PB001: Úvod do informačních technologií

Luděk Matyska

Fakulta informatiky Masarykovy univerzity

podzim 2013

Obsah přednášky

1. Programové vybavení
2. Číselné soustavy
3. Operační systémy

Programové vybavení

- Nadstavba technických prostředků
- Vrstvy operačního systému:
  - Technické vybavení
    Správa paměti
    Správa procesů
    Správa periferií (I/O)
    Správa souborů dat
    Uživatelský prostor (nepřesné)

Programové vybavení – jiný pohled

- Operační systém
  - UNIX, Linux, OS/370, MS Windows, ...
- Programovací jazyky
  - C, Pascal, Ada, Occam, ML, Prolog, perl, python, Java, ...
- Podpůrné nástroje
  - debuggery, profilery, ...
- Aplikační programy

Programovací jazyky

- Rozlišujeme
  - Styl
  - Míru abstrakce
  - „Dialekt“

Programovací jazyky – styl

- Imperativní/Procedurální: C, Fortran
- Objektově orientované: Java, C++
- Deklarativní/Funkcionální: ML, Lisp, MIRANDA
- Deklarativní/Logické: Prolog, GHC
- S jediným přiřazením: SISAL
- Produkční systémy: OPS5
- Sémantické sítě: NETL
- Neuronové sítě: SAIC ANSpec

Procedurální vs. deklarativní styl

```
fac := 1;
if n > 0 then
  for i:=1 to n do
    fac := i*fac;
```

```
|   fac(0)   := 1;
|   fac(n>0) := n*fac(n-1);
|
|
|-----------------------------
|
|   fac(0,1).
|   fac(N,F1*N) :- fac(N-1,F1).
|
```

Programovací jazyky – míra abstrakce

- Strojový jazyk: přímo kódy jednotlivých instrukcí
- Assembler: jména instrukcí, operandy, pojmenované cílové adresy skoků
- Vyšší jazyky: obecné konstrukty, tvoří „kontinuum“
  - Agregované datové typy
  - Cykly namísto skoků
  - Procedury a funkce
  - Procesy a vlákna

Programovací jazyky – implementace

- Překladače
  - Zdrojový kód–mezijazyk–strojový jazyk
  - Překlad a sestavení

Programovací jazyky – implementace

- Překladače
  - Zdrojový kód–mezijazyk–strojový jazyk
  - Překlad a sestavení
- Interprety
  - Abstraktní počítač
  - Vhodné pro složité operace (např. práce s texty, s maticemi a algebraickými objekty)

Programovací jazyky – implementace

- Překladače
  - Zdrojový kód–mezijazyk–strojový jazyk
  - Překlad a sestavení
- Interprety
  - Abstraktní počítač
  - Vhodné pro složité operace (např. práce s texty, s maticemi a algebraickými objekty)
- Just-in-time překladače (nejen Java)
  - Známy již od osmdesátých let (řešil se tak nedostatek paměti)

Výpočetní model

- Souvislost mezi *architekturou* a *jazyky*
  - von Neumannova architektura a imperativní jazyky
  - objektová architektura a objektově orientované jazyky
  - redukční architektura a funkcionální programování

Výpočetní model – varianty

- Datově orientovaný (nejpoužívanější)
- Objektový
- Funkcionální
- Logický

Aplikace

Dávají počítačům smysl

- Vědecko-technické výpočty (vojenství: atomová bomba)

Aplikace

Dávají počítačům smysl

- Vědecko-technické výpočty (vojenství: atomová bomba)
- Zpracování informací (Hollerith/IBM: census USA)

Aplikace

Dávají počítačům smysl

- Vědecko-technické výpočty (vojenství: atomová bomba)
- Zpracování informací (Hollerith/IBM: census USA)
- Zábava (počítačové hry, video-on-demand)

Aplikace

Dávají počítačům smysl

- Vědecko-technické výpočty (vojenství: atomová bomba)
- Zpracování informací (Hollerith/IBM: census USA)
- Zábava (počítačové hry, video-on-demand)
- Řízení (management strojů i lidí, real-time systémy)

Číselné soustavy

- Definovány základem: desítková, dvojková, osmičková, šestnáctková
- Volně mezi sebou převoditelné (celá čísla bez ztráty přesnosti)
- Celá čísla a zlomky
- Reálná čísla
- Konečná reprezentace

Číselné soustavy

- Definovány základem: desítková, dvojková, osmičková, šestnáctková
- Volně mezi sebou převoditelné (celá čísla bez ztráty přesnosti)
- Celá čísla a zlomky
- Reálná čísla
- Konečná reprezentace
- První počítače v desítkové soustavě

Dvojková soustava

- Základ číslo dvě:
  - pouze dvě číslice/dva stavy
  - vhodná pro reprezentaci v elektrických systémech

