че да може да се проверява по отделно за всеки случай дали вярно се решава.

Нека в момента мислим за случая, в който една затваряща табела се сменя на отваряща. Първо да анализираме тази отваряща табела на какво разстояние ще бъде използвана. Нека моментното състояние на стека преди появата на оригиналната затваряща табела е $s_0, s_1, \ldots, s_k$ ($k > 0$, s_k е на върха на стека). Тогава оригинално се появява затварящата табела и от стека се премахва s_k. От там нататък, когато табела s_{k-1} е на върха на стека, разстоянията се трупат към тази табела. Сега при промяната тези разстояния ще се трупат към новата отваряща табела, т.е. оставащите разстояния на табела s_{k-1} ще се изминават със скоростта на новата отваряща табела. Аналогично можем да забележим, че останалите разстояния на табела s_{k-2} вече ще се изминават със скоростта на табела s_k. Оригинално затварящата табела премахва s_k от стека и за да стане табела s_{k-2} на върха на стека трябва да се премахне s_{k-1}. След промяната в ролята на s_{k-2} ще влезе s_k, защото вече се появява нова отваряща табела в стека и моментите, в които s_k ще е на върха на стека ще са същите, в които s_{k-2} е била на върха оригинално. Така може да се види следната зависимост - останалите разстояния на табела s_i ще бъдат изминати със скоростта на табела s_{i+2}, защото при промяната s_{i+2} ще влезе в ролята на s_i.

Вече имаме добро характеризиране на промяната в този случай. Така ако отново обработваме заявките офлайн по същия начин от втора подзадача, можем да поддържаме няколко допълнителни стойности, за да се справим със смятането. Едната стойност е времето, с което ще допринесат останалите отворени табели в стека до края (в програмата това е `total[1]`). За това е удобно в стека освен да пазим позиция на отворена табела, да пазим и сумата от оставащите разстояния, в които ще участва табелата. Другата стойност (в програмата `total[0]`), която трябва да пазим във връзка с разгледания случай, е времето, с което ще допринесат табелите, ако стане заявка от описания вид, което всъщност са времената, ако за останалите разстояния

на s_i в стека се използва скоростта на s_{i+2} в стека. По този начин вече можем константно да преизчисляваме времето при заявки от описания вид.

Другият случай, при който се смяна отваряща табела на затваряща, е подобен. При него се оказва, че останалите разстояния на s_i започват да се изминават със скорост s_{i-2}. Затова тогава аналогично трябва да поддържаме стойността (в програмата `total[2]`) на времето, с което ще допринесат, ако за останалите разстояния на табели s_i се използва скоростта на s_{i-2}. След това при обработване на заявките от този вид трябва да се внимава разстоянието непосредствено преди сменяната на отварящата табела да се сметне с правилно изминаваната скорост, а не със скоростта на s_{k-2}.

Така пълното решение е първо да прочетем входните данни, да запишем заявките, така че да ги обработим после офлайн. След което намираме разстоянията, в които участва всяка табела като активна. Накрая пускаме една симулация, в която поддържаме три стойности, за да можем да смятаме правилно времената при заявките.

Сложност: $O(N + Q)$.

На пръв поглед тази задача изглежда по-лесна отколкото е. Един объркващ момент е, че при промяна на табелите с противоположен вид всъщност се ползват скоростите на табели през една (с разлика ± 2 в индексите). Поддържането на необходимите сметки за тези случаи се оказва сравнително пипкаво.

Автор: Илиян Йорданов