FAKULTA
INFORMATIKY
Masarykova univerzita

PB001: Úvod do informačních technologií

Luděk Matyska (Eva Hladká)

podzim 2024

Obsah přednášky

Číselné soustavy

Operační systémy

- Historie
- Účel
- Základní složky
  - Procesy

Procesy – synchronizace a plánování

- Paměť
- Systém souborů

Číselné soustavy

- Dle základu: dvojková, osmičková, šestnáctková, desítková
- Volně mezi sebou převoditelné (celá čísla bez ztráty přesnosti)
- Celá čísla a zlomky
- Reálná čísla
- První počítač v desítkové soustavě
- Konečné reprezentace

FAKULTA
INFORMATIKY
Masarykova univerzita

Dvojková soustava

- Základem je číslo dvě
  - pouze dvě číslice – dva stavy
  - vhodná pro reprezentaci v číslicových sytémech

Dvojková soustava v počítači

- Celá čísla
- Záporná čísla
  - Přímý kód
  - Inverzní kód (binární negace)
  - Dvojkový doplňkový kód (binární negace + 1)
- Pro reálná čísla
  - Rozlišitelnost (nejmenší zobrazitelné číslo): $X + \epsilon > X$ a $X + \epsilon/2 = X$
  - Přesnost (rozsah)
  - Zobrazení: mantisa m a exponent e
    $0 \leq m \leq 1 \wedge x = m.2^e$

Záporná čísla – zobrazení

- Přímý kód
  - přidáno znaménko – znaménkový bit
  - Dvě nuly: $+\mathbf{0}$ a $-\mathbf{0}$ (**10 . . . 00**)
  - Rozsah: $\langle -\mathrm{MAX}, -\mathbf{0}\rangle$ a $\langle +\mathbf{0}, +\mathrm{MAX}\rangle$
- Inverzní kód:
  - Přidáme znaménko
  - Dvě nuly: $+\mathbf{0}$ a $-\mathbf{0}$ (**11 . . . 11**)
  - Rozsah: $\langle -\mathrm{MAX}, -\mathbf{0}\rangle$ a $\langle +\mathbf{0}, +\mathrm{MAX}\rangle$

Záporná čísla – zobrazení

- Dvojkový doplňkový kód
  - Inverze bitu a přičtení jedničky
  - Pouze jedna nula (**11 . . . 11** je $-$**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. $\langle \mathbf{0}, \mathbf{255} \rangle$ nebo $\langle -\mathbf{128}, \mathbf{127} \rangle$
  - Půl slovo, 2 byte: 16 bitů, tj. $\langle \mathbf{0}, \mathbf{65535} \rangle$ nebo $\langle -\mathbf{32768}, \mathbf{32767} \rangle$
  - Slovo, 4 byte: 32 bitů, tj. přibližně $\langle -\mathbf{2.10^9}, \mathbf{2.10^9} \rangle$
  - Dvojslovo (nebo dlouhé slovo), 8 byte: 64 bitů, tj. přibližně $\langle -\mathbf{9.10^{18}}, \mathbf{9.10^{18}} \rangle$
  - Dnes již i „čtyřslovo”, 16 byte: 128 bitů, tj. přibližně $\langle -\mathbf{1,7.10^{38}}, \mathbf{1,7.10^{38}} \rangle$

Racionální čísla

- Formát dle IEEE 754
- Součásti
  - Znaménko
  - Mantisa (přímý kód, normalizace)
  - Exponent (v kódu posunuté nuly)

Racionální čísla

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

Jiné soustavy

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

FAKULTA INFORMATIKY
Masarykova univerzita

OS jako příklad

- Operační systém je velmi komplikovaný programový produkt
- Vývoj odráží změny v informačních technologiích
  - programovací jazyky
  - softwarové inženýrství
  - vývoj hardware (kvalita, kapacita, ...)
  - vývoj periferií
- (Prakticky) každý se s ním potká
- Principy návrhu proto demonstrovány na operačních systémech a jejich komponentách

FAKULTA INFORMATIKY
Masarykova univerzita

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)

FAKULTA
INFORMATIKY
Masarykova univerzita

Operační systémy: účel

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

FAKULTA INFORMATIKY
Masarykova univerzita

Operační systémy: účel

Sdílení a správa:

