Algorytmy i struktury danych

wykład 9

Plan wykładu:

- Funkcje mieszające.
- Algorytmy numeryczne.

Funkcje mieszające

Funkcja skrótu – jest to funkcja *H*, która dla do dowolnej informacji *m* przyporządkowuje niespecyficzną wartość *h*, mającą cechy pseudolosowe.

Cechy:

- skróty są zazwyczaj stałej długości (mają tą samą liczbę znaków),
- skrót jednoznacznie identyfikuje informację, dla której został utworzony.

Funkcja skrótu – jest to funkcja H, która dla do dowolnej informacji m przyporządkowuje niespecyficzną wartość h, mającą cechy pseudolosowe.

Cechy:

- skróty są zazwyczaj stałej długości (mają tą samą liczbę znaków),
- skrót jednoznacznie identyfikuje informację, dla której został utworzony.

Kolizja – dla funkcji skrótu H to taka para różnych wiadomości m_1 i m_2, dla których wartość funkcji skrótu H jest identyczna, tj. $H(m_1) = H(m_2)$.

Funkcja skrótu – jest to funkcja H, która dla do dowolnej informacji m przyporządkowuje niespecyficzną wartość h, mającą cechy pseudolosowe.

Cechy:
- skróty są zazwyczaj stałej długości (mają tą samą liczbę znaków),
- skrót jednoznacznie identyfikuje informację, dla której został utworzony.

Kolizja – dla funkcji skrótu H to taka para różnych wiadomości m_1 i m_2, dla których wartość funkcji skrótu H jest identyczna, tj. $H(m_1) = H(m_2)$.

Własności funkcji skrótu ze względu na kolizje:
- brak możliwości łatwego generowania nowych kolizji,
- odporność na znalezienie drugiego przeciwobrazu,
- odporność na znalezienie przeciwobrazu.

Funkcja skrótu – jest to funkcja H, która dla do dowolnej informacji m przyporządkowuje niespecyficzną wartość h, mającą cechy pseudolosowe.

Cechy:
- skróty są zazwyczaj stałej długości (mają tą samą liczbę znaków),
- skrót jednoznacznie identyfikuje informację, dla której został utworzony.

Kolizja – dla funkcji skrótu H to taka para różnych wiadomości m_1 i m_2, dla których wartość funkcji skrótu H jest identyczna, tj. $H(m_1) = H(m_2)$.

Własności funkcji skrótu ze względu na kolizje:
- brak możliwości łatwego generowania nowych kolizji, → Oznacza, że możliwość znalezienia kolizji dla dwóch różnych wiadomości jest znikomo mała.
- odporność na znalezienie drugiego przeciwobrazu,
- odporność na znalezienie przeciwobrazu.

Funkcja skrótu – jest to funkcja H, która dla do dowolnej informacji m przyporządkowuje niespecyficzną wartość h, mającą cechy pseudolosowe.

Cechy:
- skróty są zazwyczaj stałej długości (mają tą samą liczbę znaków),
- skrót jednoznacznie identyfikuje informację, dla której został utworzony.

Kolizja – dla funkcji skrótu H to taka para różnych wiadomości m_1 i m_2, dla których wartość funkcji skrótu H jest identyczna, tj. $H(m_1) = H(m_2)$.

Własności funkcji skrótu ze względu na kolizje:
- brak możliwości łatwego generowania nowych kolizji,
- odporność na znalezienie drugiego przeciwobrazu,
- odporność na znalezienie przeciwobrazu.

Oznacza, że możliwość znalezienia kolizji dla dwóch różnych wiadomości jest znikomo mała.

Second preimage resistance – dla danej wiadomości nie jest znana **szybka** metoda znalezienia innej różnej wiadomości, takiej, że ich skróty są sobie równe.

Funkcja skrótu – jest to funkcja H, która dla do dowolnej informacji m przyporządkowuje niespecyficzną wartość h, mającą cechy pseudolosowe.

Cechy:
- skróty są zazwyczaj stałej długości (mają tą samą liczbę znaków),
- skrót jednoznacznie identyfikuje informację, dla której został utworzony.

Kolizja – dla funkcji skrótu H to taka para różnych wiadomości m_1 i m_2, dla których wartość funkcji skrótu H jest identyczna, tj. $H(m_1) = H(m_2)$.

Własności funkcji skrótu ze względu na kolizje:
- brak możliwości łatwego generowania nowych kolizji,
- odporność na znalezienie drugiego przeciwobrazu,
- odporność na znalezienie przeciwobrazu.

