Grabowski Michał

Kamiński Jakub

Michalik Piotr

Wójcik Łukasz

Kombinatoryczna Teoria Liczb

Zadanie 9. Z list III

Dane:

1. liczba naturalna k .
2. liczba dostępnych kolorów c .
3. wielkość list l .
4. maksymalna długość rozgrywki s .

Przebieg gry:

W każdej rundzie:

1. Pierwszy gracz podaje liczbę ze zbioru [1, … , s].
2. Losowana jest lista kolorów wielkości l.
3. Drugi gracz koloruje podaną liczbę jednym z kolorów z listy.

Warunki z zwycięstwa:

1. Pierwszy gracz wygrywa, jeżeli wśród pokolorowanych liczb istnieje monochromatyczny ciąg arytmetyczny długości k (tzn. ciąg postaci i, i + j, i + 2j, . . . , i + (k − 1)j, którego wszystkie elementy są pokolorowane tym samym kolorem).
2. Drugi gracz wygrywa, gdy pokolorowanych jest s liczb, a taki ciąg się nie pojawił.

Ciąg Arytmetyczny

Definicja :

Ciąg liczbowy, w którym każdy wyraz jest sumą wyrazu bezpośrednio go poprzedzającego oraz ustalonej liczby zwanej różnicą ciągu. Zwykle zakładamy, **że** wyrazy ciągu arytmetycznego są liczbami rzeczywistymi.

Ciąg liczbowy

$$(a_n)$$

nazywamy ciągiem arytmetycznym, jeśli dla pewnej liczby r (nazywanej różnicą ciągu) zachodzi

$$(\forall n \in \mathbb{N})\ (a_{n+1} = a_n + r)$$

Równoważnie, wyrażenie jest ciągiem arytmetycznym, jeśli

$$(\forall n \in \mathbb{N})\ (\ a_{n+1} - a_n = a_{n+2} - a_{n+1})$$

Właściwości:

Można policzyć dowolny wyraz ciągu ze wzoru:

$$a_n = a_1 + (n-1)r$$

Strategia Gry

Strategia gry:

Komputer będzie grał w oparciu na algorytm Min-max. Jest to algorytm wywodzący się z teorii gier o sumie zerowej. Często używany przy grach planszowych.

Jego nazwa pochodzi od idei minimalizowania możliwych strat, albo zamiennie maksymalizacji zysku. Jednocześnie do algorytmu zostanie zastosowane obcinanie alfa-beta - ma ono na celu ucinanie gałęzi przeszukiwań, tak aby zmniejszyć ilość wykonywanych operacji oraz wykorzystanie pamięci.

Trudność przeciwnika komputerowego będzie zależna od głębokości przeszukiwania drzewa. Wartość ta będzie parametryzowana, tak, aby równomiernie rozdzielić wzrastanie inteligencji przeciwnika pomiędzy poziomami: łatwy, normalny oraz trudny.

Teoria:

W grze dwuosobowej istnieje pewna wartość V i mieszana strategia dla każdego gracza, takie że

Biorąc pod uwagę strategię gracza drugiego, najlepszą punktacją dla gracza pierwszego jest V

Biorąc pod uwagę gracza pierwszego, najlepszą możliwą dla gracza drugiego jest -V.

Oznacza to, że do każdego możliwego ruchu gracza przypisywana jest wartość punktowa. Komputer będzie podejmował decyzję tak, by zyskać najwięcej możliwych punktów rozważając wszystkie możliwe kombinacje plansze do pewnej ilości rund ‘do przodu’.

Teoria z list

Jeżeli założymy, że nasza lista pokolorowanych liczb tworzy graf doskonały. Natomiast ciąg arytmetyczny długości k pozwala na legalne pokolorowanie wierzchołków. Tzn., z grafu doskonałego eliminujemy krawędzie wzajemnie połączonych ze sobą wierzchołków, które tworzyłyby ten ciąg arytmetyczny.=======. To istnieje taka liczba k, która pozwala na legalne pokolorowanie grafu, niezależnie od wyboru list kolorów przypisanych wierzchołkom.

Jeżeli jako długość ciągu przyjmiemy długość listy. A tabela kolorów będzie się składać z pojedynczego koloru. Innymi słowy nasz graf składać się będzie z wierzchołków bez krawędzi. To wtedy istnieje takie k, równe 1, ze dla dowolnie wylosowanych list kolorów graf będzie spełniać własność ciągu arytmetycznego w rozumieniu gry. A wiec na pewno wygra gracz 1.

