Algoritmy

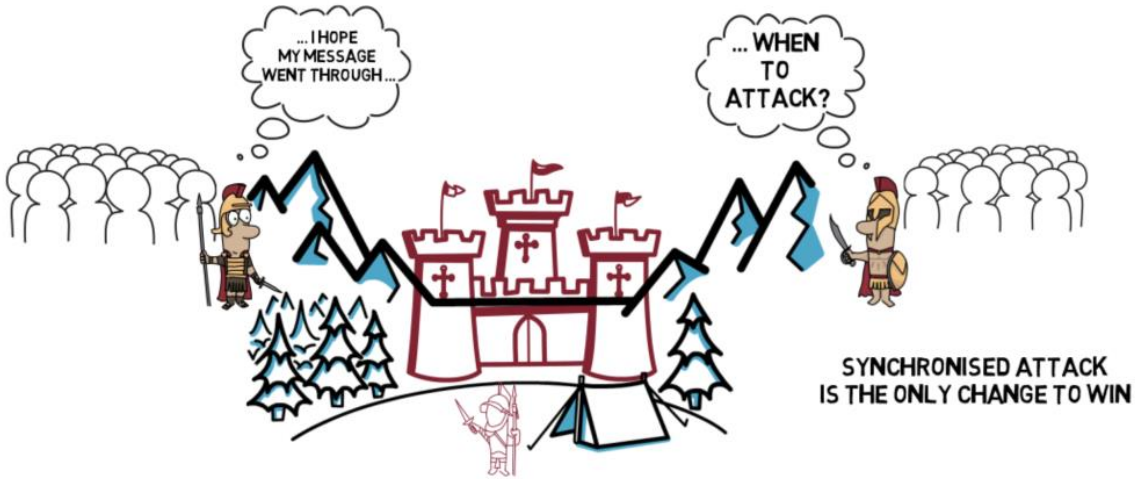
Ondřej Smola

04.04.2022 | CTC

ČÁST I.:
PROBLÉM 2 GENERÁLŮ

Problém 2 generálů

- Každý vede svoji armádu, pouze synchronizovaný útok má šanci uspět
- Komunikují pomocí zasílání zpráv s pomocí poslů
- Posel nemusí dorazit (např. zajat)
- Jak zaručit, že zaútočí ve stejný čas?



Snadné řešení ?

- Možné řešení:

1. Generál A: zaútočit za úsvitu!
2. Generál B: potvrzuji, zaútočit za úsvitu!
3. Generál A: potvrzuji, potvrzuji, zaútočit za úsvitu!
4. Generál B: potvrzuji, potvrzuji, potvrzuji, zaútočit za úsvitu!
5. …

Řešení problému dvou generálů za předpokladu nespolehlivého kanálu neexistuje!