Oznacza, że możliwość znalezienia kolizji dla dwóch różnych wiadomości jest znikomo mała.

Preimage resistance – dla skrótu h nie istnieje żadna **szybka** metoda znalezienia wiadomości m, takiej, że $H(m) = h$ (własność najsłabsza).

Second preimage resistance – dla danej wiadomości nie jest znana **szybka** metoda znalezienia innej różnej wiadomości, takiej, że ich skróty są sobie równe.

Funkcja skrótu – jest to funkcja H, która dla do dowolnej informacji m przyporządkowuje niespecyficzną wartość h, mającą cechy pseudolosowe.

Cechy:
- skróty są zazwyczaj stałej długości (mają tą samą liczbę znaków),
- skrót jednoznacznie identyfikuje informację, dla której został utworzony.

Kolizja – dla funkcji skrótu H to taka para różnych wiadomości m_1 i m_2, dla których wartość funkcji skrótu H jest identyczna, tj. $H(m_1) = H(m_2)$.

Własności funkcji skrótu ze względu na kolizje:
- brak możliwości łatwego generowania nowych kolizji,
- odporność na znalezienie drugiego przeciwobrazu,
- odporność na znalezienie przeciwobrazu.

Oznacza, że możliwość znalezienia kolizji dla dwóch różnych wiadomości jest znikomo mała.

Preimage resistance – dla skrótu h nie istnieje żadna **szybka** metoda znalezienia wiadomości m, takiej, że $H(m) = h$ (własność najsłabsza).

Second preimage resistance – dla danej wiadomości nie jest znana **szybka** metoda znalezienia innej różnej wiadomości, takiej, że ich skróty są sobie równe.

Brak własności second preimage resistance oznacza brak odporności funkcji H na kolizje.
Brak własności preimage resistance oznacza nieposiadanie przez funkcję własności second preimage resistance.

Funkcja skrótu – jest to funkcja *H*, która dla do dowolnej informacji *m* przyporządkowuje niespecyficzną wartość *h*, mającą cechy pseudolosowe.

Zastosowania:

- sygnaturowanie,
- implementacja sum kontrolnych,
- optymalizacja dostępu do danych,
- w kryptografii.

Funkcja skrótu – jest to funkcja *H*, która dla do dowolnej informacji *m* przyporządkowuje niespecyficzną wartość *h*, mającą cechy pseudolosowe.

Zastosowania:

- sygnaturowanie, → Służą do potwierdzania autentyczności danych.
- implementacja sum kontrolnych,
- optymalizacja dostępu do danych,
- w kryptografii.

Funkcja skrótu – jest to funkcja *H*, która dla do dowolnej informacji *m* przyporządkowuje niespecyficzną wartość *h*, mającą cechy pseudolosowe.

Zastosowania:

- sygnaturowanie, → Służą do potwierdzania autentyczności danych.
- implementacja sum kontrolnych, → Wykorzystywane do wykrywania błędnych danych, przekłamań w transmisjach danych itp.
- optymalizacja dostępu do danych,
- w kryptografii.

Funkcja skrótu – jest to funkcja *H*, która dla do dowolnej informacji *m* przyporządkowuje niespecyficzną wartość *h*, mającą cechy pseudolosowe.

Zastosowania:

- sygnaturowanie,
- implementacja sum kontrolnych,
- optymalizacja dostępu do danych,
- w kryptografii.

→ Służą do potwierdzania autentyczności danych.

→ Wykorzystywane do wykrywania błędnych danych, przekłamań w transmisjach danych itp.

→ Jako składnik mechanizmów dostępu do danych (tablice mieszające i asocjacyjne),

Funkcja skrótu – jest to funkcja *H*, która dla do dowolnej informacji *m* przyporządkowuje niespecyficzną wartość *h*, mającą cechy pseudolosowe.

Zastosowania:

- sygnaturowanie, → Służą do potwierdzania autentyczności danych.
- implementacja sum kontrolnych, → Wykorzystywane do wykrywania błędnych danych, przekłamań w transmisjach danych itp.
- optymalizacja dostępu do danych, → Jako składnik mechanizmów dostępu do danych (tablice mieszające i asocjacyjne),
- w kryptografii. → W kryptografii funkcja mieszająca musi spełniać wymagania:
  - brak praktycznej możliwości wygenerowanie dwóch wiadomości o takim samym skrócie (kolizja),
  - brak możliwości wnioskowania na temat wiadomości wejściowej na podstawie wartości skrótu (jednokierunkowość).

