1 Informatyka

Słowo *informatyka* w ostatnich czasach bardzo się rozpowszechniło. Zazwyczaj kojarzy się je jednoznacznie z komputerami, co nie jest całkowicie słuszne. W Stanach Zjednoczonych używa się w zamian pojęcia *computer science*, co prawdopodobnie wprowadza niektórych w błąd. W rzeczywistości informatyka, to bardzo szeroka gałąź nauki zajmująca się teoria informacji, metodami jej przechowywania, przekazywania i przetwarzania. Z pojęciem informacji jest jeszcze większy problem, jako że bardzo trudno jest znaleźć jej dobrą definicję. Nie należy mylić informacji z jej reprezentacją. Tę samą informację można reprezentować na wiele różnych sposobów - bardziej lub mniej dla nas zrozumiałych. Informacja różni się też do wiedzy, mimo że niektóre słowniki tak ją definiują. Zazwyczaj posługujemy się pojęciem informacji bez uprzedniego definiowania, pozostawiając jego rozumienie intuicjom odbiorcy.

Często dzieli się informatykę na dwie główne części, jedna dotyczy sprzętu, druga oprogramowania. Jednak często praca informatyków nie ma wiele wspólnego z komputerami, które służą im czasami tylko i wyłącznie jako inteligentne maszyny do pisania.

Jedną z ważniejszych gałęzi informatyki jest *algorytmika*. Zajmuje się ona nie tylko tworzeniem algorytmów, ale i testowaniem ich poprawności i badaniem ich właściwości. Dowodzenie poprawności programów to bardzo żmudna praca, jako że wymaga głębokiej analizy każdej (chociażby najdrobniejszej) instrukcji. Pokrewną dziedziną jest badanie poprawności układów elektronicznych (ang. *hardware verification*). Każdy procesor, czy inny układ też realizuje pewien algorytm i żeby być pewnym, że komputer wykona nasze instrukcje poprawnie powinniśmy wcześniej sprawdzić poprawność jego konstrukcji. Istnieje bardzo wiele systemów dowodzenia i weryfikacji twierdzeń, które mogą takie zadania spełniać. Firma Intel po ostatnich błędach wykrytych w procesorze Pentium podjęła wysiłki mające na celu stworzenie systemu, który mógłby weryfikować produkowane przez nich układy zanim rozpocznie się ich masowa sprzedaż.

Algorytmika bada również złożoność algorytmów. Jeden problem może być rozwiązywany na wiele różnych sposobów. Istnieje wiele różnych metod sortowania obiektów (ustawiania napisów w kolejności alfabetycznej, liczb rosnąco itp.). Niektóre metody dla posortowania N obiektów potrzebują wykonać N^2 operacji (mówimy o nich, że są złożoności N^2), inne potrafią to samo zadanie wykonać kosztem N instrukcji, co docenimy zwłaszcza wtedy, gdy musimy sortować duże ilości obiektów.

Informatyka zajmuje się również reprezentacją informacji. Danym trzeba przyporządkować pewne typy, często łączy się je w bardziej złożone struktury takie jak tablice, macierze, listy, kolejki, drzewa. Jest to tak ważny i pojemny zbiór problemów, że można dziś mówić o specjalnej dziedzinie informatyki zajmującej się strukturami danych.

W końcu gałęzią informatyki jest również nauka sztuki programowania i samo programowanie. Kolejną dziedziną jest teoria języków programowania. Takie rozważania teoretyczne można prowadzić nie dotykając w ogóle komputera. Dzisiaj każdy tworzony język programowania posiada solidne podstawy matematyczne, choć pierwsze języki powstawały bez żadnego zaplecza teoretycznego.

Dzisiaj informatyka i komputery są już obecne praktycznie w każdej z dziedzin nauki.