- Zajistit sdílení (a správu) 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)

FAKULTA
INFORMATIKY
Masarykova univerzita

OS: základní složky

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

FAKULTA INFORMATIKY
Masarykova univerzita

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

FAKULTA INFORMATIKY
Masarykova univerzita

Typy procesů

- Klasické (heawy-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

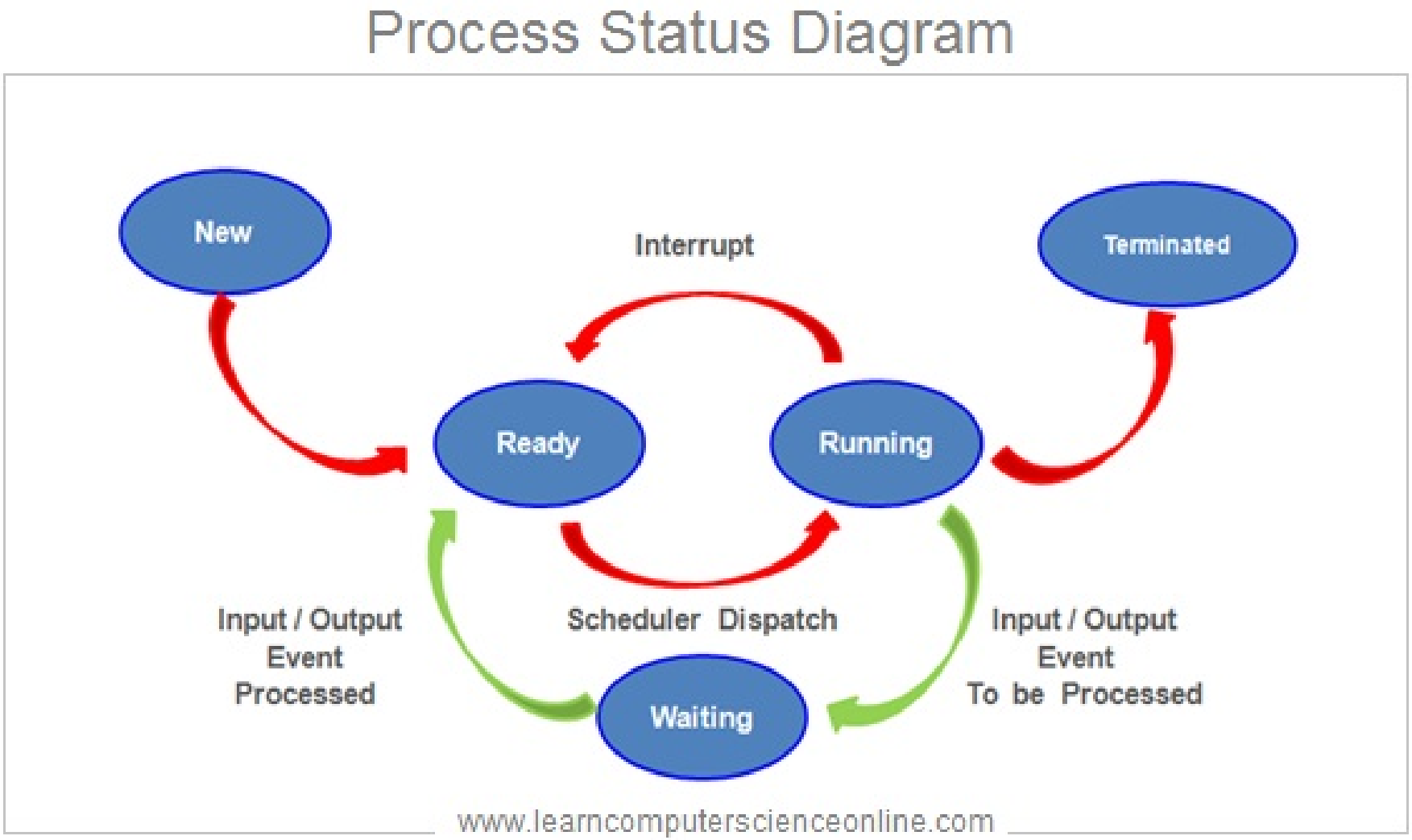
FAKULTA INFORMATIKY
Masarykova univerzita