Funkcja skrótu – jest to funkcja *H*, która dla do dowolnej informacji *m* przyporządkowuje niespecyficzną wartość *h*, mającą cechy pseudolosowe.

Zastosowania:

- sygnaturowanie, → Służą do potwierdzania autentyczności danych.
- implementacja sum kontrolnych, → Wykorzystywane do wykrywania błędnych danych, przekłamań w transmisjach danych itp.
- optymalizacja dostępu do danych, → Jako składnik mechanizmów dostępu do danych (tablice mieszające i asocjacyjne),
- w kryptografii.

W kryptografii funkcja mieszająca musi spełniać wymagania:
- brak praktycznej możliwości wygenerowanie dwóch wiadomości o takim samym skrócie (kolizja),
- brak możliwości wnioskowania na temat wiadomości wejściowej na podstawie wartości skrótu (jednokierunkowość).

Zmiana dowolnego pojedynczego bitu wiadomości powinna zmieniać średnio połowę bitów skrótu w sposób, który nie jest istotnie podatny na kryptoanalizę różnicową.

Funkcja skrótu – jest to funkcja *H*, która dla do dowolnej informacji *m* przyporządkowuje niespecyficzną wartość *h*, mającą cechy pseudolosowe.

Zastosowania:

- sygnaturowanie, → Służą do potwierdzania autentyczności danych.
- implementacja sum kontrolnych, → Wykorzystywane do wykrywania błędnych danych, przekłamań w transmisjach danych itp.
- optymalizacja dostępu do danych, → Jako składnik mechanizmów dostępu do danych (tablice mieszające i asocjacyjne),
- w kryptografii.

W kryptografii funkcja mieszająca musi spełniać wymagania:
- brak praktycznej możliwości wygenerowanie dwóch wiadomości o takim samym skrócie (kolizja),
- brak możliwości wnioskowania na temat wiadomości wejściowej na podstawie wartości skrótu (jednokierunkowość).

Zmiana dowolnego pojedynczego bitu wiadomości powinna zmieniać średnio połowę bitów skrótu w sposób, który nie jest istotnie podatny na kryptoanalizę różnicową.

Przykłady:

- MD5,
- SHA-1,
- SHA-2.

Tablica asocjacyjna – ang. associative array – tablica skojarzeniowa, jest to rodzaj abstrakcyjnej struktury danych, przechowującej parę klucz-dane, umożliwiającej dostęp do danych na podstawie klucza.

Cechy:

- tablica asocjacyjna to dane powiązane z mechanizmami dostępu,
- typ danych klucza może być praktycznie dowolny.

Tablica asocjacyjna – ang. associative array – tablica skojarzeniowa, jest to rodzaj abstrakcyjnej struktury danych, przechowującej parę klucz-dane, umożliwiającej dostęp do danych na podstawie klucza.

Cechy:

- tablica asocjacyjna to dane powiązane z mechanizmami dostępu,
- typ danych klucza może być praktycznie dowolny.

W praktyce stosuje się drzewa poszukiwań (BST, AVL, itp.) lub tablice mieszające.

Tablica asocjacyjna – ang. associative array – tablica skojarzeniowa, jest to rodzaj abstrakcyjnej struktury danych, przechowującej parę klucz-dane, umożliwiającej dostęp do danych na podstawie klucza.

Cechy:

- tablica asocjacyjna to dane powiązane z mechanizmami dostępu,
- typ danych klucza może być praktycznie dowolny.

Najczęściej są to łańcuchy znaków, ale także spotykane są liczby (całkowite, zmiennoprzecinkowe, zespolone), krotki, itp.

W praktyce stosuje się drzewa poszukiwań (BST, AVL, itp.) lub tablice mieszające.

Tablica asocjacyjna – ang. associative array – tablica skojarzeniowa, jest to rodzaj abstrakcyjnej struktury danych, przechowującej parę klucz-dane, umożliwiającej dostęp do danych na podstawie klucza.

Cechy:

- tablica asocjacyjna to dane powiązane z mechanizmami dostępu,
- typ danych klucza może być praktycznie dowolny.

Najczęściej są to łańcuchy znaków, ale także spotykane są liczby (całkowite, zmiennoprzecinkowe, zespolone), krotki, itp.

W praktyce stosuje się drzewa poszukiwań (BST, AVL, itp.) lub tablice mieszające.

