

Laboratorium PAiMSI 10

Najdłuższy Wspólny Podciąg

Witold Zimnicki - nr 200465

25 maja 2014

1. Cel ćwiczenia:

Celem ćwiczenia było zapoznanie się z algorytmem **NWP** (Najdłuższy Wspólny Podciąg) oraz jego zaimplementowanie w taki sposób, aby możliwe było zaprezentowanie wyników udowadniających jego poprawne działanie.

2. Najdłuższy Wspólny Podciąg:

- Zadaniem algorytmu **NWP** jest wyszukanie najdłuższego wspólnego łańcucha znaków w dwóch ciągach wyrazów. Unikatowość swoją zawdzięcza temu, że elementy łańcucha nie muszą leżeć obok siebie. Wykorzystywany może być do wykrywania plagiatów lub do różnego rodzaju systemów kontroli wersji.
- Problem NWP dwóch ciągów A i B o długościach odpowiednio n i m może być rozwiązany za pomocą metody programowania dynamicznego. Algorytm ten ma złożoność obliczeniową rzędu $O(n * m)$, gdzie n oraz m to długości ciągów A i B. Algorytm ten tworzy tablicę dwuwymiarową C (n na m) taką, że wartość $C[i][j]$ jest długością NWP podciągów $A[1..i]$ i $B[1..j]$. A więc po zakończeniu wypełniania tablicy C komórka $C[n][m]$ będzie zawierała wartość będącą długością NWP oryginalnych ciągów wejściowych A i B.

Odtworzenie najdłuższego podciagu:

Polega ono na utworzeniu macierzy o wymiarach $n+1 \times m+1$ (n i m to długości ciągów A i B). Pierwsza kolumna oraz wiersz są zerami. Następnie wypełniamy macierz aż do końca jednym schematem:

1. Jeśli znak kolejnego wiersza i kolumny są takie same - inkrementowana jest wartość na ukos z lewej strony.
2. Jeśli znaki kolejnego wiersza i kolumny nie są takie same - w zależności od tego która jest większa, przypisywana jest wartość sąsiada na górze, lub po lewej.

Sprawdzenie zawartości najdłuższego wspólnego podciągu:

Aby dowiedzieć się jaki to podciąg należy 'przejsć' z ostatniej komórki macierzy do komórki początkowej według następującego algorytmu:

1. jeśli któraś komórka sąsiednia do tej w której jesteśmy (z lewej strony lub u góry) ma wartość identyczną, to przechodzimy do niej.
2. jeśli jesteśmy w komórce zerowej to zakończ.
3. jeśli nie ma takiej, do ciągu odpowiedzi dopisujemy na początek literę odpowiadającą tej komórce a następnie idziemy do komórki o wartości mniejszej o 1 (po skosie do góry i w lewo).
4. jeśli nie jesteśmy w komórce [0][0] to przeskocz do punktu 1. w przeciwnym wypadku Zakończ.

W związku z tym, że w każdym kolejnym kroku można iść zarówno w lewo jak i do góry, można uzyskać dwa, lub więcej podciągów.

3. Wyniki działania:

- Wariant z jednym możliwym najdłuższym wspólnym podciągiem:

Działanie zostało sprawdzone na przykładowych ciągach znaków: "*komputer*" oraz "*komputerami*". Ich najdłuższy wspólny podciąg to "*komputer*". Sprawdzimy zatem ich działanie zaimplementowanego w ćwiczeniu algorytmu:

```

C:\Users\Witek\Documents\Visual Studio 2012\Projects\Project51PAMSIlab10\Debug\Project51PAMSIlab10.exe
PODAJ PIERWSZE SŁOWO <w macierzy, znaki przy osi wierszy>:
komputer
PODAJ DRUGIE SŁOWO <w macierzy, znaki przy osi kolumn>:
komputerami

*****
Macierz dla słow: komputer <pionowa os> oraz komputerami <pozioma os> :
0 0 0 0 0 0 0 0 0 0 0
0 1 1 1 1 1 1 1 1 1 1
0 1 2 2 2 2 2 2 2 2 2
0 1 2 3 3 3 3 3 3 3 3
0 1 2 3 4 4 4 4 4 4 4
0 1 2 3 4 5 5 5 5 5 5
0 1 2 3 4 5 6 6 6 6 6
0 1 2 3 4 5 6 7 7 7 7
0 1 2 3 4 5 6 7 8 8 8

DLUGOSC NAJDŁUŻSZEGO WSPÓLNEGO PODCIĄGU:
8
Najdluzszy wspolny podciag dla slow: komputer i komputerami to: komputer
Aby kontynuować, naciśnij dowolny klawisz . . . _
  