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

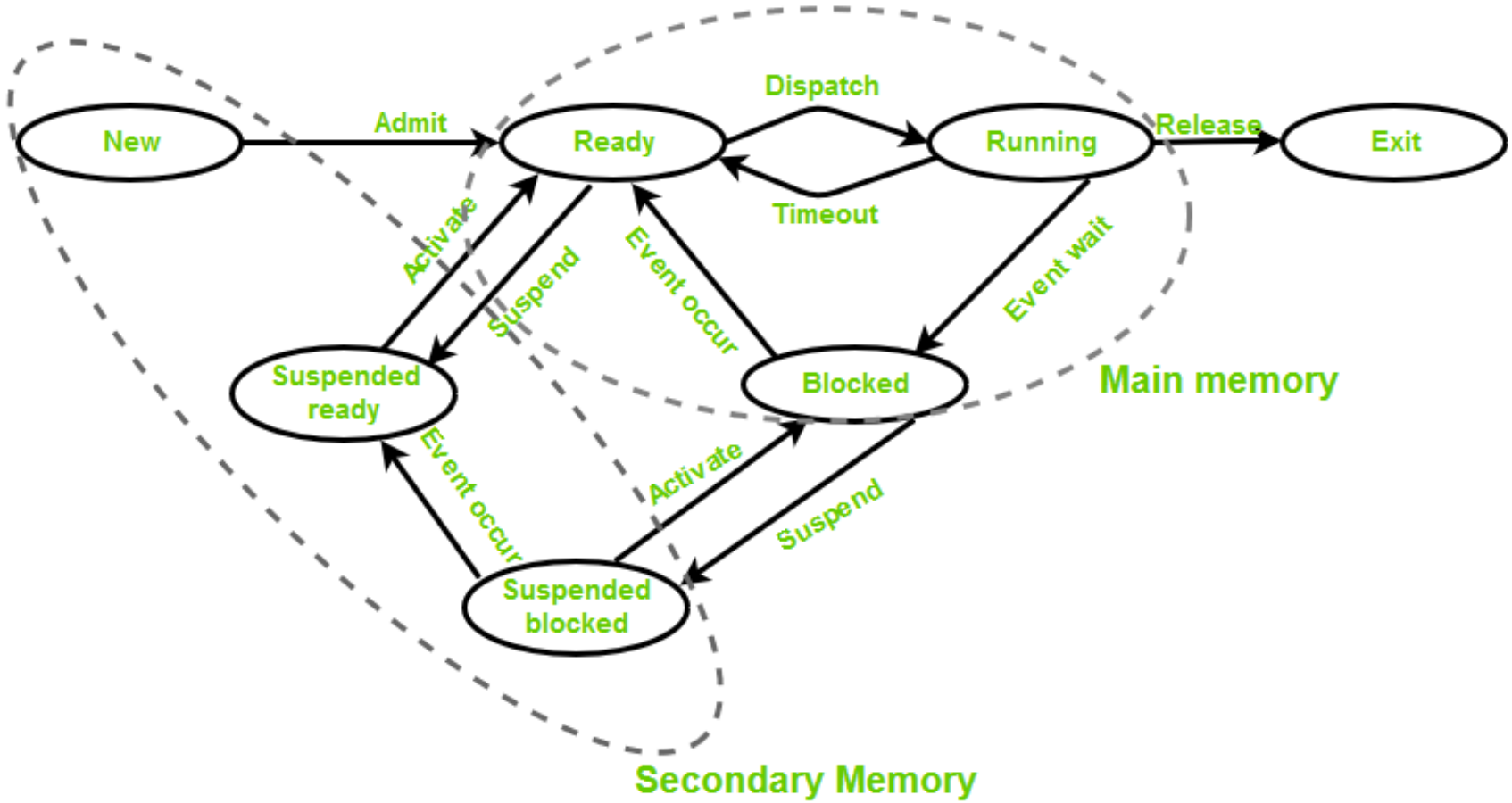
FAKULTA INFORMATIKY
Masarykova univerzita