Typowe operacje związane z działaniem na tablicach asocjacyjnych:

- przypisanie wartości do klucza, lub utworzenie klucza jeśli jest przypisanie do nieistniejącego klucza,
- aktualizacja klucza – następuje zazwyczaj podczas próby dostępu do danych,
- odwołanie do nieistniejącego klucza generuje błąd,
- usunięcie klucza i opcjonalnie danych,
- sprawdzenie występowania klucza,
- pobranie list wszystkich kluczy, wszystkich wartości lub wszystkich par (klucz-dane).

Tablica mieszająca – jest to sposób implementacji tablicy asocjacyjnej, służącej do przechowywania informacji w sposób gwarantujący możliwie największą szybkość dostępu do danych, realizowany na podstawie wskazania danych przez informacje identyfikujące (tzw. klucz).

Implementacja:

- zbudowane z tablic podstawowych,
- wykorzystują funkcje mieszającą,

Tablica mieszająca – jest to sposób implementacji tablicy asocjacyjnej, służącej do przechowywania informacji w sposób gwarantujący możliwie największą szybkość dostępu do danych, realizowany na podstawie wskazania danych przez informacje identyfikujące (tzw. klucz).

Implementacja:

- zbudowane z tablic podstawowych,
- wykorzystują funkcje mieszającą,

Tablice mieszające zbudowane są w oparciu o zwykłe tablicach indeksowanych liczbami, których dostęp do danych jest bardzo szybki, niezależnie od rozmiaru tablicy ani położenia elementu.

Tablica mieszająca – jest to sposób implementacji tablicy asocjacyjnej, służącej do przechowywania informacji w sposób gwarantujący możliwie największą szybkość dostępu do danych, realizowany na podstawie wskazania danych przez informacje identyfikujące (tzw. klucz).

Implementacja:

- zbudowane z tablic podstawowych,
- wykorzystują funkcje mieszającą,

Funkcja mieszająca, dla danego klucza, wyznacza indeks w tablicy (czyli przekształca klucz w liczbę z zadanego zakresu), przy czym funkcje mieszającą dobiera się do klucza.

Tablice mieszające zbudowane są w oparciu o zwykłe tablicach indeksowanych liczbami, których dostęp do danych jest bardzo szybki, niezależnie od rozmiaru tablicy ani położenia elementu.

Tablica mieszająca – jest to sposób implementacji tablicy asocjacyjnej, służącej do przechowywania informacji w sposób gwarantujący możliwie największą szybkość dostępu do danych, realizowany na podstawie wskazania danych przez informacje identyfikujące (tzw. klucz).

Implementacja:
- zbudowane z tablic podstawowych,
- wykorzystują funkcje mieszającą,

Funkcja mieszająca, dla danego klucza, wyznacza indeks w tablicy (czyli przekształca klucz w liczbę z zadanego zakresu), przy czym funkcje mieszającą dobiera się do klucza.

Tablice mieszające zbudowane są w oparciu o zwykłe tablicach indeksowanych liczbami, których dostęp do danych jest bardzo szybki, niezależnie od rozmiaru tablicy ani położenia elementu.

Zasady doboru funkcji mieszającej:
- Są to zwykle funkcje o małej złożoności obliczeniowej, których wartości oblicza się względem klucza. Wartością funkcji jest indeks w tablicy z daną lub wskazanie na puste miejsce, jeśli dana nie istnieje. Złożoność czasowa wynosi O(1), wada jest możliwość wystąpienia kolizji.
- Dla danych o znanych cechach można wyznaczyć doskonałą funkcję mieszającą odwzorowującą n kluczy w różne wartości z przedziału $[0..m-1]$, gdzie $m \geqslant n$ lub minimalną doskonałą funkcję mieszającą, dla której $m = n$.

Kolizja w tablicy mieszającej – występuje w sytuacji, gdy wartość funkcji mieszającej obliczonej dla klucza elementu wstawianego do tablicy pokrywa się z wartością klucza innego elementu znajdującego się już w tablicy,

Rozwiązywanie problemu kolizji:

- metoda łańcuchowa,
- adresowanie otwarte.

Kolizja w tablicy mieszającej – występuje w sytuacji, gdy wartość funkcji mieszającej obliczonej dla klucza elementu wstawianego do tablicy pokrywa się z wartością klucza innego elementu znajdującego się już w tablicy,

Rozwiązywanie problemu kolizji:

- metoda łańcuchowa,
- adresowanie otwarte.

Metoda polega na przechowywaniu elementów nie bezpośrednio w tablicy, lecz na liście związanej z danym indeksem. Wstawiane elementy dołącza się na końcu listy.

