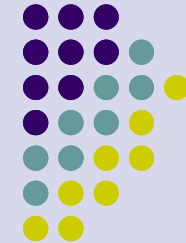
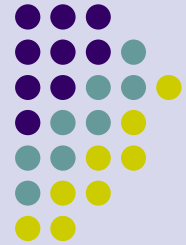


Algoritmy shody



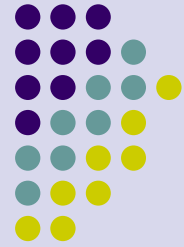
Přednášky z Distribuovaných systémů
Ing. Jiří Ledvina, CSc.

Shoda



- N procesů se chtějí dohodnout na hodnotě
 - Např. Na synchronizované akci (go/abort)
- Konsensus může být dosažen i za přítomnosti poruch
 - Porucha procesu – krach, svévolná porucha
 - Komunikační porucha – ztracené nebo zničené zprávy

Synchronní a asynchronní systémy



- Synchronní systémy
 - omezené zpoždění přenosu zpráv (latency)
 - procesy mají synchronizované hodiny
 - časy zpracování jsou omezené
 - zhroucení (crash failure) procesu může být přesně detekováno
- Asynchronní systémy - bez omezujících předpokladů
 - zhroucení (crash failure) procesu nemůže být přesně detekováno
- Failstop
 - podobné jako asynchronní, ale zhroucení je přesně detekováno



Algoritmus shody

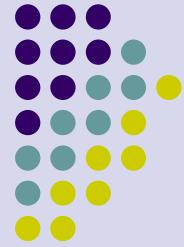
- Algoritmus shody
 - Všechny procesy P_i začínají ve stavu „nerozhodnutý“
 - Každý P_i navrne hodnotu V_i z množiny D a pošle ji ostatním procesům
 - Shody je dosaženo pokud se všechny nechybuující procesy shodnou na téže hodnotě d
 - Každý nechybuující proces P_i nastaví svou rozhodovací proměnnou na d a změní svůj stav na „rozhodnutý“



Algoritmus shody - princip

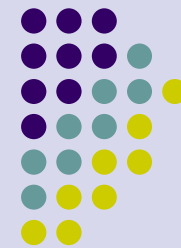
- Algoritmus probíhá ve více kolech (round)
- Jedno kolo zahrnuje
 - poslání zprávy do skupiny procesů
 - příjem zpráv z předchozího kola
 - lokální zpracování (rozhodnutí, zastavení)
- Jednoduché, neefektivní v pomalých sítích

Požadavky na shodu



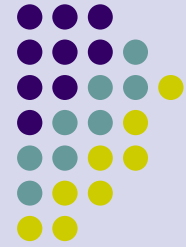
- Ukončení – každý proces nastaví vybranou (rozhodnutou) hodnotu
 - Toto není možné při krachu procesoru v asynchronních systémech
- Souhlas (dohoda) – výběr hodnoty je shodný pro všechny korektní procesy
 - Svévolné (Byzantinské) poruchy mohou způsobit nekonzistentnost a brání dohodě
- Integrita (celistvost) – pokud všechny korektní procesy P_i navrhnou tutéž hodnotu d , pak dohodnutá hodnota je d
- Shoda může zahrnovat fázi návrhu a fázi dohody

Interaktivní konzistentnost



- Interaktivní konzistentnost je speciální případ shody, kde se procesy dohodnou na vektoru hodnot, pro každý proces na jedné hodnotě
- Ukončení – každý korektní proces nastaví svůj rozhodovací vektor
- Souhlas – rozhodovací vektor je stejný pro všechny korektní procesy
- Integrita – jestliže P_i je korektní, pak se všechny korektní procesy dohodnou na V_i jako na i -tém prvku rozhodovacího vektoru
- Hodnota V_i pro chybující proces může být ignorována nebo dohodnuta shodou (konsensus)

Shoda v synchronních systémech



- Pro systém se $2k+1$ procesy, kde maximálně k procesů může být chybných potřebuje algoritmus $k+1$ kroků (s timeoutem) a s využitím základního multicastu.
- Každý proces P_i má seznam $V-i_1, V-i_2, \dots$ přenesených hodnot v krocích $1, 2, \dots$

Initially $V-i_0 = \{\}$; $V-i_1 = \{v_i\}$
for round = 1 to $k+1$ do {
 multicast ($V-i_{round} - V-i_{round-1}$)
 $V-i_{round+1} = V-i_{round}$
 for each $V-j_{round}$ received
 $V-i_{round+1} = V-i_{round+1} \cup V-j_{round}$
 }
}
 $d-i = \text{minimum } V-i_{k+2}$