Dvojková soustava v počítači

- Konečná reprezentace: interval hodnot
- Pro reálná čísla:
  - Rozlišitelnost (nejmenší zobrazitelné číslo): $\mathbf{X} + \epsilon > \mathbf{X}$ a $\mathbf{X} + \epsilon/2 = \mathbf{X}$
  - Přesnost (rozsah)
  - Zobrazení: mantisa $\mathtt{m}$ a exponent $\mathtt{e}$
    $\mathbf{0} \leq \mathtt{m} \leq \mathbf{1} \wedge \mathbf{x} = \mathbf{m}.\mathbf{2}^{\mathtt{e}}$
- Záporná čísla:
  - Přímý kód
  - Inverzní kód
  - Dvojkový doplňkový kód

Záporná čísla – zobrazení

- Přímý kód:
  - Přidáme znaménko
  - Dvě nuly: $+\mathbf{0}$ a $-\mathbf{0}$ ($\mathbf{10\ldots00}$)
  - Rozsah: $\langle -\mathrm{MAX}, -\mathbf{0}\rangle$ a $\langle +\mathbf{0}, +\mathrm{MAX}\rangle$
- Inverzní kód:
  - Přidáme znaménko
  - Dvě nuly: $+\mathbf{0}$ a $-\mathbf{0}$ ($\mathbf{11\ldots11}$)
  - Rozsah: $\langle -\mathrm{MAX}, -\mathbf{0}$ a $\langle +\mathbf{0}, +\mathrm{MAX}\rangle$

Záporná čísla – zobrazení

- Dvojkový doplňkový kód:
  - Inverze bitu a přičtení jedničky
  - Pouze jedna nula ($\mathbf{11\ldots11}$ je $\mathbf{-1}$)
  - Nesymetrický rozsah: $\langle -\mathrm{MAX} - \mathbf{1}, \mathbf{-1}\rangle$ a $\langle \mathbf{+0}, +\mathrm{MAX}\rangle$
- Skutečně používán v počítačích

Rozsahy čísel

- Podle počtu bitů:
  - Byte: 8 bitů, tj. $\langle \mathbf{0, 255} \rangle$ nebo $\langle \mathbf{-128, 127} \rangle$
  - Půl slovo, 2 byte: 16 bitů, tj. $\langle \mathbf{0, 65535} \rangle$ nebo $\langle \mathbf{-32768, 32767} \rangle$
  - Slovo, 4 byte: 32 bitů, tj. přibližně $\langle \mathbf{-2.10^9, 2.10^9} \rangle$
  - Dvojslovo (nebo dlouhé slovo), 8 byte: 64 bitů, tj. přibližně $\langle \mathbf{-9.10^{18}, 9.10^{18}} \rangle$

Racionální čísla

- Formát dle IEE 754
- Součásti:
  - Znaménko
  - Mantisa (přímý kód, normalizace, **m** bitů)
  - Exponent (v kódu posunuté nuly, **n** bitů)

Racionální čísla

- Normalizace mantisy (exponent má $\mathbf{n}$ bitů):
  - Nejvyšší bit vždy jedna: $\mathbf{1.aaaaaa}$; $\mathbf{1}$ nezapisujeme
  - Nejmenší kladné číslo: $\mathbf{1.0 \times 2^{-2^{n-1}+1}} (\mathbf{2^{-127}}$ pro $\mathbf{n = 8})$
  - Největší číslo: $\mathbf{1.0 \times 2^{2^{n-1}}} (\mathbf{2^{128}})$
- Exponent ($\mathbf{n}$ bitů, dvojková soustava)
  - Přičteme $\mathbf{2^{n-1} - 1}$ ($=127$ pro $\mathbf{n = 8}$), abychom získali správnou hodnotu pro uložení
  - $\mathbf{000000000}$ je $\mathbf{-127}$
  - $\mathbf{111111111}$ je $\mathbf{128}$
- Zvláštní a nenormalizovaná čísla

Racionální čísla II

- Rozsah zobrazení: ⟨Největší záporné, Největší kladné⟩
- Přesnost zobrazení: počet bitů mantisy+**1**
- Rozlišitelnost: nejmenší nenulové číslo
  - Normalizované vs. nenormalizované ($\mathbf{2^m}$krát menší, **m** počet bitů mantisy)

Jiné soustavy

- Osmičková
  - $\mathbf{001\,101\,101\,111_2 = 1557_8 = 879_{10}}$
- Šestnáctková
  - $\mathbf{0011\,0110\,1111_2 = 36F_{16} = 879_{10}}$
- Používány především pro „hutný“ zápis binárních čísel

Operační systémy – trocha historie

- Bootstrap loader
- Spooling
  - Nezávislé zavádění programu a jeho vykonávání
  - Vyžaduje DMA (Direct Memory Access)
  - Zavedlo *multiprogramování*
  - Stále zpracování *dávek* (batch processing)
- Timesharing
  - Virtualizace počítače/CPU
  - Zpracování *interaktivních* úloh
  - Souvisí se zavedení *disků* (Direct Access Storage Device, DASD od IBM, 60tá léta)