Średnia złożoność wyszukiwania jest sumą liniowego wyszukiwania elementu w liście i zależy od współczynnika wypełnienia listy, czyli stosunku liczby elementów do wielkości tablicy. Ponieważ złożoność pesymistyczna wyszukiwania wynosi O(n), czasami zamiast list stosuje się drzewa. Zaletą metody łańcuchowej jest szybkość i prostota usuwania elementów z listy.

Kolizja w tablicy mieszającej – występuje w sytuacji, gdy wartość funkcji mieszającej obliczonej dla klucza elementu wstawianego do tablicy pokrywa się z wartością klucza innego elementu znajdującego się już w tablicy,

Rozwiązywanie problemu kolizji:

- metoda łańcuchowa,
- adresowanie otwarte.

Metoda polega na wstawianiu elementu do tablicy w innym miejscu niż wynikałoby to z wartości funkcji mieszającej. Nowa lokalizacja określana jest przez dodanie do wartości funkcji mieszającej wartości tzw. funkcji przyrostu $p(i)$, gdzie i oznacza numer próby wstawienia.

Metody adresowania otwartego:

- wyszukiwanie liniowe – funkcja przyrostu postaci $p(i) = i$,
- wyszukiwanie kwadratowe – $p(i) = i^2$,
- mieszanie podwójne – $p(i) = i \cdot H'(K)$, gdzie H' jest dodatkową funkcją mieszającą dla klucza K.

Metoda polega na przechowywaniu elementów nie bezpośrednio w tablicy, lecz na liście związanej z danym indeksem. Wstawiane elementy dołącza się na końcu listy.

Średnia złożoność wyszukiwania jest sumą liniowego wyszukiwania elementu w liście i zależy od współczynnika wypełnienia listy, czyli stosunku liczby elementów do wielkości tablicy. Ponieważ złożoność pesymistyczna wyszukiwania wynosi $O(n)$, czasami zamiast list stosuje się drzewa. Zaletą metody łańcuchowej jest szybkość i prostota usuwania elementów z listy.

Tablica mieszająca – jest to sposób implementacji tablicy asocjacyjnej, służącej do przechowywania informacji w sposób gwarantujący możliwie największą szybkość dostępu do danych, realizowany na podstawie wskazania danych przez informacje identyfikujące (tzw. klucz).

Uwagi:

- W przypadku wyszukiwania liniowego może pojawić się problem grupowania, to znaczy koncentracji miejsc zajętych w pewnych zakresach indeksów przy małej zajętości innych obszarów tablicy. Problem ten jest w znacznym stopniu zredukowany w przypadku wyszukiwania kwadratowego i praktycznie wyeliminowany dla mieszania podwójnego.
- W przypadku adresowania otwartego istotnym problemem jest skomplikowanie procesu usuwania elementu, w sytuacji gdy w tablicy znajdują się inne, o tej samej wartości funkcji mieszającej.
- Wady tablic mieszających:
  - duża złożoność pesymistyczna wyszukiwania, wynosząca O(n),
  - duża złożoność czasowa związana z obliczaniem wartości dobrej funkcji mieszającej,
  - zwiększony czas wyszukiwania danych w tablicach o małych rozmiarach, związany z cache'owaniem danych w nowoczesnych mikroprocesorach – wyszukiwanie danych metodą sekwencyjną w małych tablicach przy danych ułożonych obok siebie będzie szybsze niż w przypadku użycia tablic mieszających.

Algorytmy numeryczne

Algorytmy numeryczne – służą do rozwiązywania zagadnień nie mających rozwiązania analitycznego lub gdy korzystanie z istniejących rozwiązań jest zbyt kosztowne.

Cechy:

- wykorzystywane są do rozwiązywania zagadnień matematycznych za pomocą komputera,
- korzystają z liczb zapisanych w formatach używanych w komputerach,
- obliczenia są zazwyczaj przybliżone – dokładność jest dobierany w zależności od typu algorytmu i potrzeb użytkownika.

Algorytmy numeryczne – służą do rozwiązywania zagadnień nie mających rozwiązania analitycznego lub gdy korzystanie z istniejących rozwiązań jest zbyt kosztowne.

Cechy:
- wykorzystywane są do rozwiązywania zagadnień matematycznych za pomocą komputera,
- korzystają z liczb zapisanych w formatach używanych w komputerach,
- obliczenia są zazwyczaj przybliżone – dokładność jest dobierany w zależności od typu algorytmu i potrzeb użytkownika.

Źródła błędów w obliczeniach numerycznych:
- błąd danych wejściowych,
- błędy zaokrągleń w czasie obliczeń.