Często mówi się o różnych naukach komputerowych, za których przykłady mogą służyć:

- Matematyka komputerowa. Istnieje wiele dziedzin matematyki, w których korzysta się z komputerów. Na przykład w teorii liczb korzysta się z maszyn liczących dla badania własności liczb, czy sprawdzania hipotez na dużych liczbach. Bardzo prosty w sformułowaniu, choć niesamowicie trudny do udowodnienia problem czterech barw doczekał się dzięki komputerom sprawdzenia dla wielu wyrafinowanych przypadków. Problem ten definiowany jest jako stwierdzenie „*Każdą mapę można pokolorować czterema różnymi kolorami tak, aby każde dwa państwa, które ze sobą sąsiadują otrzymały różne kolory*". Stwierdzenie to czekało bardzo długo na dowód, aż doczekało się takiego, co do którego poprawności nie ma pewności, jako że jest zbyt skomplikowany, żeby można było go zrozumieć. W logice komputery przyczyniły się do stworzenia projektu QED (łac. *quod erat demonstrandum* czyli co należało dowieść), który za cel stawia sobie sformalizowanie i zweryfikowanie wszystkich dowodów matematycznych, jakie kiedykolwiek opublikowano.

- Fizyka komputerowa. Mocy obliczeniowa przychodzi z pomocą fizykom cząstek elementarnych, pozwala na wykonywanie symulacji atmosfery, czy też pozwala uniknąć konieczności wykonywania prób nuklearnych na rzecz komputerowych symulacji.

- Chemia komputerowa. Gałąź nauki która łączy chemię i fizykę (chemia kwantowa) potrafi wykorzystać doszczętnie każdą moc maszyn, jako, ze obliczenia energii cząsteczek wymagają bardzo dużych kalkulacji zabierających komputerom całe dni, tygodnie albo i dłuższe okresy.

- Biologia komputerowa korzysta ze zdobyczy informatyki dla wykonywania symulacji ekologicznych czy też badania obiegu substancji i energii w przyrodzie)

- Nauki o poznaniu (ang. *cognitive sciences* wykorzystują komputery do symulowania zjawisk zachodzących w mózgu.

- Ekonomia komputerowa dzięki komputerom może przewidywać rozwój gospodarki na poprzez wykorzystanie dostępnych danych do prowadzenia symulacji. Oczywiście wszelkie tego typu symulacje nie dają 100-procentowej pewności, a stworzenie takiego systemu byłoby bardzo korzystne ze względu na możliwość zweryfikowania poglądów niektórych polityków zanim zaczną eksperymentować na swoich wyborcach.

- Nauki humanistyczne także nie uchroniły się od wpływu komputerów. Maszyny wykorzystywano np. do badań mających na celu ustalenie na którą wyspę rzeczywiście dopłynął Kolumb odkrywając Amerykę. Komputery są wykorzystywane w archeologii, geografii, a nawet w badaniach literackich.

- Nauki prawnicze korzystając z systemów baz danych mogą istotnie usprawnić sobie wyszukiwanie informacji w gąszczu przepisów.

2 Systemy operacyjne

Sprzęt komputerowy sam w sobie nie stanowi żadnego użytecznego urządzenia. Dopiero w połączeniu z oprogramowaniem zaczyna być godny zainteresowania. Jeden z typów oprogramowania stanowią *systemy operacyjne*. Precyzyjne zdefiniowanie czym one są jest dość trudnym zadaniem. Łatwiej jest opisać zadania takiego systemu:

> System operacyjny nadzoruje i koordynuje posługiwanie się sprzętem przez różne programy użytkowe, które pracują na zlecenie różnych użytkowników.

System operacyjny jest więc podstawowym oprogramowaniem komputera, które zarządza całością sprzętu. Stanowi pomost pomiędzy użytkownikiem i uruchamianymi przez niego programami a komputerem. Zadanie systemu operacyjnego w komputerze można porównać do zadania rządu w państwie. Sam nie wykonuje żadnych użytecznych funkcji. Tworzy tylko środowisko, w którym inne programy mogą wykonywać pożyteczne prace. System operacyjny jest więc zarządcą zasobami komputera.

2.1 Geneza systemów operacyjnych

Aby łatwiej zrozumieć istotę systemu operacyjnego, dobrze jest spojrzeć wstecz i zobaczyć w jaki sposób doszło do stworzenia takiego systemu.

