**ZŁOŻONOŚĆ OBLICZENIOWA**

Złożoność obliczeniowa to ilość zasobów komputerowych koniecznych do wykonania programu realizującego algorytm. Przedstawiamy ją jako funkcję pewnego parametru, określającego rozmiar rozwiązywanego zadania. Wyróżniamy złożoność pamięciową i czasową.

Złożoność pamięciowa to ilość pamięci wykorzystanej w celu realizacji algorytmu, wyrażana w liczbie bajtów lub liczbie zmiennych typów elementarnych. Złożoność czasowa jest to czas wykonania algorytmu. Złożoności czasowej nie wyrażamy w standardowych jednostkach czasu, gdyż w zależności od tego, jaką maszyną dysponujemy, algorytm będzie wykonywał się na różnych maszynach w różnym czasie. Dlatego złożoność obliczeniową czasową wyrażamy w ilości operacji dominujących.

* Złożoność optymistyczna określa zużycie zasobów dla najkorzystniejszego zestawu danych.
* Złożoność średnia określa zużycie zasobów dla typowych (losowych) danych.
* Złożoność pesymistyczna określa zużycie zasobów dla najbardziej niekorzystnego zestawu danych.

Złożoności wielomianowe:

* Liniowa
* Kwadratowa
* Sześcienna
* Logarytmiczna
* Quasiliniowa
* NP-zupełna
* Wykładnicza
* Silnie NP-zupełna
* NP-trudna

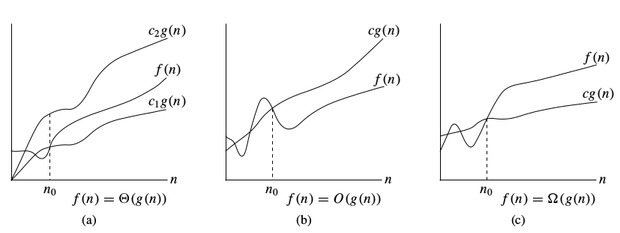
Notacje złożoności obliczeniowej:

* Notacja O (“duże O”)
* Notacja Θ
* Notacja Ω

Mówimy, że (funkcja złożoności obliczeniowej jest rzędu funkcji ,   
jeśli potrafimy znaleźć takie oraz takie , iż dla każdego prawdziwa jest nierówność:

Mówimy, że jeśli istnieją stałe dodatnie , i   
takie , że dla każdego prawdziwa jest nierówność:

Mówimy, że jeśli istnieją stałe dodatnie i takie, że dla każdego prawdziwa jest nierówność:

**MODELE OBLICZEŃ**

Automat skończony deterministyczny , to dowolna taka piątka , w której:

* to skończony zbiór stanów,
* to skończony alfabet wejściowy,
* to funkcja przejścia - to stan, do którego przechodzimy ze stanu q na skutek wczytania znaku a,
* to stan początkowy,
* to zbiór stanów akceptujących.

Automat skończony wczytuje dane słowo znak po znaku i po wczytaniu całego słowa udziela odpowiedzi, czy słowo to należy do ustalonego języka. Przejścia między stanami zachodzą w automacie skończonym na skutek wczytywania kolejnych znaków analizowanego słowa. Automaty skończone odpowiadają klasie gramatyk liniowych wg. hierarchii Chomsky’ego.

W automacie deterministycznym, dla każdego znaku jaki może się pojawić na wejściu i dla każdego stanu automatu, ze stanu tego wychodzi dokładnie jedno przejście odpowiadające danemu znakowi – w każdej sytuacji działanie automatu musi być określone jednoznacznie.

Automat skończony ma skończoną liczbę stanów. Oznacza to, że dysponuje on stałą pamięcią dodatkową, tzn. wielkość pamięci dodatkowej nie zależy od długości wczytywanego słowa i jest ograniczona przez stałą.

Automat skończony niedeterministyczny , to dowolna taka piątka , w której:

* to skończony zbiór stanów,
* to skończony alfabet wejściowy,
* to funkcja przejścia - to stan, do którego przechodzimy ze stanu q na skutek wczytania znaku a,
* to zbiór stanów początkowych,
* to zbiór stanów akceptujących.