Algorytmy numeryczne – służą do rozwiązywania zagadnień nie mających rozwiązania analitycznego lub gdy korzystanie z istniejących rozwiązań jest zbyt kosztowne.

Cechy:
- wykorzystywane są do rozwiązywania zagadnień matematycznych za pomocą komputera,
- korzystają z liczb zapisanych w formatach używanych w komputerach,
- obliczenia są zazwyczaj przybliżone – dokładność jest dobierany w zależności od typu algorytmu i potrzeb użytkownika.

Źródła błędów w obliczeniach numerycznych:
- błąd danych wejściowych,
- błędy zaokrągleń w czasie obliczeń.

Próbki danych są zaokrąglane, ze względu na brak możliwości przechowywania danych o nieskończonej liczbie cyfr.

Algorytmy numeryczne – służą do rozwiązywania zagadnień nie mających rozwiązania analitycznego lub gdy korzystanie z istniejących rozwiązań jest zbyt kosztowne.

Cechy:
- wykorzystywane są do rozwiązywania zagadnień matematycznych za pomocą komputera,
- korzystają z liczb zapisanych w formatach używanych w komputerach,
- obliczenia są zazwyczaj przybliżone – dokładność jest dobierany w zależności od typu algorytmu i potrzeb użytkownika.

Źródła błędów w obliczeniach numerycznych:
- błąd danych wejściowych,
- błędy zaokrągleń w czasie obliczeń.

Błędy związane z błędem odwzorowania i/lub braku nieskończonej dokładności. Na przykład reprezentację liczby 0.10d można zrealizować jako 0.00100000b, co jest w rzeczywistości równe 0.125d.

Próbki danych są zaokrąglane, ze względu na brak możliwości przechowywania danych o nieskończonej liczbie cyfr.

Algorytmy numeryczne – służą do rozwiązywania zagadnień nie mających rozwiązania analitycznego lub gdy korzystanie z istniejących rozwiązań jest zbyt kosztowne.

Nadmiar i niedomiar:

- W komputerach stosuje się następujący model zapisu liczby zmiennoprzecinkowej:

$$R=(-1)^{s}\cdot m\cdot 2^{C}$$

gdzie:
s – znak,
m – znormalizowana mantysa (część ułamkowa),
c – cecha (część całkowita).

- Na zapisanie całej liczby poświęca się określoną liczbę bitów, przy czym: za dokładność liczby odpowiada mantysa, za zakres liczby odpowiada cecha.
- Jeśli zapis liczby wymaga *d* bitów, to mantysę można wykorzystać *t* bitów, natomiast na zapis cechy (*d* – *t* – 1) bitów (jeden bit odejmuje się na znak), wtedy:
  - maksymalna cecha wynosi $c_{max}=-c_{min}=2^{d-t-1}$, a mantysa należy do przedziału: $\left\langle \frac{1}{2},1 \right)$
  - liczby, które można reprezentować są z przedziału:

$$-\frac{1}{2}\cdot 2^{C_{min}}\leqslant |R|\leqslant 2^{C_{max}}$$

  - nie będzie możliwości reprezentowania liczb spoza tego zakresu:
    – jeśli liczba posiada cechę $C < C_{min}$, to występuje tzw. niedomiar,
    – dla liczb, których cecha $C > C_{max}$ występuje nadmiar.

Algorytmy numeryczne – służą do rozwiązywania zagadnień nie mających rozwiązania analitycznego lub gdy korzystanie z istniejących rozwiązań jest zbyt kosztowne.

Zastosowania algorytmów numerycznych:

- całkowanie,
- znajdowanie miejsc zerowych wielomianów stopnia większego niż 2,
- rozwiązywanie układów równań liniowych w przypadku większej liczby równań i niewiadomych – korzystanie ze wzorów na dokładne wartości pierwiastków równań stopnia 3 i 4 jest niepraktyczne, dla równań stopnia wyższego niż 4 takich wzorów nie ma,
- rozwiązywanie równań i układów równań różniczkowych,
- znajdowanie wartości i wektorów własnych (wykorzystywanych w mechanice kwantowej),
- aproksymacje, czyli przybliżanie nieznanych funkcji (np. reprezentujących zmiany zjawisk fizycznych).
- ...