Właściwości minimax:

Złożoność czasowa: $O(b^m)$ - suma wszystkich poziomów $(b_0+b_1+b_2+...b_m)$

Złożoność pamięciowa: O(m) - zakładając poruszanie się po 1 ścieżce drzewa na raz.

Zastosowanie obcinania **alfa-beta** pozwala zmniejszyć złożność czasową do

$$O(n^{\frac{m}{2}})$$

Opis algorytmu:

1. Funkcja Score.
    a. Przelicza obecny stan gry do wartości punktowej dla obydwu graczy.
    b. Rozgałęzienie drzewa ruchów B.
2. Zależne od wielkości planszy s.
    a. Stałe dla danej gry.
3. Głębokość D.
    a. Będzie określana przez poziom trudności gry..
    b. Określa ile ruchów do przodu symulować będzie nasz algorytm.

Strategia pierwszego gracza:

- Wybieramy liczbę, taką, by stworzyła najwięcej ciągów w różnych kolorach.
    - Każdy nowy utworzony ciąg o jakimś kolorze **C** - zyskujemy **X** punktów.
- Wybieramy liczbę, tak, aby potencjalne wybranie przez przeciwnika nieprzychylnego koloru zmniejszyło jak najmniejszą liczbę ciągów.
    - Każdy potencjalnie zniszczony ciąg o kolorze **C,** tracimy **Y** punktów.

Strategia drugiego gracza:

Wybieramy kolor, taki, który potencjalnie jak najbardziej zmniejszy liczbę ciągów na planszy.

Przeliczamy ilość wszystkich ciągów na planszy.

Następnie dla wszystkich kolorów:

Podstawiamy kolor.

Wyliczamy ilość ciągów arytmetycznych obecnych na planszy.

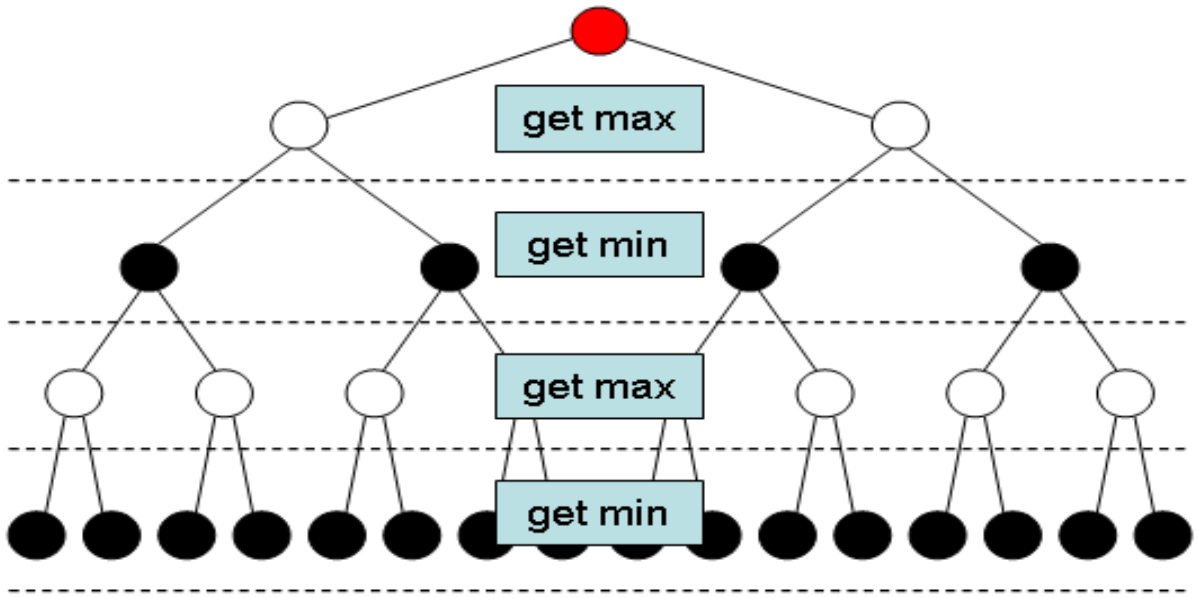
Obliczamy różnicę ilości ciągów.

Powstałą różnicę mnożymy przez pewien współczynnik wyniku Z.