OS: Stavy procesu



FAKULTA INFORMATIKY
Masarykova univerzita

OS: Stavy procesu (s externí pamětí)



FAKULTA INFORMATIKY
Masarykova univerzita

OS: problém časování

- Periferie výrazně pomalejší než procesor
- Příklad
  - 1 GHz Pentium IV: **1.10^9** operací za sekundu
  - Běžný disk: 10 ms pro přečtení 1 byte
  - Poměr **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˜milion instrukcí)
    Počkej na dokončení
    ```
  - Příliš těžkopádné a složité
    ```

FAKULTA INFORMATIKY
Masarykova univerzita

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*)

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)

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

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

OS: Systém souborů

- Základní funkce:
  - Vytvoření souboru
  - Čtení a psaní z/do souboru
  - Odstranění (smazání) souboru
  - Spuštění souboru (soubor=program)
- Podpora na úrovni operačního systému

Struktura systému souborů

- Hierarchické systémy:
  - Kořen (root)
  - Adresáře jako speciální typ (meta)souboru: drží informace o souborech, nikoliv jejich vlastní data
- Databázové systémy:
  - Soubory (jejich části resp. jejich metadata) jako položka v databázi
  - Bohatší množina operací
  - Složitější implementace

  Apache Hadoop má prvky databázového systému souborů
  - Hadoop Distributed File System
  - Soubory jsou rozděleny na části, které jsou distribuovány na prvky clusteru
  - Primární přístup k souborům přes nativní rozhraní HDFS

Struktura souborů

- Posloupnost bytů – vnitřní struktura pro OS neznáma
- Posloupnost záznamů (records)
- Strom – každý uzel má vlastní klíč

- Výše uvedeny **příklady** struktury, ne všechny varianty

Typ a přístup

- Typy souborů (v UNIXovém OS)
  - Řádné: běžné soubory
  - Adresáře: udržení hierarchické struktury
  - Speciální: přístup ke konkrétnímu zařízení (`/dev/mouse`, `/dev/audio`, `/dev/lp`); speciální `/proc` systém
  - Blokové: náhodný přístup na základní úrovni (`/dev/hd`, `/dev/kmem`)
- Přístupové metody; příklady:
  - Sekvenční
  - Náhodný (random)
  - Indexsekvenční (není v běžném UNIXu)