Pierwsze komputery obsługiwane były całkowicie przez człowieka. Program zapisywano na taśmach (lub kartach) perforowanych, które były bezpośrednio czytane i analizowane przez główna jednostkę przetwarzania. Wówczas programistą i operatorem była najczęściej ta sama osoba. Mogła więc ona bezpośrednio po pojawieniu się błędu wykonania programu usuwać go i powtórzyć pracę od początku. Już wtedy zauważono, że obsługa pewnych standardowych zdarzeń potrzebna jest większości programów i zazwyczaj wygląda bardzo podobnie, jeśli nie dokładnie tak samo. To spostrzeżenie doprowadziło do pojawienia się pewnych ogólnych funkcji obsługi urządzeń. Tak powstawały pierwsze *biblioteki* czyli zestawy funkcji, które można bez zmiany kodu źródłowego wielokrotnie wykorzystywać. Bardzo szybko okazało się też, że potrzebne jest opracowanie języka, który pozwoliłby użytkownikowi pisać zrozumiałe dla niego instrukcje zamiast mało czytelnych symboli rozumianych przez komputer. Tak powstały pierwsze kompilatory pierwszych języków programowania: Fortran i Cobol.

Konieczność obsługi komputera przez samego autora programu wymagała prowadzenia harmonogramów pracy maszyny. Żeby wykonać swoje obliczenia należało uprzednio zarezerwować dla siebie maszynę, z góry dokładnie określając początek i koniec pracy. Wiązało się to z pewnym marnotrawieniem mocy obliczeniowych. Kiedy jakiś użytkownik zakończył swoje zadanie wcześniej komputer czekał bezczynnie na następnego operatora. Z drugiej strony jeśli ktoś nie zdążył osiągnąć zamierzonych efektów musiał szukać kolejnego terminu i rezerwować maszynę jeszcze raz. Szybko jednak znaleziono pewne metody usprawnienia pracy. Przede wszystkim wprowadzono zawód operatora. Programiści zlecali jemu zadania, a on wykonywał je na maszynie. Miało to również swoje wady, gdyż

utrudniało szybkie poprawianie prostych błędów w programach. Zaczęto też grupować „podobne" zadania. Programy, które wymagały uprzedniego załadowania fortranu stawały się jedną grupą te pisane w Cobolu inną itd. Dzięki temu można było załadować dany kompilator tylko raz na początku grupy zadań zamiast ładować go każdorazowo przed wymagającym go programem.

Operator komputera miał ograniczone możliwości kontrolowania stanu uruchamianych procesów. Kiedy maszyna przestała wysyłać wyniki nie wiadomo było, czy to dlatego, że trwają obliczenia, czy też program wpadł w nieskończoną pętle. Powstała więc idea *automatycznego porządkowania procesów* (ang. *automatic job sequencing*). Polegało to na tym, że pewien mały program zarządzał całością pracy maszyny. Zwany był on *monitorem rezydującym*, ponieważ ładowany był do pamięci operacyjnej zaraz po uruchomieniu komputera i tam pozostawał przez cały czas działania maszyny. Składał się z trzech modułów: programu ładującego, modułu porządkowania zadań oraz interpretatora kart sterujących. Na specjalnych kartach zapisywano mu informacje niezbędne do odpowiedniego ładowania zadań i ich danych.

Rozwój jednostek przetwarzających informacje był i jest dużo szybszy, niż rozwój urządzeń wejścia-wyjścia. Gdy do wprowadzania danych używano kart i taśm perforowanych różnica prędkości dawała się szczególnie odczuć. Kiedy pojawiły się taśmy magnetyczne o dużo szybszym dostępie wprowadzono *tryb pośredni* w pracy urządzeń zewnętrznych. Dane z kart były kopiowane na taśmy przez osobne urządzenia, a procesor czytał dane bezpośrednio z taśm, co znacznie skróciło przestoje procesora. Podobnie zorganizowano przepływ danych wyjściowych. Już wtedy pojawiły się pierwsze dążenia do uniezależnienia oprogramowania od urządzeń na których jest uruchamiane. Pozwalało to unikać modyfikacji kodu programów z każdą zmianą sprzętową.

Oczywiście wprowadzenie trybu pośredniego nie zaowocowało całkowitym wyeliminowaniem oczekiwania procesora na dane. Zrodził się więc pomysł *buforowania*. Polega on na równoległym wykonywaniu obliczeń i operacji wejścia-wyjścia, przez co dane można przygotować procesorowi zanim o nie poprosi przyspieszając tym samym cały proces.