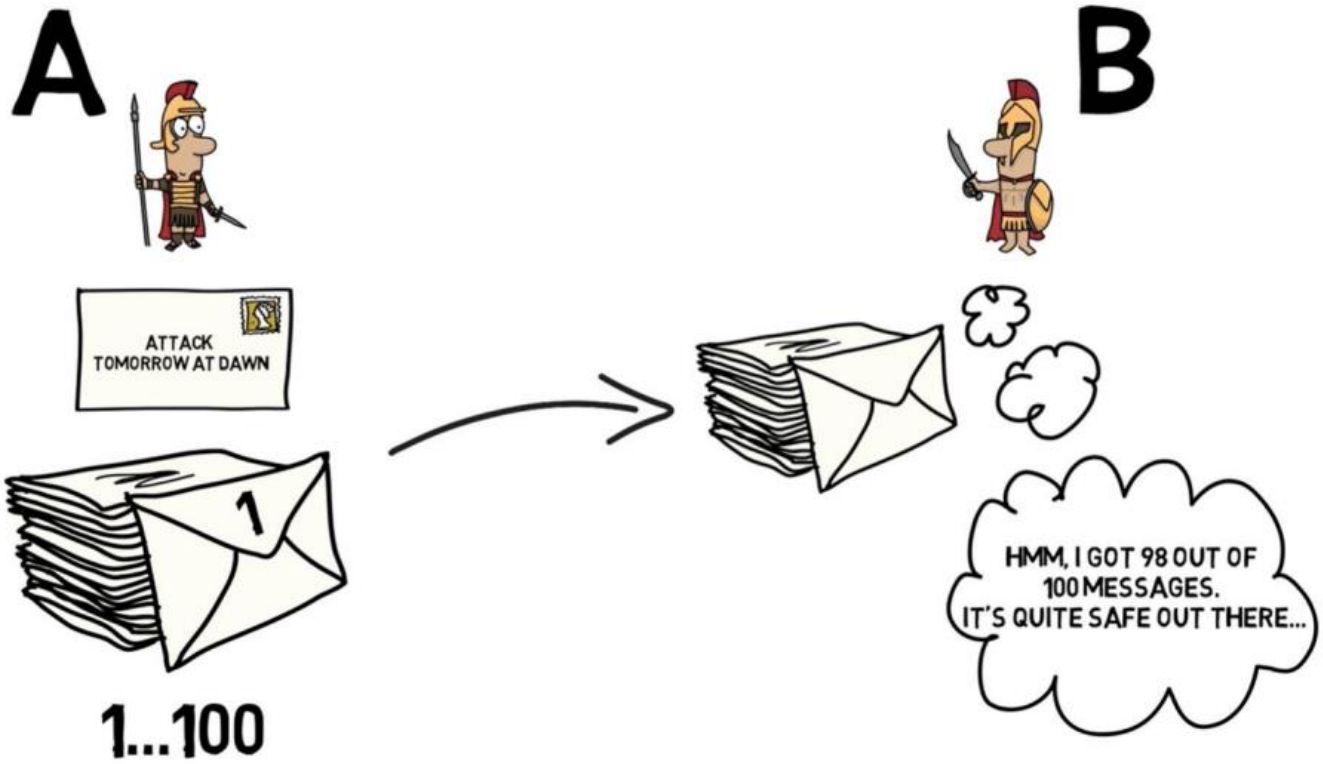
Pro zajímavost: Byzantinské chyby

Pragmatické řešení

- Musíme akceptovat nespolehlivost komunikačního kanálu a nesnažit se ji nejistotu eliminovat, ale snížit na přijatelnou mez
- Předpokládejme, že pravděpodobnost chycení kurýra je p a že opakované zachycení kurýra jsou nezávislé jevy.
- Možné řešení:
  1. Generál A pošle $\boldsymbol{n}$ kurýrů a každopádně zaútočí za úsvitu
  2. Generál B zaútočí za úsvitu pokud k němu dorazí aspoň jeden kurýr
- Pravděpodobnost nekoordinovaného útoku je $p\ n$.
- Snížit pravděpodobnost nekoordinovaného útoku můžeme posláním více kurýru … za cenu nutnosti poslat více kurýrů

