PA160

Synchronizace času

Čas na jednom uzlu

- Hodiny resp. *časovač* (timer)
  - Oscilující krystal křišťálu
  - Čítač
    - Každá oscilace sníží hodnotu o jedna
    - Přerušení při hodnotě nula

    Každé přerušení je *tik*
  - Holding register
    - Iniciace čítače po přerušení
  - Po přerušení zvýšen uložený čas

Vícenásobní procesory

- Každý uzel má vlastní časovač
  - Čas není automaticky synchronizován
  - Časové pokřivení (clock skew)
- Vazba na absolutní reálný čas
- Problémy
  - Vzájemná synchronizace uzlů
  - Synchronizace s reálným časem

Absolutní čas

- Původní definice časové jednotky
  - 1 sekunda je 1/86 400 slunečního dne
- Aktuální – atomové hodiny
  - Přechody Cesia 133
  - 1 sekunda je doba za niž proběhne 9 192 631 770 přechodů
- Mezinárodní atomový čas
  - Průměr měření cca 50 světových laboratoří

Universal Coordinated Time, UTC

- Atomový a sluneční čas se rozcházejí
- Skoková sekunda
  - Kompenzace rozchodu s rotací Země
  - Přidána když rozdíl mezi slunečním a atomových časem vzroste na 800 ms
- Výsledný čas je Univerzální koordinovaný čas (Universal Coordinated Time, UTC)
  - Nahradil GMT
- UTC dostupné celosvětově

Synchronizace hodin

- Základní východiska
  - Množina uzlů s vlastními hodinami
  - Přerušení H krát za sekundu
- $C_p(t)$ je čas měřený hodinami na stroji p
  - Ideálně $C_p(t) = t$ pro všechna p
  - Realita: Pokud existuje ρ takové, že platí

$$1 - \rho \leq \frac{dC}{dt} \leq 1 + \rho$$

    pak hodiny C_p pracují se specifikací ρ.
  - ρ definováno výrobcem (Maximální rychlost posunu času)
  - Chceme-li, aby se hodiny rozešly nejvýše o δ, musíme je synchronizovat nejméně každých $\delta/2\rho$ sekund

Cristianův algoritmus

- Máme *časový server*
  - Nejlépe napojený na absolutní čas
- Každých $\delta/2\rho$ posílá každý uzel dotaz serveru
- Server odpoví (jak nejrychleji může) s vlastním (UTC) časem
- Naivní řešení: Uzel změní svůj čas

Problémy

- Podstatný – reakce na zpoždění
  - Čas přestane mít lineární průběh
    - Nečekané a nežádoucí efekty
  - Řešení
    - Snížení absolutního času na přerušení
- Malý – doba komunikace
  - Změř čas mezi zasláním (T_0) a přijetím (T_1) požadavku
  - Přičti čas přenosu $(T_0 + T_1)/2$
  - Možno ještě započítat čas zpracování na serveru (je-li znám)

Berkeley algoritmus

- Aktivní časový server
  - Periodicky se dotazuje uzlů na jejich absolutní čas
  - Spočítá průměrný čas podle časů uzlů
  - Pošle všem tento nový čas
- Vhodné pokud sever nemá absolutní čas

Decentralizovaná řešení

- Resynchronizační intervaly
  - Globálně dohodnutý „počátek“ T_0
  - $i$tý interval začíná v čase $T_0 + iR$
  - $i$tý interval končí v čase $T_0 + (i+1)R$
  - R je dohodnutý systémový parametr
- Všichni pošlou broadcast se svým časem na začátku každého intervalu
- Zpracování na uzlu
  - Dojde S zpráv
  - Spočítá se průměr (s vyloučením m odlehlých hodnot)
  - Možné zlepšení při znalosti doby propagace zpráv

NTP protokol

- Network Time Protocol
  - NTP version 3 (RFC1305), version 2 (RFC1119), version 1 (RFC1059)
  - S(imple)NTP: RFC1769
- Hierarchická organizace
  - Stratum 1 přímo napojené na absolutní čas
  - Celkem až 16 úrovní (Stratum 16)
- Vysoce škálovatelné
- Více jak jeden server
  - Možná reakce na výpadek či ztrátu přesnosti
- Servery `tik.cesnet.cz` a `tak.cesnet.cz`

Logický čas

- Absolutní čas není vždy důležitý
- Podstatný *relativní* čas (souvislost dějů)
- *Logický čas*
  - Nevyžaduje absolutní synchronizaci
  - Potřeba shody na uspořádání

Lamportovy časové známky

- Relace „událo se dříve“ (happens-before)
- $a \longrightarrow b$ znamená, že se všechny procesy dohodly, že a nastalo dříve než b.
- Současně platí, že pokud a událost je zaslání konkrétní zprávy a b je událost přijetí téže zprávy, pak nutně $a \longrightarrow b$
- Vlastnosti
  - $a \longrightarrow b$ je tranzitivní
  - Události x a y jsou *souběžné* (concurrent), pokud neplatí ani $x \longrightarrow y$ ani $y \longrightarrow x$

Realizace

- Každý proces má vlastní logické hodiny
- Pro události uvnitř jednoho procesu je zajištění relace „událo se před“ triviální
- Synchronizace mezi procesy probíhá jako součást zasílání zpráv
  - Každá zpráva obsahuje časovou známku odesílatele (T_s
  - Pokud je čas přijímajícího (T_r) menší (mladší) než časová známka v přijímané zprávě, nastaví se $T_r = T_s + 1$
- Dodatečná podmínka
  - Žádné dvě události nenastanou ve stejný čas

Shrnutí

- Lamportův algoritmus umožňuje zajistit globální čas v distribuovaném systému
- Vlastnosti
  - Pokud a nastane před b ve stejném procesu, $C(a) < C(b)$
  - Pokud je a zaslání a b přijetí téže zprávy, $C(a) < C(b)$
  - Pro všechny různé události a a b platí $C(a) \neq C(b)$
- Algoritmus umožňuje globální uspořádání událostí v distribuovaném systému