АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА
ИГРА С ПЯСЪК

Предполагам повечето (почти всички) от състезателите ще имат затруднения с получаването на 100 точки дори и при пълен фийдбек, и то не защото задачата е сложна, ами защото не са си научили стандартните алгоритми.

Нека разгледаме първо подходите, които можем да приложим ако *не* знаем какво да правим. Най-лесното е да направим динамично, в което пазим колко пясък има във всяка от оставащите чашки. Тъй като чашките са много, стейтът ще представим с vector, като за динамична таблица ще ползваме STL-ския map. Съвсем стандартно за игра с двама човека, всеки стейт е или печеливш, или губещ. Стейт, в който не може да бъде направен ход е очевидно губещ. За всеки друг стейт, ако можем да закараме противника в губещ стейт, то текущият е печеливш. Обратно, ако с никой ход не можем да вкараме противника в губещ стейт, то значи текущият е губещ.

Учудващо, това решение се справя доста добре – всъщност би хванало цели 40 точки (тъй като планирам да има пълен фийдбек, реших да не слагам ограничения в условието и да оставя по-находчивите състезатели, които биха пробвали нещо такова, да бъдат приятно изненадани). Малко по-нататък ще видим и защо се справя толкова добре.

Дори малко по-добре би било да приложим хеширане – вместо да пълним мап с ключ вектор, можем да хешираме този вектор и да ползваме хештаблица. При добра имплементация (константен ъпдейт на хешовете при преливане на чашка и троен хеш за избягване на колизии) това решение може да хване 60, 70, а даже 80 точки. Най-простата имплементация (вградената в езика хештаблица, или пък мап в който ключовете са хешове, вместо вектори), хваща около 50 точки.

Добре, а сега да видим и истинското решение. Тази задача е пример за impartial game – игра, в която двамата играчи имат еднакви възможни ходове и еднакви условия за печалба. Съществува една SG теорема, която казва, че такива игри са еквивалентни на играта Nim. Също така дават начин, по който можем да определим дали една позиция е печеливша или губеща.

Скоро ще пусна тема в informatika.bg, в която тази теория ще бъде разгледана по-подробно, но за сега ще споделя единия параграф, с които Румен ми обясни как се решават такива задачи (и от тогава до сега съм срещнал около 7-8 такива по най-различни състезания):

На всеки стейт ще дадем едно число SG. Позициите, от които не можем да направим ход (тоест губим в текущия ход), имат SG-число 0. SG-то на всяка друга позиция се получава по следния начин. Разглеждаме всички възможни ходове от текущата позиция. Избирайки всеки от тях разцепваме задачата на няколко (в конкретната задача най-много две) подзадачи. За всеки възможен ход, вземаме XOR-а на SG числата на подзадачите, които се получават, и го пъхаме в един сет. SG-числото за текущата позиция е първото цяло, неотрицателно число, което *не се* среща в сета.

Защо това е така е съвсем нетривиален въпрос, който, обаче, не ни интересува за да можем да решаваме такива задачи.

Ако вече сте изчислили SG-то на даден стейт, няма нужда да го изчислявате отново (тъй като то няма как да се промени). (Напомня ли ви на нещо това? :))

Сега да разгледаме и самия стейт. Ако си изберем некрайна чашка и я изсипем в някой от съседите ѝ, разделяме задачата на две подзадачи. По-важното е, че подзадачите са независими, тъй като никога не можем да изсипем чашка от единия под-интервал в другия. Съответно ни интересува само кой е под-интервалът (най-ляв и най-десен индекс) и потенциално колко пясък има в най-лявата и най-дясната чашка (в тях може да сме изсипали на предходен ход и да не са с началното си количество).

Тук състезателите трябва да са забелязали, че този стейт е доста голям: 100 * 100 * 250 * 250. Това е единственото усложнение в задачата за да не е учебниково SG – трябва да си оптимизираме стейта. По-наблюдателните сигурно са забелязали, че минималното количество пясък, което имаме в началото във всяка от чашките, не е стандартното 0 или 1, което бихме имали в повечето други задачи. Вместо това то е 25. Защо? Ами представете си една песъчинка от някоя чаша. Можем да я изсипем в някоя съседна. После в друга. После в трета, и така нататък – но най-много 9 пъти. След 9-тия път последната чаша със сигурност не може да бъде прелята в друга, тъй като има 250 милилитра (при всяко пресипване тези 25 милилитра се акумулират с други поне 25).

Затова оптимизацията на стейта е следната – ще пазим отново ляв и десен индекс, но вместо количеството пясък в най-лявата и най-дясната чашка, ще пазим колко от по-левите сме прелели в най-лявата, и колко от по-десните сме прелели в най-дясната. Тъй като това може да е между 0 и 9 пъти, то ще имаме таблица с размер [100][100][10][10], което вече е окей. Всъщност това е *максималният* брой различни валидни стейтове, които може да има – това обяснява и защо решението с мап работеше толкова добре!

Автор: Александър Георгиев