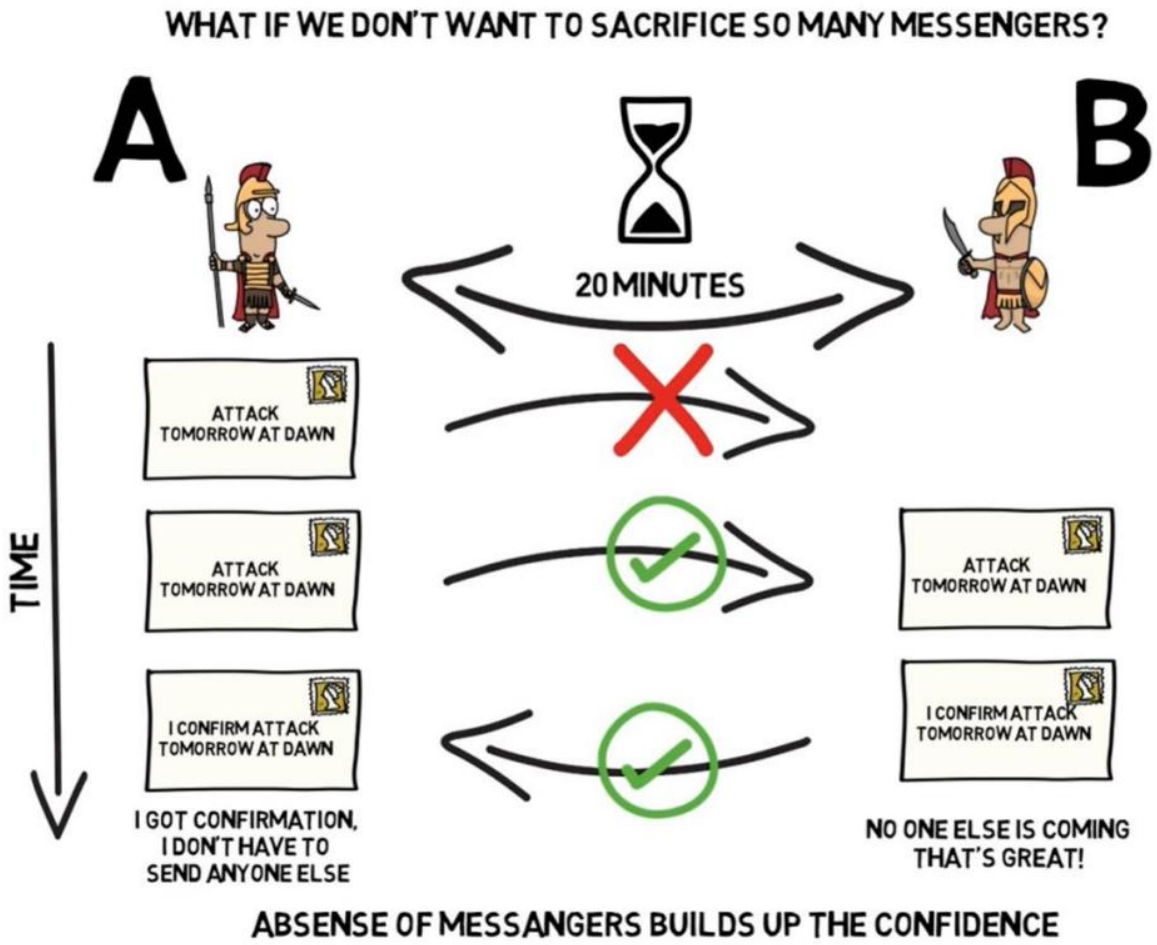
- **Jistoty koordinovaného útoku ale dosáhnout nemůžeme (pokud p > 0).**