Operační systémy: účel

- *Zkrásnění*:
  - Zjednodušení práce s počítačem
    - Práce s pamětí
    - Práce se soubory
    - Přístup k periferiím

Operační systémy: účel

Sdílení:

- Zajistit sdílení vzácných zdrojů
- Musí zajistit:
  - Aby to vůbec fungovalo
  - Aby to fungovalo účinně (využití, propustnost, rychlost odezvy)
  - Aby to fungovalo správně
    - Omezení následků chyb (avšak pozor na chyby v samotném operačním systému)
    - Oprávnění k přístupu (autentizace a autorizace)

OS: problém časování

- Periferie výrazně pomalejší než procesor
- Příklad
  - 1 GHz Pentium IV: $\mathbf{1.10^9}$ operací za sekundu
  - Běžný disk: 10 ms pro přečtení 1 byte
  - Poměr $\mathbf{1 : 10\,000\,000}$
  - Stejné zpomalení člověka: 1 úhoz na klávesnici cca 20 dní.
- Možné řešení: prokládání I/O a výpočtu
  - `Spusť diskovou operaci`
    `Prováděj instrukce nad jinými daty (alespoň 1 M instrukcí)`
    `Počkej na dokončení`
  - Příliš těžkopádné a složité

OS časování: jiné řešení

```
Proces 1 {
  Spusť diskovou operaci
  Počkej na dokončení
  Zpracuj získaná data
}
Proces 2 {
  Nějaká jiná aplikace
}
```

- Přehlednější
- OS musí „přepínat“ mezi procesy (*priorita*)

OS: paměť

- Většina paměti nevyužita
  - Zpracování cyklu (zbytek programu)
  - Zpracování konkrétních dat (ostatní neaktivní)
  - Čekání na I/O
- *Virtualizace* paměti
  - Data a programy na disku
  - Do paměti *na žádost*
  - Umožňuje
    - Každý program má „celou“ paměť
    - Program může adresovat více jak rozsah fyzické paměti
- Ochrana paměti

OS: základní složky

- Procesy a jejich správa
- Paměť a její správa
- Periferie a jejich správa
- Systém souborů
- Ochrana a bezpečnost

Procesy

- Proces je abstrakce průchodu programem
  - Sekvenční model: program = 1 proces
  - Paralelní model: program > 1 proces
- Proces má *interní stav*, charakterizovaný
  - programovým čitačem (program counter)
  - zásobníkem (volání funkcí a procedur)
  - vlastní pamětí pro data

Typy procesů

- Klasické (heavy-weight) procesy (např. UNIX)
  - Všechna data privátní
  - Sdílen pouze program (read-only)
- *Lehké* (light-weight) procesy či Vlákna (threads)
  - Minimum vlastní paměti
  - Většina dat sdílena

Procesy detailněji

- Vytvoření procesu
  - `fork()` a jeho varianty
  - *Přesná* kopie původního procesu
  - *Rodič* a *potomek*
  - První proces v OS vytvářen jinak (`init` v Unixu)
- Stavy
  - Start/vytvoření, připraven (ready), běží (running), je blokován (čeká), skončil

Synchronizace – problém

- Race condition: soupeření v čase
  - ```
    Proces P {
        Load RegistrA, X
        Load RegistrB, Y
        Add RegistrA, RegistrB
        Store RegistrA, X   # X+=Y
    }
    ```
- Dvě instance procesu P, používají stejná X a Y
- Nedefinovatelné výsledky
  - Je-li na začátku X=Y=1, pak na konci může být X=2 nebo X=3
    ```

Synchronizace – řešení

- Kritická sekce
  - Semafory: celočíselné proměnné (čitače)
  - Monitory: vyšší konstrukty programovacího jazyka
    Je možné semafor implementovat pomocí monitoru a naopak
- Smrtelné objetí (deadlock)
- Odstranění sdílených zdrojů: zasílání zpráv
  - Synchronizace na úrovni zasílání a přijímání zpráv
  - Buffery

Procesy – plánování

- Sdílení (timesharing)
  - časové kvantum
  - přerušení
- Prioritní
  - Statistické
  - Real-time
- Plánovač (scheduler)

Správa paměti

- Dvě základní operace:
  - alokuj/přiděl paměť (velikost, vrací počáteční adresu)
  - dealokuj/uvolni paměť (velikost a počáteční adresa)
  - Většinou závislé (lze uvolnit jen přesně totéž, co jsme alokovali dříve)
  - Doplňková operace: změň rozsah alokované paměti (reallocate)
- Organizace paměti
- Čištění paměti (garbage collection)

Správa paměti OS

- Virtualizace paměti – nutno uvolnit fyzickou paměť
- Swapping
  - Celých procesů
  - „Děr“ v paměti
- Stránkování
- Segmentace