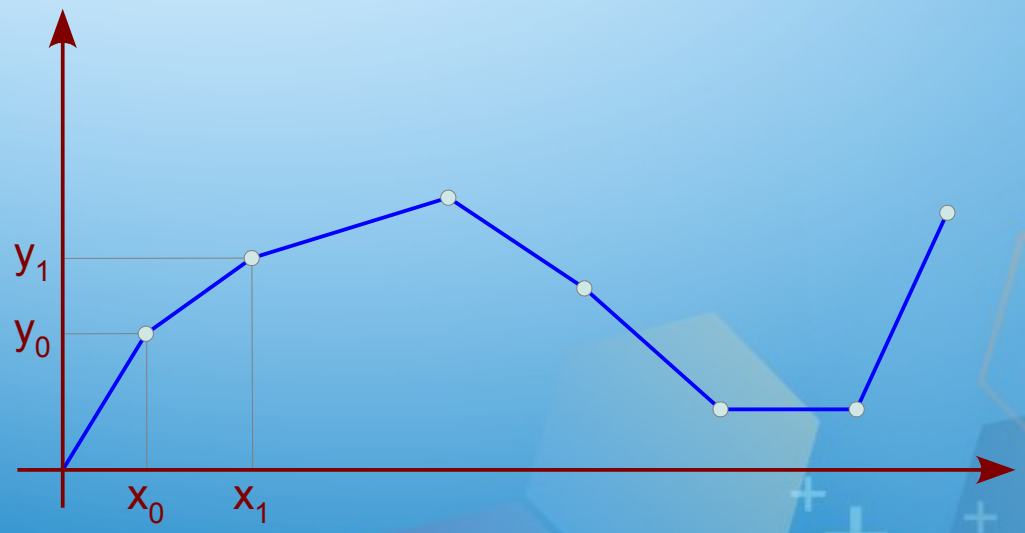
Interpolacja liniowa – polega na przybliżeniu wartości funkcji nieliniowej przez funkcję liniową o kreślonym przedziale liczbowym.

Algorytm:

Jeśli x określa wartość z przedziału $x_0 < x < x_1$, a $y_0 = f(x_0)$ i $y_1 = f(x_1)$ są wartościami danej funkcji, oraz $h = x_1 - x_0$ oznacza odstęp pomiędzy argumentami, wówczas liniowa interpolacja wartości $L(x)$ funkcji f dana jest jako:

$$L(x) = y_0 + \frac{y_1 - y_0}{h}(x - x_0)$$

Przykład:



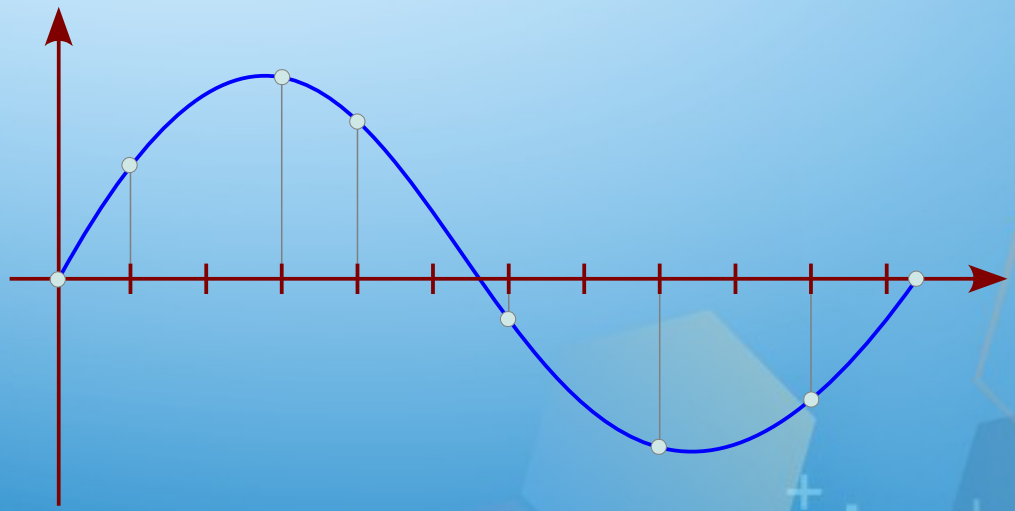
Interpolacja wielomianowa – interpolacja Lagrange'a – polega na znalezieniu wielomianu Lagrange'a stopnia n, przyjmującego w $n + 1$ punktach (węzłach) te same wartości co przybliżana funkcja.

Algorytm:

Metoda interpolacji polega na wybraniu $n + 1$ punktów $x_0, x_1, ..., x_n$ należących do dziedziny funkcji f, dla których są znane wartości tej funkcji: $y_0 = f(x_0), y_1 = f(x_1), ..., y_n = f(x_n)$.