Pragmatické řešení

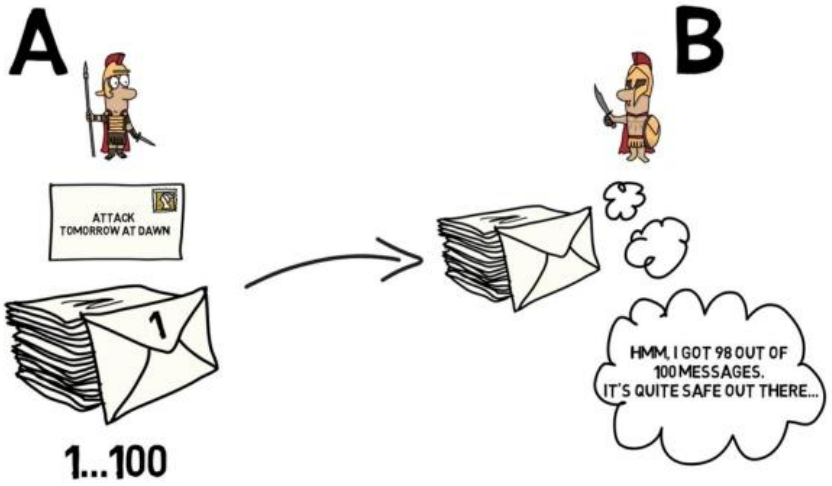


Drahé

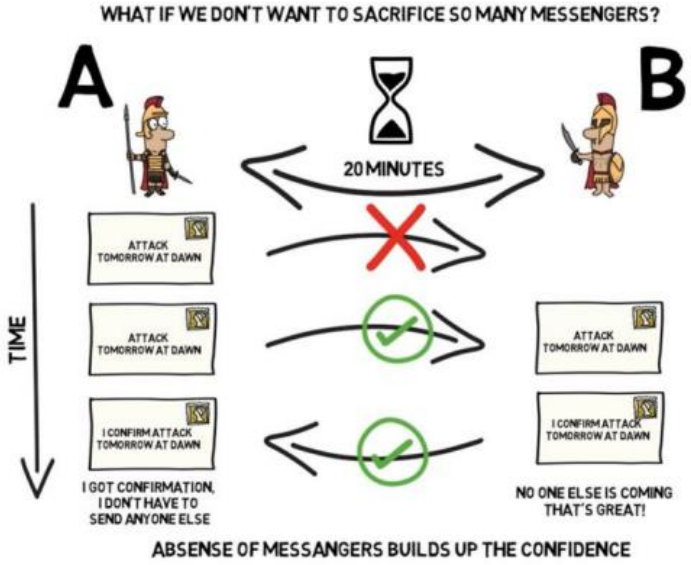
Alternativní řešení



Alternativy



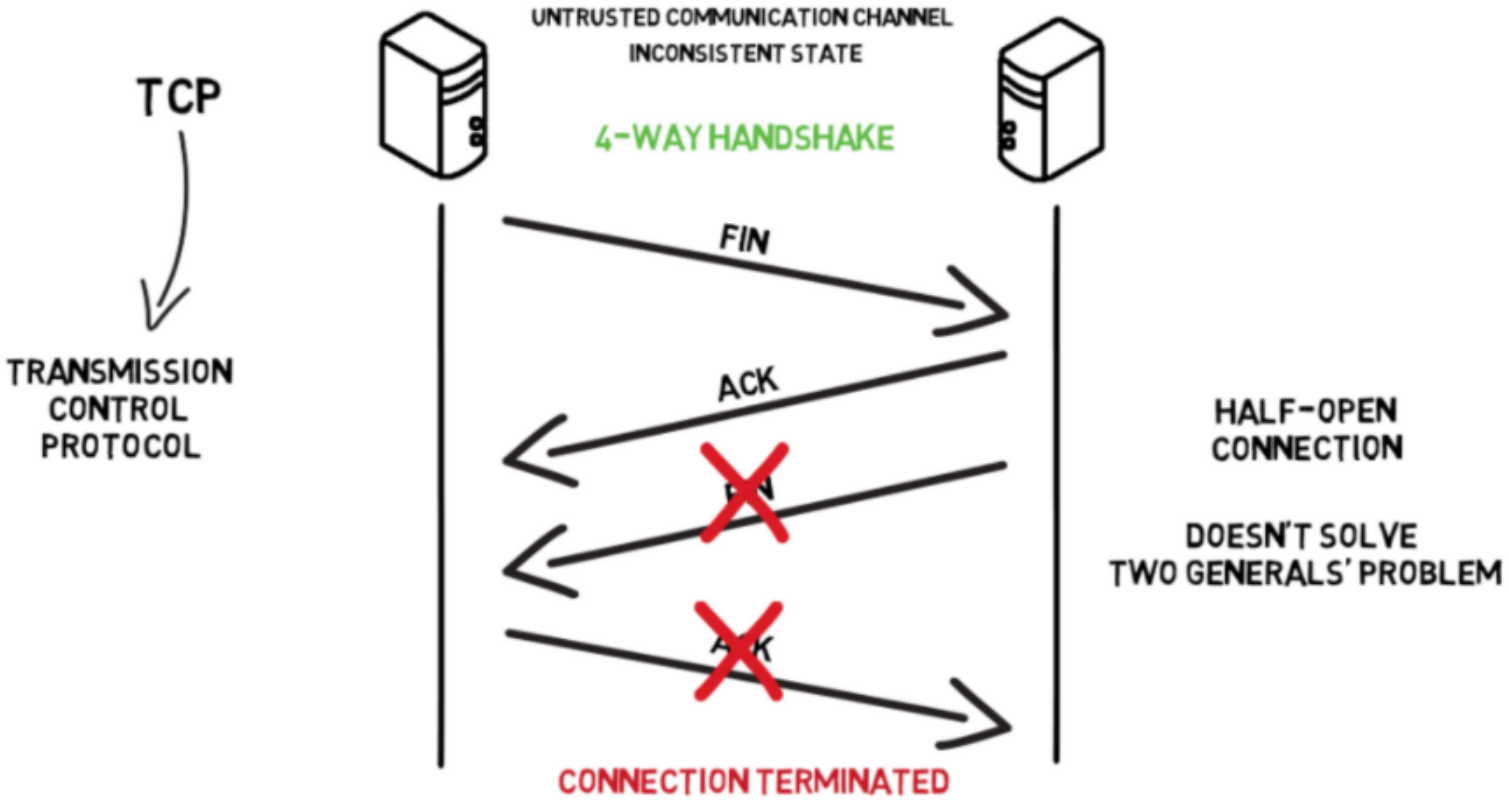
×



drahé

pomalé

2 generálové v IT (4 way handshake)



