

## Distribuované algoritmy

### PA150 ◊ Principy operačních systémů

Jan Staudek

<http://www.fi.muni.cz/usr/staudek/vyuka/>



Verze : podzim 2020

## Distribuovaný systém, distribuovaný algoritmus

- **Distribuovaný systém, DS**
  - ✓ množina **autonomních** výpočetních komponent (jednotek, procesorů, zařízení, . . . **vzájemně propojených** nějakou komunikační strukturou)
- **Distribuovaný algoritmus, DA**
  - ✓ agregace algoritmů běžících v jednotlivých **komponentách DS**
- **Připomenutí pojmu algoritmus**
  - ✓ přesný **návod či postup**, kterým lze vyřešit daný typ úlohy
  - ✓ teoretický princip řešení jisté třídy obdobných problémů
  - ✓ skládá se z konečného počtu jednoduchých (elementárních), jednoznačně a přesně definovaných kroků
  - ✓ končí, poskytuje výsledek, v (libovolně velkém) konečném počtu kroků

## Složitost DA

- Jeden a tentýž problém lze obvykle řešit více různými způsoby a je žádoucí mít možnost měřit efektivitu konkrétních řešení z hlediska potřebných zdrojů pro jejich realizaci podle vhodných kritérií
- Určitě mají význam sekvenční (lokální) doby výpočtů a (lokální) nároky na paměť
- Pro DA jsou však charakterističtější míry vyjadřující
  - ✓ **komunikační složitost**
  - ✓ **globální nároky na paměť** v celém DS
- Tyto míry jsou založené na podobných procedurách pro výpočet složitosti sekvenčních systémů a jsou poměrně široce standardizované

## Složitost DA

- **Komunikační složitost, C**
  - ✓ *Message Complexity*, **složitost výměny zpráv**
  - ✓ *Bit Complexity*, **bitová složitost**
- **Časová složitost, T**
  - ✓ **Doba vnitřního, lokálního zpracování**
  - ✓ **Zpoždění dané přenosy zpráv**
- **Paměťová složitost, M**
  - ✓ Množství paměti potřebné pro realizaci výpočtu DA
  - ✓ **Totální paměťové nároky** v rámci celého DS
  - ✓ **Maximální paměťové nároky** v jedné komponentě DS
  - ✓ **Redundance a vyváženost** v komponentách DS

## Komunikační složitost DA

- *Message Complexity*, složitost výměny zpráv  
Celkový počet zpráv vyměněných distribuovaným algoritmem
- *Bit Complexity*, bitová složitost  
Počet bitů (množství informace) přenášených DA
- Při výkladu používáme složitost výměny zpráv, poněvadž v našich DA je komunikace vyjadřovaná v pojmech abstraktních zpráv vyměňovaných mezi komponentami
- Velikost této míry je typicky daná počtem komunikačních událostí (vyslání/příjem zprávy)
  - ✓ Každá zpráva může přenášet jakýkoliv datový prvek, tj. paměťové kapacity komunikačních kanálů mohou být libovolně velké

## Časová složitost DA

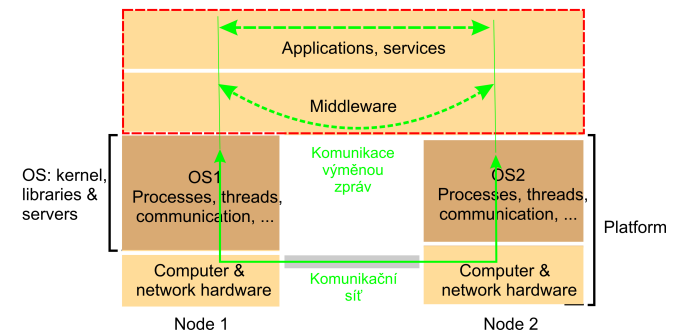
- Kolik časových jednotek uplyne od startu do ukončení DA
- Dobu vnitřního, lokálního zpracování zanedbáváme
- Časová složitost při synchronním plánování
  - ✓ časová složitost je daná počtem synchronních běhů do ukončení DA
- Časová složitost při asynchronním plánování
  - ✓ Používají se idealizované normalizační předpoklady:
    - čas vnitřního, lokálního zpracování je zanedbatelný, každá komponenta může provést libovolný konečný výpočet v nulové čase
    - maximální přenosové zpoždění trvá nejvýše 1 časovou jednotku
  - ✓ Časová složitost je daná celkovým normalizovaným časem provedení

