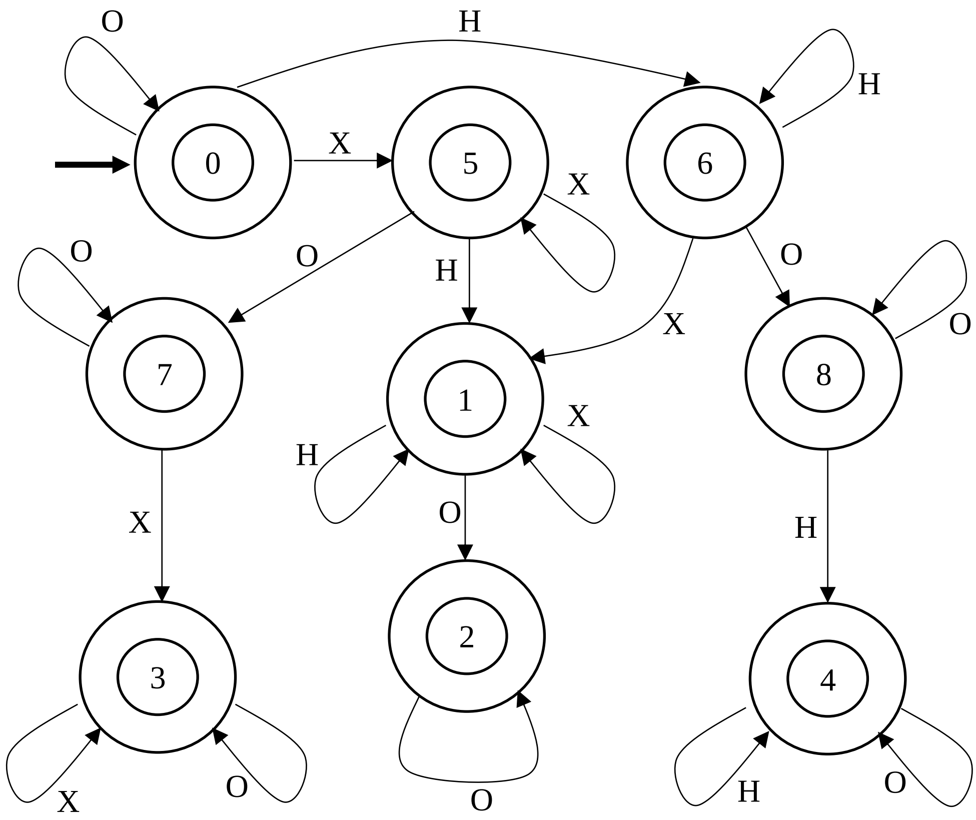
Анализ на задача XOH

Наивните изчерпващи решения включват:

- проверка на всички 3^N на брой N-буквени думи (`XOH_naive.cpp`);
- генериране на думите, съобразено с правилата (`XOH_constr.cpp`).

„Хубавото изречение“ може да бъде описано със следния ориентиран граф-синтактична диаграма (КДА):

От	**Към**	**Буква**
0	0	O
0	5	X
0	6	H
1	1	X
1	1	H
1	2	O
2	2	O
3	3	X
3	3	O
4	4	O
4	4	H
5	1	H
5	5	X
5	7	O
6	1	X
6	6	H
6	8	O
7	3	X
7	7	O
8	4	H
8	8	O



Създаването на подобна диаграма води до динамично решение, линейно относно N: на всяка стъпка броят на изреченията в състояния от 1 до 8 се увеличава с броя изречения на предишната стъпка в съседните им състояния (в състояние 0 винаги има само едно изречение). Така след N стъпки търсеният отговор е сумата от числата във всички състояния (`XOH_DFA.cpp`).

Таблица 1

N	**R**	D_1	D_2	D_3
1	3	6	10	12
2	9	16	22	24
3	25	38	46	48
4	63	84	94	96
5	147	178	190	192
6	325	368	382	384
7	693	750	766	768
8	1443	1516	1534	
9	2959	3050		
10	6009			

До подобна (и много по-проста) рекурсивна зависимост може да се стигне чрез елементарно изследване на получаваната числова редица. Резултатите за малки N могат да се получат с най-наивни алгоритми. Ето до какво води „методът на разликите“ за първите 10 стойности (*Таблица 1*). Стойностите в колонки D_i се получават като разлика на стойността в лявата колонка, долния ред и стойността в лявата колонка, същия ред. Продължаването на метода води до повтаряне на колонка D_3. Само че продължаване не е и нужно: стойностите в D_3 очевидно се задават с формулата $D_{3,N} = 3 \cdot 2^{N+1}$. Също толкова е ясно, че всяко от числата в D_2 е с две по-малко от съответното му в D_3, следователно $D_{2,N} = 3 \cdot 2^{N+1} - 2$. Тогава (*Таблица 2*), тръгвайки от колонка D_2, лесно изчисляваме по обратния път търсената колонка **R** (`XOH_dyn.cpp`). Можем, впрочем, да добавим и „празното изречение“ към „хубавите“, то формално отговаря на дефиницията.

Таблица 2

N	D_2	D_1	**R**
0	4	2	1
1	10	6	3
2	22	16	9
3	46	38	25
4	94	84	63
5	190	178	147
6	382	368	325
7	766	750	693
8	1534	1516	1443
9		3050	2959
10			6009

За зададените ограничения, обаче, линейно решение не е достатъчно. Но ако разпишем полученото за D_1 и оттам – за **R**, ще видим явните формули. $D_{1,N} = 6 \cdot (2^N - 1) - 2(N - 1) = 3 \cdot 2^{N+1} - 2N - 4$, откъдето $R = 1 + 6(1 + 2 + 2^2 + \cdots + 2^{N-1}) - 2(1 + 2 + 3 + \cdots + N - 1) - 4N = 1 + 6(2^N - 1) - 2\frac{N(N-1)}{2} - 4N$ и окончателно $R = 3 \cdot 2^{N+1} - (N^2 + 3N + 5)$. Това вече дава възможност за логаритмична изчислителна сложност чрез „бързо повдигане на степен“ при намирането на 2^N по модул M.

Автор: Павлин Пеев