Závěr

- Řešitelnost problémů v distribuovaných systémech
  - Řada zdánlivě jednoduchých problému nemá v distribuovaných systémech 100% řešení, ale mají pragmatická řešení.
  - Některé problémy 100% řešení mají… ale jen za určitých předpokladů.
- Cílem je získat praktické a teoretické znalosti tak, aby bylo možné problémy správně klasifikovat a vyřešit

ČÁST II.:
DISTRIBUOVANÝ SYSTÉM

Cloud je distribuovaný systém

- V každém okamžiku
  - Velkého množství strojů komunikujících mezi sebou uvnitř datacentra
  - Obrovského množství klientů komunikující se stroji z vnějšku
  - Klienti komunikující mezi sebou (peer-to-peer, Torrenty)

- Masivní škálovatelnost
- Dynamická povaha (stroje přidávány/odebírány)

- Cloud == distribuovaný systém (dnes prakticky synonymum)

Distribuovaný systém

- Předchozí přezdívky
  - Peer to peer
  - Gridy
  - Clustery
  - Stroje sdílené v čase (např. Linux)

- Názvy se mění ale principy zůstávají a jsou již dlouho známy
  - Lamportovy časové značky (1978)
  - Consensus (Paxos – 1989, Raft – 2003)
  - NTP (1985)

Distribuovaný systém

- Distribuovaný systém je kolekce entit, které jsou autonomní, programovatelné, asynchronní, poruchové a komunikující spolu po nespolehlivém komunikační médiu

- Entita – proces na zařízení (PC, Smartphone, Raspberry Pi)
- Komunikační médium – bezdrátová/drátová síť

- Pesimisticky

  "Systém, ve kterém selhání počítače, o které jste vůbec nevěděli tušení, že existuje, učiní váš vlastní počítač nepoužitelný. "[Leslie Lamport]"

Distribuovaný systém

- Několik procesů asynchronně přijímajících a posílajících zprávy pomocí sítě
  - Mají vnitřní stav a vzájemně spolu komunikují
  - Lokální výpočty jsou deterministické
- Rozdíl oproti paralelním (multiprocesorovým) systémům
  1. Neznalost globálního stavu
     - proces obvykle nezná lokální stavy ostatních procesů
  2. Nedostupnost globálního časového rámce
     - události nelze uspořádat podle jejich času výskytu
  3. Nedeterminismus
     - Souběh procesu je nedeterministický, opakovaný běh může generovat různé výsledky

Distribuovaný systém

- Výhody
  - Nárůst výkonu
  - Škálovatelnost
  - Odolnost vůči výpadkům/poruchám
- Nevýhody
  - Procesy mohou kdykoliv selhat (příp. byzantinské chyby)
  - Výměna zpráv může selhat a probíhá v nepředvídatelném čase
  - Nedosažitelný běh reálného času
    - Každý proces má vlastní lokální čas, který může být rozdílný
  - Nutnost si modelovat čas uměle – logický čas

Distribuovaný systém

- Příklady
  - P2P sítě (BitTorrent)
  - MMOG (World of Tanks)
  - Blockchain (Bitcoin)
  - Mezibankovní platby
  - Internet of Things
  - Kubernetes cluster

Model

- Výpočet
- Interakce
- Čas
- Selhání

Výpočet

- Proces
  - jednotka výpočtu v distribuovaném systému (někdy nazýván uzel, hostitel, element, …)
- Množina (skupina) procesů: označována Π – je složena ze souboru N jednoznačně identifikovaných procesů $p1, p2, \ldots, pN$.
- Klasické předpoklady DS:
  - množina procesů je konstantní (N je dobře definováno)
  - procesy se navzájem znají
  - všechny procesy provádějí kopii stejného algoritmu (souhrn těchto kopií vytváří distribuovaný algoritmus)