```

Rysunek 1: Wariant 1

- Wariant z dwoma możliwymi najdłuższymi wspólnymi podciągami (gdy idziemy od ostatniego elementu w górę, lub w lewo):

Działanie zostało sprawdzone na przykładowych ciągach znaków: *giggiigg* oraz *igigi*. Ich najdłuższy wspólny podciąg to **igig** oraz **gigi**. Sprawdźmy zatem ich działanie zaimplementowanego w ćwiczeniu algorytmu:

```

C:\Users\Witek\Documents\Visual Studio 2012\Projects\Project51PAMSlab10\Debug\Project51PAMSlab10.exe
PODAJ PIERWSZE SŁOWO <w macierzy, znaki przy osi wierszy>:
giggiigg
PODAJ DRUGIE SŁOWO <w macierzy, znaki przy osi kolumn>:
igigi
*****
Macierz dla słow: giggiigg < pionowa os > oraz igigi < pozioma os > :
0 0 0 0 0 0
0 0 1 1 1 1
0 1 1 2 2 2
0 1 2 2 3 3
0 1 2 2 3 3
0 1 2 3 3 4
0 1 2 3 3 4
0 1 2 3 4 4
0 1 2 3 4 4
0 1 2 3 4 4
DLUGOSC NAJDŁUŻSZEGO WSPÓLNEGO PODCIĄGU:
4
Najdłuższy wspólny podciąg dla słow: giggiigg i igigi to: igig
Druga możliwość najdłuższego wspólnego podciagu słow giggiigg i igigi to: gigi
Aby kontynuować, naciśnij dowolny klawisz . . . _

```

Rysunek 2: Wariant 2

- **Wariant z brakiem wspólnego podciągu:**

Działanie zostało sprawdzone na przykładowych ciągach znaków: **abcdef** oraz **ghijkl**. Sprawdźmy działanie zaimplementowanego w ćwiczeniu algorytmu:

```
C:\Users\Witek\Documents\Visual Studio 2012\Projects\Project51PAMSIlab10\Debug\Project51PAMSIlab10.exe
PODAJ PIERWSZE SLOWO <w macierzy, znaki przy osi wierszy>:
abcdef
PODAJ DRUGIE SLOWO <w macierzy, znaki przy osi kolumn>:
ghijkl

*****
Macierz dla slow: abcdef <pionowa os> oraz ghijkl <pozioma os> :
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
DLUGOSC NAJDLUZSZEGO WSPOLNEGO PODCIAGU:
0
BRAK WSPOLNEGO PODCIAGU
Aby kontynuować, naciśnij dowolny klawisz . . .
```

Rysunek 3: Wariant 3

4. Wnioski:

- Wyniki testu potwierdzają poprawność działania zaimplementowanych funkcji **NWP**.
- Ćwiczenie pokazuje, że algorytm NWP może być przydatny w kontrolowaniu wersji dokumentów, książek czy też wykrywaniu plagiatów.
- Przy dużych rozmiarach ciągów uwidacznia się stosunkowo duża złożoność czasowa algorytmu $O(n * m)$, gdzie n i m to długości zadanych ciągów znaków .
- Algorytm wyszukuje wspólny najdłuższy podciąg, nawet jeśli elementy ciągów zadanych nie leżą obok siebie. Nie wolno go zatem mylić z algorytmem *najdłuższego wspólnego podłańcucha*.