Automat niedeterministyczny jest automatem, którego działania nie da się w pełni przewidzieć. Będąc w tym samym stanie i wczytując ten sam znak może wykonać różne przejścia. Przejść może być dowolnie wiele, a nawet może ich nie być wcale. Automat taki może również mieć wiele stanów początkowych.

Automat ze stosem (stosowy) to dowolna taka szóstka , w której:

* to skończony zbiór stanów,
* to skończony alfabet wejściowy,
* to skończony alfabet stosowy,
* to skończona relacja przejścia – określa ona dla danego aktualnego stanu, znaku na wierzchołku stosu i znaku czekającego na wczytanie, do jakiego stanu należy przejść i co należy włożyć na stos (uprzednio zdjąwszy element z wierzchołka stosu)
* to stan początkowy,
* to symbol początkowy na stosie

Automat stosowy, podobnie jak automat skończony, ma skończony zbiór stanów i stan początkowy. Dodatkowo jest on wyposażony w stos, na który może wkładać elementy, podglądać element znajdujący się na wierzchu i zdejmować elementy. Przejścia automatu stosowego są trochę bardziej skomplikowane niż przejścia w automacie skończonym, gdyż oprócz wczytywania znaków z wejścia i zmiany stanów obejmują również operacje na stosie. Automat taki może być niedeterministyczny i może zawierać   
ε-przejścia, tzn. wykonaniu przejścia nie musi towarzyszyć wczytanie znaku z wejścia. Automaty stosowe są pozbawione stanów akceptujących – automat będzie sygnalizował akceptację wejścia opróżniając stos.

Automat stosowy w jednym kroku obliczeń wykonuje następujące czynności:

1. Podgląda znak czekający na wczytanie na wejściu.
2. Podgląda element na wierzchołku stosu.
3. Na podstawie tych informacji wybiera jedno z przejść do wykonania.
4. Jeśli to nie jest ε-przejście, to wczytywany jest jeden znak z wejścia.
5. Zdejmowany jest element z wierzchołku stosu.
6. Pewna liczba określonych elementów może być włożona na stos.
7. Zgodnie z przejściem zmieniany jest stan.

**SCHEMAT BUDOWY PROCESORA**

Procesor, także CPU (*Central Processing Unit*) – to główny element komputera, urządzenie cyfrowe sekwencyjne, którego zadaniem jest wykonywanie rozkazów i sterowanie pracą wszystkich pozostałych bloków systemu (m.in. pamięci i układów wejścia-wyjścia). Procesor przetwarza informacje, wykonując na niej elementarne operacje zwane instrukcjami maszynowymi (bądź rozkazami). Ciąg takich instrukcji realizujący konkretne zadanie przetwarzania informacji nazywamy programem.

Wszystkie mikroprocesory zawierają podobne elementy:

1. układ sterowania i synchronizacji, który kontroluje pracę procesora i wytwarza sygnały potrzebne do sterowania niektórymi elementami komputera
2. arytmometr, czyli układ, który wykonuje operacje arytmetyczne i logiczne
3. rejestry, tj. układy pamięci
4. wbudowana pamięć podręczna cache
5. koprocesor matematyczny
6. wewnętrzne szyny łączące elementy procesora

**UKŁADY SEKWENCYJNE I KOMBINACYJNE**

Układem kombinacyjnym nazywamy układ cyfrowy, w którym sygnały wyjsciowe sa jednoznacznie okreslone przez aktualne wartosci sygnałów wejsciowych. Układ kombinacyjny o wejsciach i wyjsciach jest w pełni opisany przez podanie funkcji przełaczajacej , gdzie jest zbiorem dopuszczalnych stanów wejsc, a jest zbiorem mozliwych stanów wyjść.

System funkcjonalnie pełny to zestaw typów bramek, z których mozna zbudowac dowolny układ kombinacyjny.

Przykłady najwazniejszych systemów bramek funkcjonalnie pełnych:

* AND, OR, NOT
* NAND
* NOR.

Przykłady układów kombinacyjnych: sumator, multiplekser, demultiplekser.