Interakce

- Procesy komunikují zasíláním zpráv
  - $send(m, p)$ pošle zprávu m procesu p
  - $receive(m)$ přijme zprávu m
- Zprávy mohou být v některých případech jednoznačně identifikovány
  - odesílatelem zprávy
  - sekvenčním číslem, které je lokální odesílateli
- Klasickým předpoklad: každý pár procesů je propojen obousměrným komunikačním kanálem (point-point messaging)
  - plně propojená topologie může být realizována pomocí směrování (routingu)

Čas

- Přesné globální hodiny, KTERÉ by umožnily globální uspořádání výpočetních kroků v DS neexistují.
- Každý proces má své lokální hodiny.
- Lokální hodiny nemusí ukazovat přesný čas.
- Synchronizace lokálních hodin je možná jen s určitou přesností.

- Časovou značku nelze použít jako způsob spolehlivého řazení událostí

Čas uvnitř DS

- V distribuovaných systémech je obtížné uvažovat o čase nejen kvůli absenci globálních hodin, ale také proto, že obecně nelze dát časové limity na komunikaci a délku výpočtů.
- Různé možné modely:
  - Asychronní DS
  - Synchronní DS
  - Částečně synchronní DS

Synchronní vs asynchronní

- Asynchronní systém
  - Žádné časové limity na relativní rychlost vykonávání procesů.
  - Žádné časové limity na trvání přenosu zpráv.
  - Žádné časové limity na časový drift lokálních hodin.
- Synchronní systém
  - známe horní limit na relativní rychlost vykonávání procesů
  - známé horní limit na dobu přenosu zpráv.
  - procesy mají lokální hodiny a je znám horní limit na rychlosti driftu lokálních hodin vzhledem k globálním hodinám

Částečně synchronní

- Částečná sychronicita: pro většinu systému je relativně snadné definovat časové limity, které platí většinu času. Občas se ale mohou vyskytnout období, během kterých tyto časové limity neplatí.
  - zpoždění procesů: např. swappování, garbage collection
  - zpoždění komunikace: přetížení sítě, ztráta zpráv (vyžadující jejich opakovaný přenos)
- Prakticky užitečné systémy jsou částečně synchronní, to umožňuje v praxi vyřešit problémy, které jsou za předpokladu plně asynchronních DS neřešitelné.
  - některé z časových úseků synchronního běhu DS jsou dostatečně dlouhá na to, aby distribuovaný výpočet skončil

Selhání

- Procesy a komunikační kanály mohou selhat
- Selhání procesu
  - havárie (crash/fail-stop): proces přestane vykonávat algoritmus (a reagovat na zprávy) •
  - libovolné (byzantské) selhání: proces může pracovat dále (a reagovat na zprávy), ale vykonává chybný algoritmus (z důvodu softwarový chyby nebo úmyslu)
- Selhání kanálu
  - ztráta zprávy (message drop): zpráva není doručena cílovému procesu (např. kvůli přetížení sítě nebo přetečení zásobníku v OS u přijímacího procesu)
  - rozdělení (partitioning): procesy jsou rozdělené do disjunktních množin (oddílů - partitions) tak, že v rámci oddílu je komunikace možná, ale mezi oddíly nikoliv

Předpoklady na komunikační kanál

- Spolehlivé doručování
  - Pokud proces p pošle zprávu procesu q ani p a ani q nehavaruje, pak q nakonec zprávu obdrží.
- Žádná duplikace
  - Žádná zpráva není doručena vícekrát než jednou.
- Žádné vytváření
  - Je-li zpráva m doručena procesu p, tak zpráva m byla dříve poslána nějakým procesem q procesu p.
- Garantované pořadí doručování:
  - Odešle-li proces p procesu q zprávy $m1$ a $m2$, tak pokud byla $m1$ odeslána dříve než $m2$, tak $m2$ nemůže být doručena aniž by předtím byla doručena $m1$.

Chybné předpoklady

- Řada DS je **zbytečně komplexních**, komplexita je způsobena chybami, které je třeba později záplatovat. Chyby vycházejí z **mylných předpokladů**.