Struktura na disku

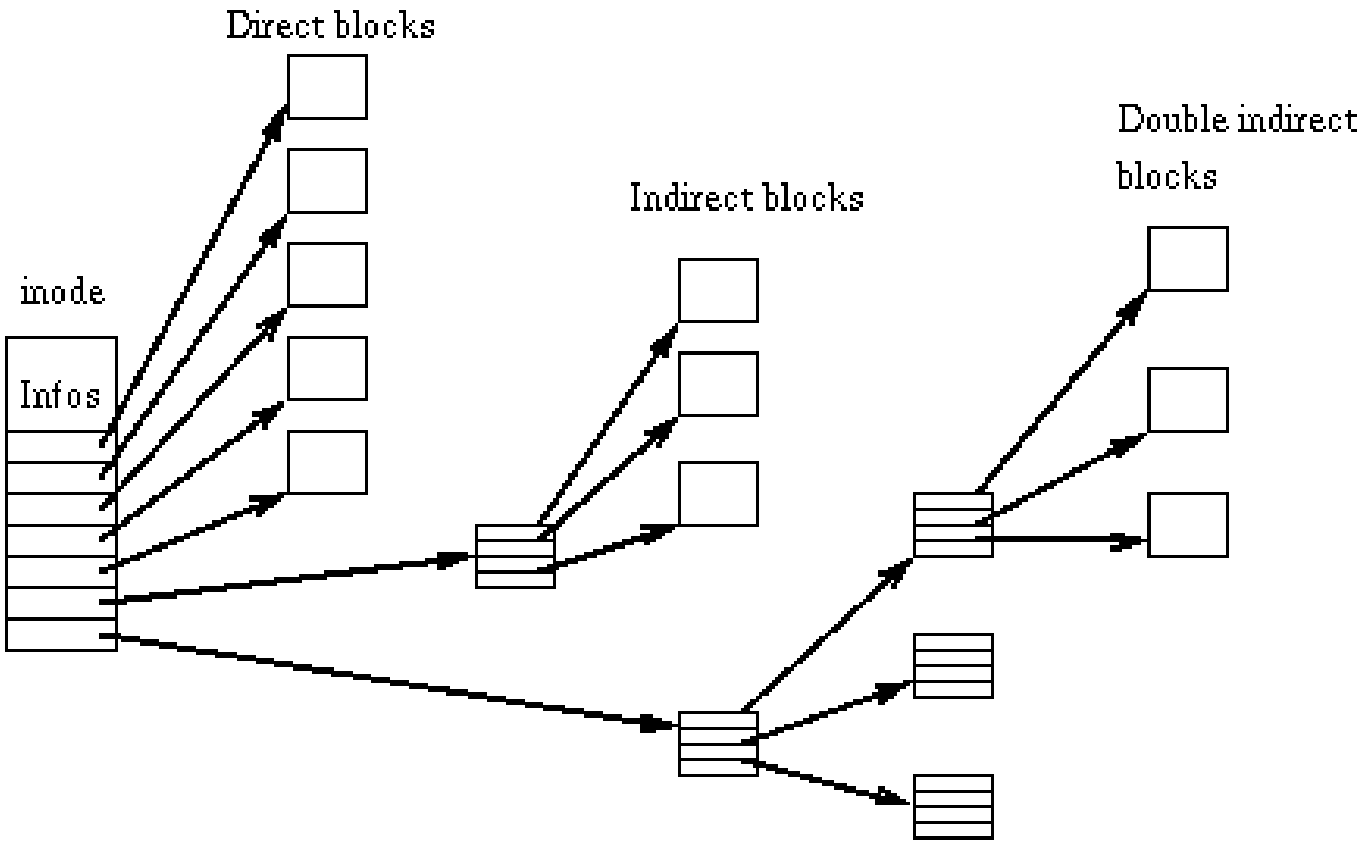
- Možné typy
  - Souvislé
    - souvislé posloupnost bloků (složitá alokace, plýtvání místem)
  - Provázaný seznam:
    - každý blok odkazuje na další (může růst, vyšší režie – pro ukazatel, složitý náhodný přístup)
  - Indexové:
    - Např. FAT (File Allocation Table) v MS DOSu
    - Tabulka pro všechny bloku na disku
    - Provázány odkazem na další blok daného souboru
  - `inodes`

Struktura – inodes

- Podobné indexové organizaci
- Pevná délka tabulky pro každý soubor
  - Kratší soubory adresovány přímo
  - Pro delší soubory alokována další tabulka
  - Tabulky provázány hierarchicky (1., 2. a 3. úroveň)
- Flexibilní, malá režie

FAKULTA INFORMATIKY
Masarykova univerzita

inodes



Ext2-inodes

Systémy souborů se žurnálováním

- Problém interní konzistence informací uvnitř systému souborů
  - co se stane při výpadku proudu či nečekaném zhroucení operačního systému
- Riziko nekonzistentních dat při postupném zápisu
  - část dat nebo metadat není ještě zapsána
- Klasické řešení: **fsck**
  - procházení všech datových struktur v systému souborů
  - nalezení a oprava nekonzistencí (zpravidla několik průchodů)
  - velmi pomalé pro velké systémy souborů
- Alternativa: zapsat zvlášť prováděné změny a teprve poté je skutečně realizovat
  - úspěšný zápis všech dat a metadat vede ke smazání údajů

  journaled file system
- V případě výpadku se použijí tato data na zajištění interní konzistence systému souborů

Volné bloky

- V tabulce
- Bitový vektor
- Provázaný seznam
- Většinou zpracovávány podle FCFS (First Come First Served)

Vyrovnávací paměť

- Obecně přístup pro skrytí *zpožděnı´* (latence)
- Nejčastěji používané bloky/soubory uloženy v paměti
- Pouze pro čtení (snazší) nebo i pro zápis
- Problém: konzistence při přístupech/zápisech z více míst
- Základní typy
  - Write-through: okamžitě po zápisu i na disk
  - Write-back: až po určité době (30 s)