## Paměťová složitost DA

- Totální paměťové nároky v rámci celého DS
- Maximální paměťové nároky v jedné komponentě DS
- Redundance
  - ✓ počet kopií datového elementu udržovaného v DS
  - ✓ DS je kompaktní, pokud je minimální i maximální redundance všech datových elementů je shodná
- Vyváženost
  - ✓ DA má vyvážené požadavky na paměť, pokud je maximální paměťová složitost komponenty DS úměrná poměru počtu datových elementů DA ( $m$ ) a počtu komponent DS ( $n$ )
  - ✓ DA má perfektně vyvážené požadavky na paměť, pokud je maximální paměťová složitost komponenty =  $m/n$ , příp.  $m/n + 1$

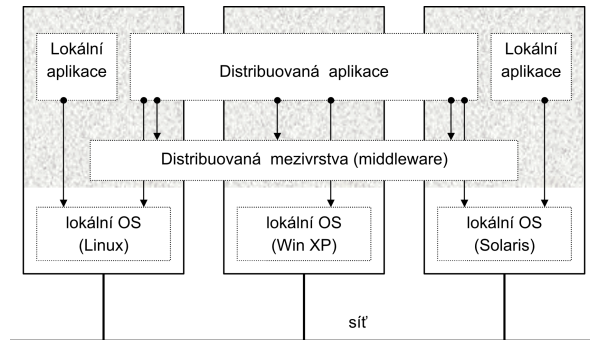
## Distribuované prostředí přednášky

- libovolný lokální OS + rozšiřující vrstva poskytující distribuované prostředí + další služby (transakce, ...)
- prostředí pro distribuované aplikace – middleware (OSF DCE, CORBA, DCOM, Globe, ...)



## Distribuované prostředí přednášky

- libovolný lokální OS + rozšiřující vrstva poskytující distribuované prostředí + další služby (transakce, ...)
- prostředí pro distribuované aplikace – middleware (OSF DCE, CORBA, DCOM, Globe, ...)



## Předpoklady pro model použitý při našem studiu DA

- Komunikační síť je silně souvislá
- Procesy komunikují pouze výměnou zpráv
- Komunikace výměnou zpráv je pseudo-asynchronní, zdržení zprávy v kanálu může být libovolné, vždy je konečné, výpadky komunikačního přenosu lze detekovat hlídáním časových limitů
- Komunikační kanály zprávy neztrácejí, neduplikují, nemodifikují
- Přenos zpráv komunikačními kanály se řídí politikou FIFO
- Procesy nepadají
- Procesy znají pouze své sousedy, nikoli topologii celého DS
- Procesy mají jedinečné identifikátory (pid)

## Konfigurace, přechod, provedení

- Globální stav DS daný **stavem jeho procesů** a **zprávami obsaženými v jeho kanálech** je **konfigurací** DS
- Konfigurace vzniká postupně, po krocích zvaných **přechody**
- **Systém přechodů** sestává z
  - ✓ množiny **konfigurací**  $C$
  - ✓ binární relace **přechodu**  $\rightarrow$  na  $C$
  - ✓ množiny **iniciálních konfigurací**  $I \subseteq C$
- Konfigurace  $\gamma \in C$  je **terminální**, pokud neexistuje  $\gamma \rightarrow \delta$  pro žádnou z  $\delta \in C$
- **Provedení** algoritmu je posloupností konfigurací  $\gamma_0\gamma_1\gamma_2\dots$ , kde  $\gamma_0 \in I$  a  $\gamma_i \rightarrow \gamma_{i+1}$  pro všechna  $i \geq 0$
- Konfigurace  $\delta$  je **dosazitelná** pokud  $\gamma_0\gamma_1\gamma_2\dots\gamma_k = \delta$ , kde  $\gamma_0 \in I$  a  $\gamma_i \rightarrow \gamma_{i+1}$  pro všechna  $0 \leq i < k$