Rozwinięciem idei buforowania jest *spooling* (ang. *simultaneous peripheral operation on-line*). Dzięki wykorzystaniu dysków (znacznie szybszych od taśm magnetycznych), udało się niemal całkowicie uniezależnić wzajemnie wejście, procesor i wyjście. Dane wejściowe i wyjściowe mogą być składowane na dysku, przez co można nawet wykonywać równolegle operacje wyjściowe jednego procesu, obliczenia innego oraz operacje wejściowe kolejnego zadania.

Inteligentniejsze systemy posiadają cechę *wieloprogramowości*. Polega ona na tym, że tworzy się kolejkę zadań do wykonania i w przypadku gdy pierwsze z nich zacznie czekać na jakieś zdarzenie (np. na wprowadzenie danych przez użytkownika), wówczas drugie zadanie z kolejki jest ładowane do procesora. Kiedy i to zacznie czekać - następne itd. Kiedy już pierwsze z zadań otrzyma oczekiwany sygnał i będzie mogło dalej pracować przejmie procesor, a inne zadania będą czekać na jego zakończenie.

Jeszcze lepszym rozwiązaniem jest *wieloprocesowość*. W systemach wieloprocesowych wszystkie zasoby komputera są dzielone pomiędzy aktywne procesy. W szczególności wszystkie zadania dzielą między siebie czas procesora: system operacyjny przydziela im

kolejno (bardzo krótkie) odcinki czasu, przez co robią one wrażenie równoległego działania.

2.2 Systemy rozproszone

W tworzonych ostatnio systemach komputerowych zmierza się często do rozdzielania zadań między wiele procesorów. Istnieją systemy, w których procesory używają wspólnej pamięci operacyjnej i taktowane są wspólnym zegarem. Są jednak i takie, w których wiele całkowicie różnych maszyn (często nawet fizycznie od siebie odległych) dzieli się zadaniami. Są to tzw. *systemy rozproszone*. Systemy takie mają kilka podstawowych zalet:

Podział zasobów. Dzięki połączeniu komputerów mogą one korzystać ze wspólnych urządzeń. Nie trzeba wówczas każdego komputera kompletnie wyposażać. Jeden może udostępniać innym swoją drukarkę, inny swoje zasoby dyskowe, dzięki czemu nie trzeba tworzyć wielu kopii oprogramowania i wielu użytkowników może korzystać jednocześnie z tych samych danych.

Przyśpieszenie obliczeń. W sytuacji kiedy zadanie składa się z kilku niezależnych od siebie części można każdy z kawałków zlecić innemu procesorowi, przez co zadanie wykona się szybciej.

Niezawodność. Kiedy jeden z węzłów sieci ulegnie awarii, reszta systemu może wciąż funkcjonować, a nawet przejąć zadania uszkodzonego podsystemu.

Łączność. Łączenie komputerów w sieci daje możliwość porozumiewania się między sobą procesów nawet bardzo od siebie odległych. Dzięki temu możliwe jest na przykład funkcjonowanie *poczty elektronicznej*, bez której wielu ludzi nie potrafiłoby już dzisiaj sprawnie funkcjonować.

2.3 Systemy czasu rzeczywistego

Niekiedy istnieje potrzeba przetwarzania danych na bieżąco - nie można sobie pozwolić na jakiekolwiek opóźnienia, gdyż na niektóre zaistniałe sytuacje trzeba reagować natychmiast. Tak szybkie i sprawne systemy są zazwyczaj niezbędne do

- nadzorowania eksperymentów naukowych
- sterowania procesami przemysłowymi
- wizualizacji danych pochodzących z czujników czy mierników

Na nic zdałby się system sterujący produkcją samochodów, który mógłby nie zdążyć wstrzymać ruchu ramienia robota zanim tez zgniótłby karoserię budowanego pojazdu.

2.4 Systemy jednostanowiskowe

Systemy, które mają służyć tylko jednej osobie nie muszą zazwyczaj spełniać najwyższych wymagań. Najczęściej stosuje się w nich mechanizmy buforowania i spoolingu, rzadziej spotykamy w nich wielozadaniowość i możliwość pracy w sieci.

