Generacja podziałów zbioru – algorytm

Sprawozdanie z laboratorium 2 – Piotr Sarna LK1

Cel ćwiczenia

Podczas zajęć poznaliśmy algorytm, generujący podziały (partycje) zbioru „n”-elementowego. Algorytm ma za zadanie podzielić dany zbiór zawierający liczby całkowite od 1 do „n” na mniejsze partycje. Liczba partycji, w których możemy rozdzielić liczby ze zbioru wynosi od 1 do „n”, tj. możemy albo umieścić w jednej partycji wszystkie liczby ze zbioru lub każdą liczbę ze zbioru umieścić w osobnej partycji.

Wstęp teoretyczny

Podział (partycja) to dowolnie ułożony i niepusty podzbiór zbioru „n”-elementowego. Na przykład, jeśli chcemy podzielić zbiór 5-cio elementowy: {1, 2, 3, 4, 5}, możemy go podzielić na 3 partycje: {1, 2}, {3}, {4, 5}.

W partycjach nie ma znaczenia ilość elementów, o ile żaden z nich nie jest pusty.

Do określenia na ile sposobów możemy podzielić zbiór służy liczba Bella, dana wzorem:

$$B_n = \Sigma_{k=0}^{n}\{\frac{n}{k}\}$$

Źródło:
https://inf.ug.edu.pl/~mdziemia/kombinatoryka/liczby_bella_eks.pdf

Gdzie $\{\frac{n}{k}\}$ to liczba Stringa II rodzaju, która definiuje liczbę „k”-blokowych podziałów „n”-elementowego zbioru.

Opis algorytmu

Algorytmem, który mieliśmy zaimplementować jest algorytm, który generuje wszystkie możliwe podziały zbioru „n”-elementowego. Jako parametr przyjmował on liczbę „n”, określającą wielkość zbioru.

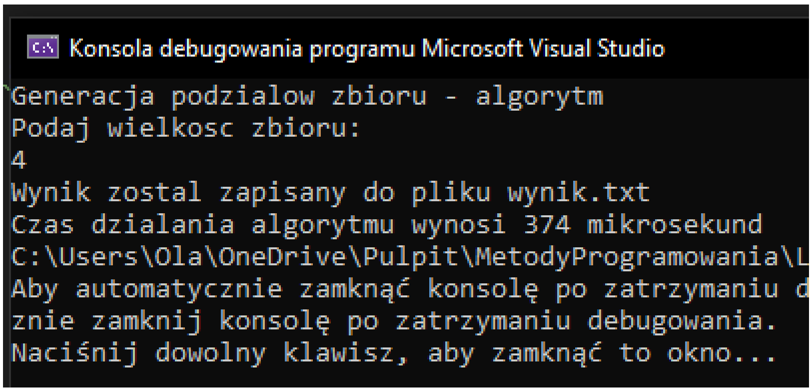
Zapis algorytmu w pseudokodzie:

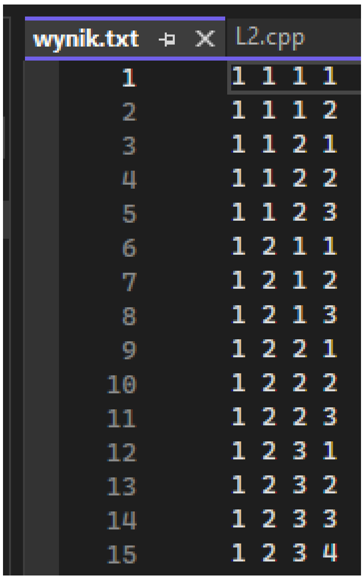
inicjalizacja:
V0 For [i] From [0] To [n], execute the following steps:
  V0.1 Set [$a_i \leftarrow 1$].
  V0.2 Set [$b_i \leftarrow 1$].
powtarzaj:
  V1 Set [$c \leftarrow n$].
  V2 While [$a_c = n$ Or $a_c > b_c$] Do [$c \leftarrow c - 1$].
  V3 If [$c = 1$] Report the end of enumeration.
  V4 Set [$a_c \leftarrow a_c + 1$].
  V5 For [i] From [$c + 1$] To [n], execute the following steps:
    V5.1 Set [$a_i \leftarrow 1$].
    V5.2 Set [$b_i \leftarrow \max(a_{i-1}, b_{i-1})$].
  V6 wyprowadź a

Źródło: Giorgos Stamatelatos, Pavlos S. Efraimidis, Lexicographic Enumeration of Set Partitions, arXiv - CS - Discrete Mathematics, 2021

Na zdjęciu widzimy zmieniony zapis algorytmu, generujący podzbiory zbioru od 1 do „n”. Moja implementacja również zawiera tą zmianę.

Prezentacja działania mojej implementacji w C++





Co jednak oznaczają te liczby? Oznaczają one, w której partycji ma się znajdować poszczególny element zbioru „n”-elementowego (w tym przypadku za „n” podana jest liczba 4), więc zbiór wygląda następująco: {1, 2, 3, 4}. W linijce nr 13 podany jest podział „1 2 3 2”, oznaczający podział zbioru na partycje:

{1}, {2, 4}, {3}

P1 P2 P3

n	Czas pracy algorytmu
2	257 µs
4	372 µs
6	3412 µs
8	76609 µs
10	2579582 µs

Wnioski

Algorytm jest bardzo nieefektywny dla dużych „n”, ponieważ liczba Bella rośnie ekstremalnie szybko – szybciej, niż jakakolwiek funkcja wielomianowa

Bibliografia

- materiały z wykładów nt. obiektów kombinatorycznych
- W. Lipski, „Kombinatoryka dla programistów”, WNT, Warszawa 1989
- Giorgos Stamatelatos, Pavlos S. Efraimidis, Lexicographic Enumeration of Set Partitions, arXiv - CS - Discrete Mathematics, 2021