Następnie posługujemy się algorytmem Minimax z obcinaniem alfa-beta, który wybiera strategię w zależności od tego którym graczem steruje.

funkcja minimax(węzeł, głębokość)
  zwróć alfabeta(węzeł, głębokość, -∞, +∞)

```
funkcja alfabeta(węzeł, głębokość, α, β)
   jeżeli węzeł jest końcowy lub głębokość = 0
      zwróć wartość heurystyczną węzła
   jeżeli przeciwnik ma zagrać w węźle
      dla każdego potomka węzła
         β := min(β, alfabeta(potomek, głębokość-1, α, β))
         jeżeli α≥β
            przerwij przeszukiwanie  {odcinamy gałąź Alfa}
      zwróć β
   w przeciwnym przypadku {my mamy zagrać w węźle}
      dla każdego potomka węzła
         α := max(α, alfabeta(potomek, głębokość-1, α, β))
         jeżeli α≥β
            przerwij przeszukiwanie  {odcinamy gałąź Beta}
      zwróć α
```



Funkcja porusza się po drzewie złożonym z możliwych ruchów oraz zwróci wartość heurystyczną dla wybranych liści, które przekazują wartość swoim rodzicom. Ta wartość heurystyczna jest zależna od naszej funkcji ewaluującej, liczącej potencjalną opłacalność poszczególnych ruchów, a więc opłacalność dla gracza maksymalizującego.

Innymi słowy, węzły prowadzące w stronę zwycięstwa będą zwracać większą ilość punktów. Natomiast węzły prowadzące do przegranej odpowiednio mniejszą ilość. Z powodu asymetryczności

gry potrzebny jest algorytm opisujący działanie funkcji ewaluacyjnej opłacalności ruchów gracza 1 i gracza 2.

Jednocześnie zostanie zastosowane obcinanie alfa-beta. Funkcja, kiedy napotyka krawędzie, które widocznie wychodzą poza przedział obliczeń - pomija je. Dzięki temu możemy zaoszczędzić zasoby obliczeniowe, jak i czas obliczeń.

Wykrywanie końca gry

Jeżeli oba poniższe warunki są spełnione, gra toczy się dalej. W przeciwnym wypadku, gra rozpoczyna wyłanianie zwycięzcy.

1. Czy została jakakolwiek liczba do pokolorowania

   Po każdym ruchu gracza, w pętli będą sprawdzane wszystkie liczby ze zbioru s. Jeżeli każda liczba posiada już kolor, gra kończy się.

2. Czy istnieje monochromatyczny ciąg arytmetyczny spełniający warunki.

Algorytm:

Po każdym ruchu gracza, czyli pokolorowaniu danej liczby sprawdzamy czy został utworzony wymagany ciąg monochromatyczny. Najpierw sprawdzamy czy ilość liczb pokolorowanych w danym kolorze jest większa bądź równa wymaganej długości ciągu arytmetycznego. Jeżeli nie jest to przechodzimy do następnej tury, w przeciwnym wypadku kontynuujemy sprawdzanie ciągu.

W następnym kroku znajdujemy maksymalną wartość różnicy ciągu dla danego koloru. Liczymy ją za pomocą wzoru:

$$j_{max} = \frac{(z - i)}{(k - 1)}$$

Gdzie:

i - indeks pierwszej liczby w danym kolorze
z – indeks ostatniej liczby w danym kolorze
k – pożądana długość ciągu arytmetycznego

Potem dla wartości od 1 do j_{max} sprawdzamy czy kolejne liczby w danym kolorze tworzą taki ciąg.

Jeżeli w którymś momencie ciąg został przerwany to sprawdzamy czy dla kolejnych liczb w danym kolorze może zostać utworzony wymagany ciąg tj. czy ilość pozostałych liczb jest większa lub równa od wymaganej długości ciągu i czy odległość pomiędzy pierwszą i ostatnią z pozostałych liczb daje możliwość utworzenia ciągu o różnicy wartości j.

Sprawdzanie kontynuujemy do momentu aż ciąg zostanie znaleziony lub skończymy sprawdzanie szukanie ciągów dla j_{max}.

Wyłanianie zwycięzcy

Gra kończy się jeżeli po którymś z ruchów został znaleziony ciąg monohromatyczny lub gdy wszystkie liczby zostały pokolorowane. Jeżeli po ostatnim ruchu nie został znaleziony ciąg to wygrywa osoba kóra wybierała liczby.