Byzantinští generálové



- Tři nebo více generálů se potřebuje vzájemně seznámit se svými záměry
- Jeden nebo více generálů může být zrádce, který dává chybné informace
- K řešení tohoto problému používají protokol
 - Každý z generálů posílá svou informaci ostatním (předpokládáme spolehlivou komunikaci)
 - Jakmile generál shromáždí všechny hodnoty, pošle vektor hodnot ostatním generálům
 - S využitím hlasovacího mechanismu může každý generál získat správné hodnoty

Byzantinští generálové



- Problém není řešitelný pro $N \leq 3$
- Pro k zrádců je třeba $2k+1$ loajálních generálů
 - Problém nemůže být řešen pokud chybný proces lze konzistentně
 - Lze použít zabezpečení zprávy (šifrování, digitální podpis)

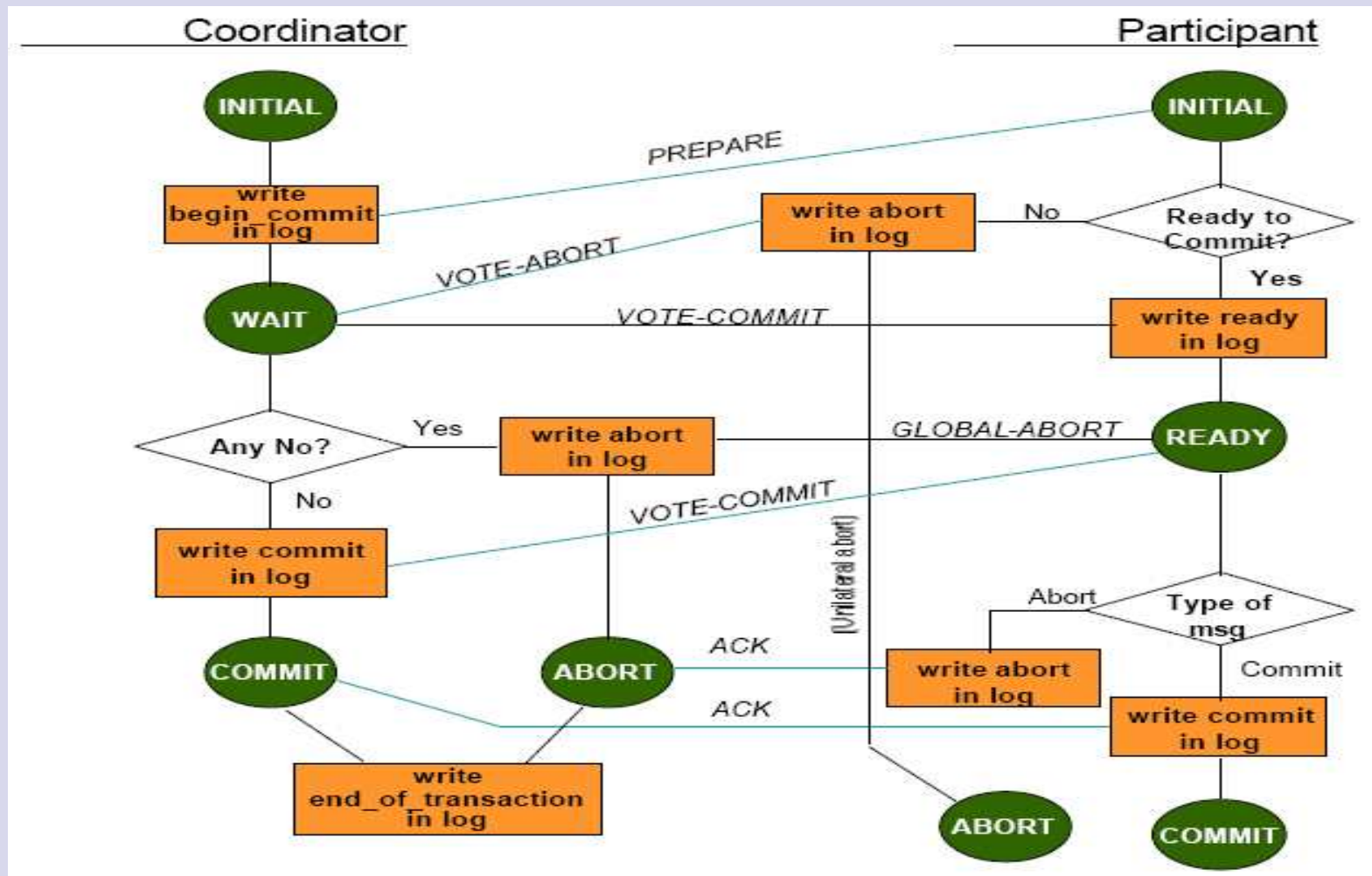
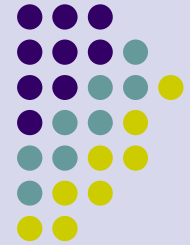
•