- Typické mylné předpoklady
  - Síť je spolehlivá
  - Síť je zabezpečená
  - Síť je homogenní
  - Topologie sítě se nemění
  - Síť ma nulovou latenci
  - Neomezená kapacita sítě
  - Systém má jednoho administrátora

Shrnutí

- Distribuované systémy jsou všude kolem nás a jejich význam a složitost dále roste.
- Základním rozdílem mezi paralelními a distribuovanými výpočty jsou: absence sdílené paměti, absence globálních hodin a nezávislá selhání.

ČÁST III.:
ZÁKLADNÍ PROBLÉMY

Základní problémy

- Detekce selhání a členství ve skupině
- Časové značky
- Konsenzus
- Volba vůdce

- V tomto předmětu pouze lehký úvod, detaily a implementace bude součástí předmětů v navazujícím studiu

Detekce selhání a členství ve skupině

- Systémy založeny na skupinách procesů
  - cloudy / datová centra
  - replikované servery
  - distribuované databáze
- Frekvence selhání roste lineárně s počtem procesů ve skupině => selhání jsou běžná.

- V následujícím předpokládáme havárii procesu (crash-stop) jako model selhání a komunikaci přes ztrátový/nedokonalý FIFO kanál.

Detekce selhání a členství ve skupině

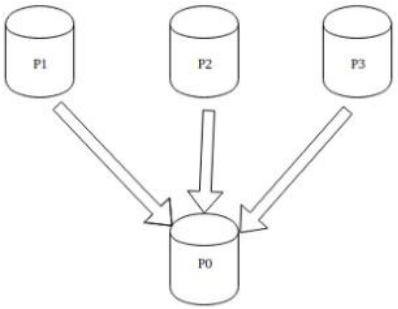
- Podprotokoly
  - Seznam členů
  - Správa členství
    - Příchod nového člena
    - Odchod člena
    - **Tiché selhání procesu**

- Vlastnosti
  - Úplnost: každý selhaný člen bude detekován
  - Přesnost: každý detekovaný člen opravdu selhal
  - Rychlost detekce
  - Škálovatelnost

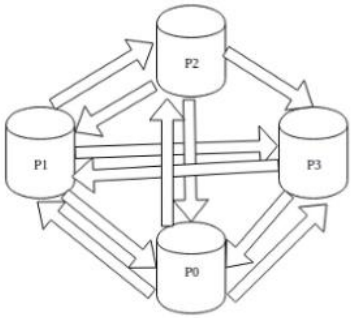
- V praxi nelze dosáhnout úplnost a přesnosti zároveň, cílem je úplnost a co největší přesnost s ohledem na další vlastnosti

Základní protokoly

- Centralizovaný heartbeat
- Kruhový heartbeat
- All-to-all heart
- SWIM Failure Detector
  - Scalable weakly consistent infection-style proces group membership protocol)



Centralized Heartbeating



All to All Heartbeating

Časové značky – reálný čas

- Vlastnosti
  - Mimoběžnost: rozdíl mezi časy
  - Drift: rozdíl v rychlosti změny času
- Algoritmy
  - Cristianův algoritmus
  - NTP
- V reálném světě vždy nenulová chyba synchronizace.
- Dokud bude latence nenulová a neznámá, chyby se nezbavíme! Pomocí fyzických hodin lze uspořádávat jen události v „pomalých“ distribuovaných výpočtech

Časové značky - logické

- Kauzální vztah mezi událostmi: první událost může ovlivnit druhou
- Místo časové značky používáme hodnotu

