Keys and Grates

Идея: Михаил Иванов
Разработка: Михаил Иванов

Не умаляя общности, пусть $h > s$. Давайте представим себе оптимальный маршрут Китнисс, как он будет выглядеть? Китнисс будет чередовать хождение налево и хождение направо. Понятно, что если она добралась до какой-то точки в конце хождения налево, то следующее хождение налево должно закончиться строго дальше. В самом деле, зачем иначе Китнисс потребовалось идти налево? Если затем, чтобы подобрать ключ или выйти в люк, то она могла это же сделать и во время предыдущего хождения. А других причин быть и не могло — вскрывать замки на этом отрезке Китнисс не могла, они и так все были вскрыты ранее. Кроме того, понятно, что самое последнее хождение закончится в h, и это будет хождение направо, иначе Китнисс могла покинуть тоннель раньше. Формально, Китнисс выберет некоторый отрезок тоннеля $[L; R]$, содержащий и h, и s, выберет в нём несколько точек L_m, R_m, L_{m-1}, R_{m-1}, L_{m-2}, R_{m-2}, $\ldots$, L_0, R_0 и в этом порядке посетит их, по пути открывая все замки и подбирая все ключи от замков из отрезка $[L; R]$. При этом $L = L_0 < L_1 < \ldots < L_m \le s \le R_m < R_{m-1} < R_{m-2} < \ldots < R_0 = R = h$.

Теперь остаётся лишь выбрать эти отрезки и эти точки. Давайте постепенно это делать с конца. R_0 выбрать легко: это просто h. Как теперь выбрать L_0? Для того, чтобы добраться до R_0, Китнисс надо иметь на руках все ключи от замков из отрезка $[s; R_0]$. Рассмотрим все из этих ключей, которые находятся левее s, выберем самый левый такой ключ — его-то положение и будет точкой L_0. Далее аналогично: чтобы добраться до L_0, надо, чтобы все замки в $[L_0; s]$, ключи от которых правее s, можно было открыть; значит, в качестве R_1 надо взять самый правый ключ, открывающий замок из $[L_0; s]$. Продолжаем так делать, пока не окажется, что на очередном отрезке вообще нет замков, требующих ключей с другой половины тоннеля; тогда мы просто объявляем, что этот очередной отрезок и будет первым отрезком, который мы будем проходить.

Может оказаться, что такого момента никогда не настанет. Это окажется так, если мы обнаружим для какого-то j, что $L_j \ge L_{j+1}$ или $R_j \le R_{j+1}$ — это точно значит, что есть циклическая зависимость. А именно, давайте про пару из замка и люка/ключа (возможно, открывающего другой замок) будем говорить, что замок *запирает* этот люк/ключ, если замок находится между данным люком/ключом и s. Тогда понятно, что Китнисс не сможет подобрать ключ до того, как откроет все замки, запирающие этот ключ, и не сможет выйти в люк, пока не откроет все замки, запирающие люк. Неравенство вроде $R_j \le R_{j+1}$ гарантированно означает, что есть пара i, j, что i-й замок запирает j-й ключ, а j-й замок запирает i-й ключ, и при этом один из замков запирает люк. Тогда Китнисс не сможет открыть ни один из замков, не вскрыв сначала другой — это мы и назвали ранее циклической зависимостью. А, поскольку без вскрытия одного из этих замков Китнисс не добраться до люка, то задача в этом случае неразрешима. Также заметим, что если хоть один нужный нам замок запирает свой собственный ключ, то задачу тоже не решить.

Такое решение работает за $\mathcal{O}(n \log n)$: сортируем все замки, потом для каждого ключа находим ближайший к нему замок, который запирает этот ключ (или флаг, что ключ не заперт никаким замком). Дальше для замков справа от s построим массив префиксных минимумов координат нужных для них ключей, а для замков слева от s — массив суффиксных максимумов координат нужных для них ключей. Теперь, как описывалось ранее, построим L_0 и R_0, переберём все замки на $[L_0; R_0]$ и проверим, что никакой из них не запирает свой собственный ключ. Наконец, построим $R_1, L_1, \ldots$, проверяя каждый раз, что не нарушается строгая монотонность L_i и R_i. Чтобы, скажем, зная R_i, найти L_i, найдём g_j — ближайший к R_i замок, запирающий R_i (в точке R_i всегда будет либо люк, либо ключ). Если g_j не существует, то процесс окончен, Китнисс может беспрепятственно идти от s к R_i. Если g_j существует и правее s, то мы находим в массиве префиксных минимумов самый левый ключ k_q, требующийся для вскрытия всех замков в $[s; g_j]$. Если $k_q \ge s$, то опять же Китнисс может просто идти к R_i, и все нужные для этого ключи будут лежать у неё на пути — другими словами, процесс построения последовательностей L_i, R_i окончен. В противном случае мы говорим, что $L_i = k_q$, и продолжаем процесс.

Наконец, чтобы найти ответ, вычислим $(s - L_m) + \sum_{i=0}^{m} (R_i - L_i) + \sum_{i=0}^{m-1} (R_i - L_{i+1})$ — длину пути

Китнисс.