## Události

- Každý přechod v DS je vázaný na jistou **událost** v některém z procesů DS
  - ✓ V případě synchronních DS na dvě události ve dvou procesech DS
- Proces může generovat **vnitřní** událost a událost **vyslání** zprávy
- V procesu může nastat událost **příjmu** zprávy
- Proces je **iniciátor**, pokud jeho první událostí je jeho vnitřní událost nebo událost vyslání zprávy
- DA je **centralizovaný**, pokud existuje právě jeden iniciátor
- **Decentralizovaný** DA může mít více iniciátorů

## Požadované vlastnosti distribuovaných algoritmů

- **Bezpečnost**, *Safety*, **Nothing bad happened yet**
  - ✓ Sledovaná podmínka: **Globální stav DS je ve konfiguraci, ze které je normálními stavovými přechody nedosažitelný jistý, konkrétní nežádoucí stav**
  - ✓ Cíl – např. dosáhne se vzájemné vyloučení kritických sekcí procesů, zabrání se uváznutí, . . .
  - ✓ Typicky se dokazuje indukcí: **jestliže  $X$  platí pro  $n = 1$  a jestliže  $X$  platí pro  $n = m$  a pro  $n = m + 1$ , pak  $X$  platí pro všechna  $n$**
  - ✓ Narušení bezpečnosti (tj. narušení dosažitelnosti cíle algoritmu) se prokazuje v konečném počtu kroků řešení
  - ✓ Řešení problému nenarušující bezpečnost je **korektní řešení**
  - ✓ Podmínka bezpečnosti musí být splněná v každé konfiguraci každého provedení algoritmu, je **invariantem**.  
Předpoklad  **$P$**  je **invariantem**, pokud platí  $P(\gamma)$  pro všechny  $\gamma \in I$  a jestliže existuje  $\gamma \rightarrow \delta$ , pak platí i  $P(\delta)$ .

## Požadované vlastnosti distribuovaných algoritmů

- **Živost**, *Liveness*, **Something good eventually happens**
  - ✓ Vlastnost globálního stavu DS zajišťující, že **jistou posloupností normálních stavových přechodů je dosažitelný jistý, konkrétní žádoucí stav**
  - ✓ Např. v konečném počtu kroků algoritmu se zvolí vedoucí uzel v síti nebo proces žádající o vstup do kritické cesty získá právo vstoupit do kritické sekce
  - ✓ Narušení podmínky živosti se prokazuje pouze v nekonečném počtu kroků řešení
  - ✓ Korektní řešení problému nenarušující živost je **úplné, kompletní řešení**
  - ✓ Podmínka živosti musí být splněná v některé konfiguraci každého provedení algoritmu

## Typové synchronizační úlohy v distribuovaném prostředí

- Zjištění globálního stavu distribuovaného systému
  - ✓ viz přednáška  
*Distribuované prostředí, čas, stav, distribuované algoritmy*
- Vzájemné vyloučení kritických sekcí
  - ✓ kritická sekce – pasáž běhu procesu (vlákna) operující se zdrojem v čase přístupným jedinému procesu (vláknou)
- Volební problém – volba řídicího (*master*) uzlu, volba „lídra“
  - ✓ v množině kooperujících procesů (vláken) smí mít jediný proces (jediné vlákno) statut řídicího prvku (*master*)
  - ✓ řídicím prvkem může být kterýkoliv z kooperujících procesů (vláken)
  - ✓ řídicí proces bude plnit roli serveru, . . .

## Typové synchronizační úlohy v distribuovaném prostředí

- Řešení problému sdělování zpráv všem procesům náležejících do skupiny procesů, multicasting
  - ✓ se zárukou doručení zprávy všem procesům skupiny
  - ✓ se zárukou doručování zpráv v definovaném pořadí (FIFO, . . .)
- Problém dosažení shody
  - ✓ všechny (nechybující) procesy z množiny procesů poskytujících jistou službu musí deklarovat shodnou výstupní hodnotu
  - ✓ **Příklad:**  
Vesmírná mise je řízena  $x$  počítači paralelně aby se zajistila spolehlivost řízení i v případech, kdy může dojít k selhání až  $y$  počítačů  
Řízení mise musí dospět k rozhodnutí:  
*pokračovat v misi / návrat z mise*  
na základě doporučení většiny neselehavších počítačů.  
Jak velké musí být  $x$ , pokud lze kvalifikovaně odhadnout  $y$  ?