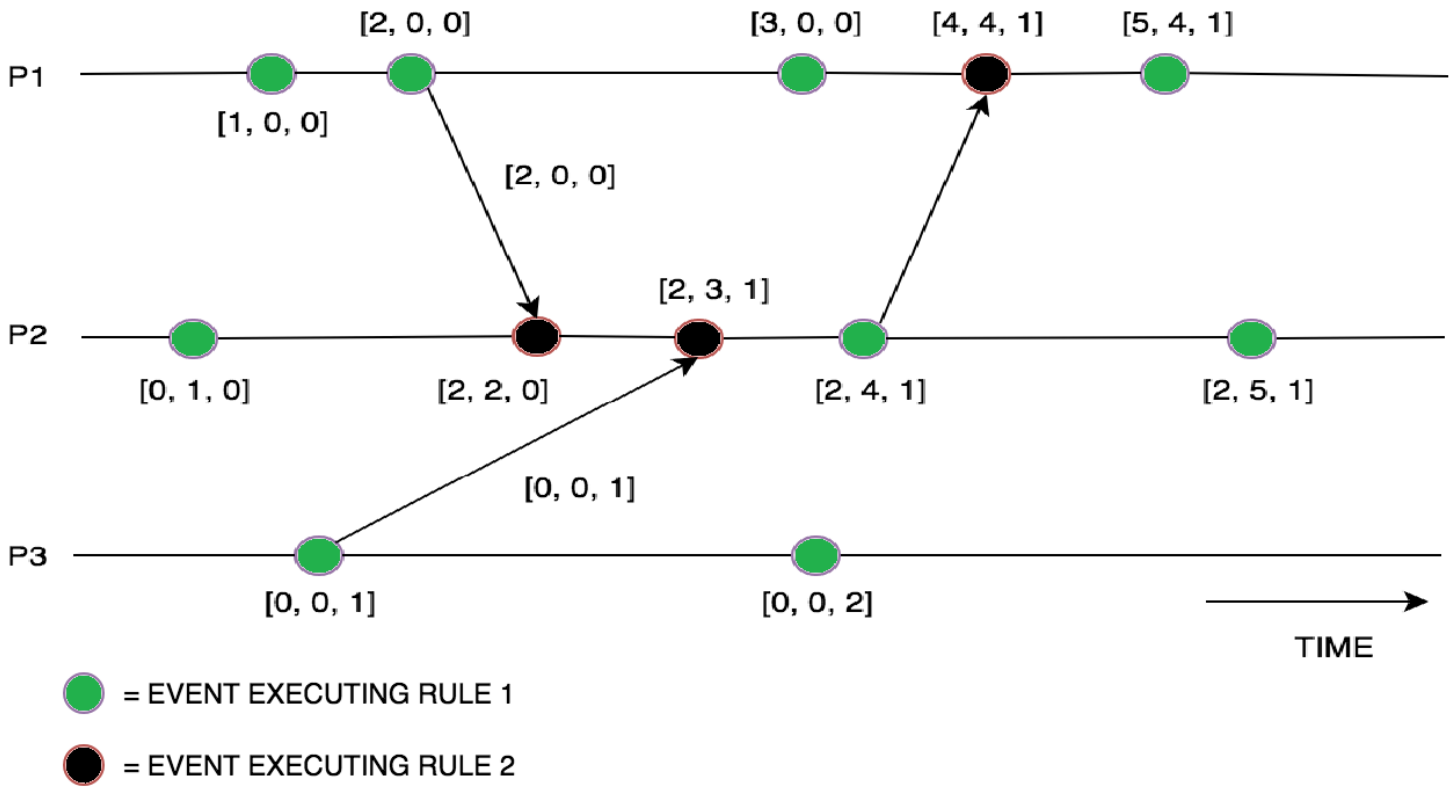
- Lamportovy logické hodiny
  - jedna hodnota
  - respektují kauzalitu
  - ale nerozliší současné události
- Vektorové hodiny
  - vektor o velikosti dle počtu členů
  - implikují kauzalitu
  - potřebují více místa, ale rozliší současné události

Synchronizace

- Každý proces pi si udržuje lokální vektorové hodiny Vi a nastavuje

1. Před provedením akce v procesu pi se Vi inkrementuje o 1, tj. $Vi[i] := Vi[i] + 1$
2. Pošle-li proces pi zprávu m procesu pj , nastaví vektorovou časovou značku $ts(m)$ zprávy m na Vi
   (poté, co provedl krok 1)
3. Proces pj po přijetí zprávy m
   1. nastaví své hodiny $Vj[k] := \max Vj[k], ts(m)[k]$ pro všechna $k = 1, \ldots, N$ (tzv. sloučení)
   2. poté inkrementuje $Vj[j]$ a předá zprávu m aplikaci.

Příklad vektorové hodiny



Konsenzus

- Model
  - Asynchronní systém se selháními
  - procesy mohou havarovat (fail-stop, tj. nikoliv byzantsky) ▪
  - zprávy se mohou ztrácet (ale dodržují pořadí → nedokonalý FIFO kanál)
  - musí garantovat bezpečnost a měl by maximalizovat živost (dostupnost)
  - obojí garantovat nelze (FLP teorém)