3 Struktury systemów komputerowych

Procedury z obsługą przerwań

- zapamiętanie adresu przerwanego rozkazu
- wyłączanie innych przerwań
- priorytety przerwań - przykład złącza asynchronicznego 1200 bodów: znaki przychodzą co ok. $8ms$ ($8000\mu s$, a procedura obsługi może wstawić znak do bufora w $20\mu s$
- szybsze urządzenia korzystają z DMA (ang. *direct memory access*)

Struktura wejścia-wyjścia

- system sterowany zdarzeniami - przerywane oczekiwanie
- urządzenie wejścia-wyjścia sygnalizuje przerwanie
  - na początku operacji - każe procesorowi czekać
  - po zakończeniu operacji
- tablica stanów urządzeń
- dualny tryb pracy - tryb monitora i użytkownika
- ochrona sprzętowa
  - system pośredniczy w wykonywaniu rozkazów uprzywilejowanych
  - system musi chronić wektory przerwań i procedury obsługi
  - rejestry bazowy i graniczny
  - zastosowania przerwania zegarowego
    * zapobieganie nieskończonym pętlom
    * podział czasu między procesy
    * pamiętanie bieżącego czasu

Programy mogą obsługiwać urządzenia wejścia-wyjścia za pośrednictwem systemu operacyjnego:

- specjalny rozkaz systemu
- dowolny wyjątek (ang. *exception*)

Różne klasy komputerów

- systemy wieloprocesorowe
  - łagodna degradacja (awaria jednego z procesorów)
  - procesor zapasowy (system Tandem)
  - wieloprzetwarzanie symetryczne - kopia systemu operacyjnego pracuje na każdym procesorze (Encore Unix na Multimax)
  - wieloprzetwarzanie asymetryczne (z procesorem głównym) - Remote Job Entry
- Komputery osobiste
  - buforowanie, spooling, wielozadaniowość
  - brak ochrony
  - coraz większa potrzeba znajomości systemu operacyjnego
- Ewolucja systemów
  - 1965-1970 - MULTICS na GE645 (MIT)
  - 1970 - Unix na PDP-11
  - 1980 - Unix na mikrokomputerach

Struktury systemów operacyjnych

- system podzielony na moduły
- zarządzanie procesami
  - tworzenie i usuwanie procesów
  - zatrzymywanie i wznawianie procesów
  - dostarczanie mechanizmów synchronizacji procesów
  - dostarczanie mechanizmów komunikacji między procesami
  - dostarczanie mechanizmów obsługi blokad
- zarządzanie pamięcią operacyjną
  - ewidencja zajętych obszarów pamięci i ich własności
  - decydowanie o tym, które procesy są ładowane do pamięci
  - przydzielanie i zwalnianie pamięci
  - zbieranie śmieci (garbage collection)
- zarządzanie pamięcią pomocniczą
  - zarządzanie wolnymi obszarami
  - przydzielanie pamięci
- zarządzanie systemem wejścia-wyjścia
  - system buforowo-notatnikowy
  - interfejs do programów obsługi urządzeń sprzętowych
  - programy obsługi poszczególnych urządzeń
- zarządzanie plikami
  - tworzenie i usuwanie plików zwykłych i katalogów
  - elementarne operacje do manipulowania plikami
  - odwzorowywanie plików na obszary pamięci pomocniczej
  - składowanie na trwałych nośnikach
- system ochrony
- praca sieciowa
- system interpretacji poleceń

Usługi systemu operacyjnego

- wykonywanie programów
- operacje wejścia-wyjścia
- manipulowanie systemami plików
- komunikacja (pamięć dzielona, przesyłanie komunikatów)
- wykrywanie błędów
- podział zasobów
- rozliczanie
- ochrona danych

Programy systemowe

- manipulowanie plikami
- tworzenie i zmiana zawartości plików
- informowanie o stanie systemu
- kompilatory i interpretatory języków programowania
- programy komunikacyjne
- programy użytkowe, interpretator poleceń

Struktura systemu

- prosta struktura (MS-DOS)
- podejście warstwowe (Unix)
- maszyny wirtualne
  - każdy proces otrzymuje wirtualną kopię komputera
  - rozłączność różnych sesji - nie potrzeba mechanizmów ochrony
  - brak podziału zasobów - wirtualne sieci