Następnie znajduje się wielomian co najwyżej stopnia n, dla którego zachodzą zależności: $W(x_0) = y_0, W(x_1) = y_1, ... W(x_n) = y_n$.

Przykład:



Interpolacja wielomianowa – interpolacja Lagrange'a – polega na znalezieniu wielomianu Lagrange'a stopnia n, przyjmującego w n+1 punktach (węzłach) te same wartości co przybliżana funkcja.

Metoda wyboru wielomianu:

1. Dla pierwszego węzła o wartości $f(x_0)$ znajduje się wielomian, który w tym punkcie przyjmuje wartość $f(x_0)$, a w pozostałych węzłach, tj. w x_1, x_2, ..., x_n, ma wartość zero.
2. Dla kolejnego węzła znajduje się podobny wielomian, który w drugim węźle przyjmuje wartość $f(x_1)$, a w pozostałych węzłach x_0, x_2, ..., x_n, ma wartość zero.
3. Dodaje się do ostatnio obliczonego wielomianu do wielomian wyznaczony poprzednio.
4. Dla każdego z pozostałych węzłów znajduje się podobny wielomian, za każdym razem dodając go do wielomianu wynikowego.
5. Wielomian będący sumą wielomianów obliczonych dla poszczególnych węzłów jest wielomianem interpolującym.

Uwagi:

- w praktyce ten typ interpolacji ma zastosowanie dla niewielkiej liczby punktów.

Szybka transformata Fouriera (FFT) – to algorytm obliczający dyskretną transformatę Fouriera oraz transformatę do niej odwrotną.

Transformata Fouriera:

Dla danych liczb zespolonych $x_0, x_1, ..., x_{n-1}$ dyskretna transformata Fouriera dana jest jako:

$$X_k = \sum_{j=0}^{n-1} x_j e^{\frac{-2\pi i}{n} j \cdot k}$$

gdzie k = 0, 1, ... n-1. Złożoność obliczeniowa algorytmu wynosi $O(n^2)$.

Szybka transformata Fouriera (FFT) – to algorytm obliczający dyskretną transformatę Fouriera oraz transformatę do niej odwrotną.

Transformata Fouriera:

Dla danych liczb zespolonych $x_0, x_1, ..., x_{n-1}$ dyskretna transformata Fouriera dana jest jako:

$$X_k = \sum_{j=0}^{n-1} x_j e^{\frac{-2\pi i}{n} j \cdot k}$$

gdzie $k = 0, 1, ... n-1$. Złożoność obliczeniowa algorytmu wynosi $O(n^2)$.

Szybka transformata Fouriera:

Algorytm bazuje na metodzie „dziel i zwyciężaj”, przekształcając rekurencyjnie transformatę $N = N_1 \cdot N_2$ na transformaty N_1 i N_2, dla których złożoność obliczeniowa będzie wynosiła $O(n)$ operacji mnożenia. Najpopularniejszą wersją algorytmu jest FFT o podstawie 2, w której wektor próbek wejściowych musi mieć długość $n = 2^k$, gdzie k to pewna liczba naturalna. Złożoność obliczeniowa szybkiej transformata Fouriera wynosi $O(n \log_2 n)$.

Szybka transformata Fouriera (FFT) – to algorytm obliczający dyskretną transformatę Fouriera oraz transformatę do niej odwrotną.

Transformata Fouriera:

Dla danych liczb zespolonych $x_0, x_1, ..., x_{n-1}$ dyskretna transformata Fouriera dana jest jako:

$$X_k = \sum_{j=0}^{n-1} x_j e^{\frac{-2\pi i}{n} j \cdot k}$$

gdzie $k = 0, 1, ... n-1$. Złożoność obliczeniowa algorytmu wynosi $O(n^2)$.

Szybka transformata Fouriera:

Algorytm bazuje na metodzie „dziel i zwyciężaj”, przekształcając rekurencyjnie transformatę $N = N_1 \cdot N_2$ na transformaty N_1 i N_2, dla których złożoność obliczeniowa będzie wynosiła $O(n)$ operacji mnożenia. Najpopularniejszą wersją algorytmu jest FFT o podstawie 2, w której wektor próbek wejściowych musi mieć długość $n = 2^k$, gdzie k to pewna liczba naturalna. Złożoność obliczeniowa szybkiej transformata Fouriera wynosi $O(n \log_2 n)$.

Zastosowanie:

- cyfrowe przetwarzanie sygnałów (DSP),
- pochodną algorytmu jest dyskretna transformata cosinusowa (DCT), stosowana w algorytmach kompresji danych: JPEG, MP3 itp.

Koniec wykładu