2-fázový commit (2PC)

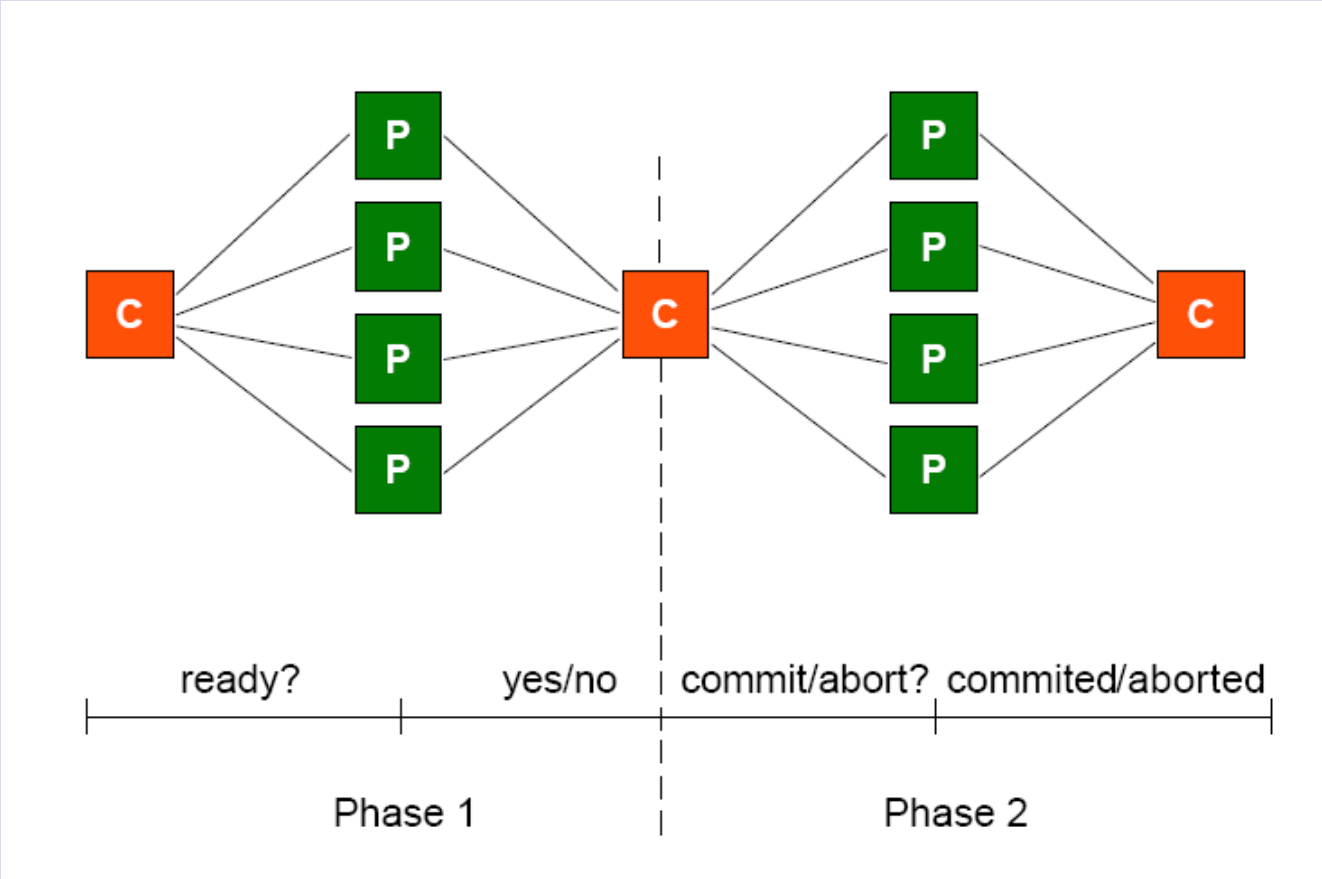
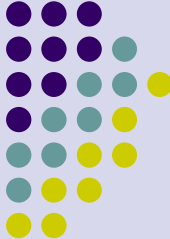


- **Koordinátor** : Koordinátor je proces, který vyvolá transakci a řídí její průběh
 - **Účastník (participant)** : proces, který se účastní provedení transakce v jiném uzlu
1. **Phase 1**: Koordinátor pošle ostatním účastníkům požadavek – zprávu účastníci volí mezi provedením (commit) a zrušením (abort)
 2. **Phase 2**: Všichni provedou akci
 3. **Pravidla provedení**: Koordinátor provede transakci pokud všichni účastníci volí její provedení

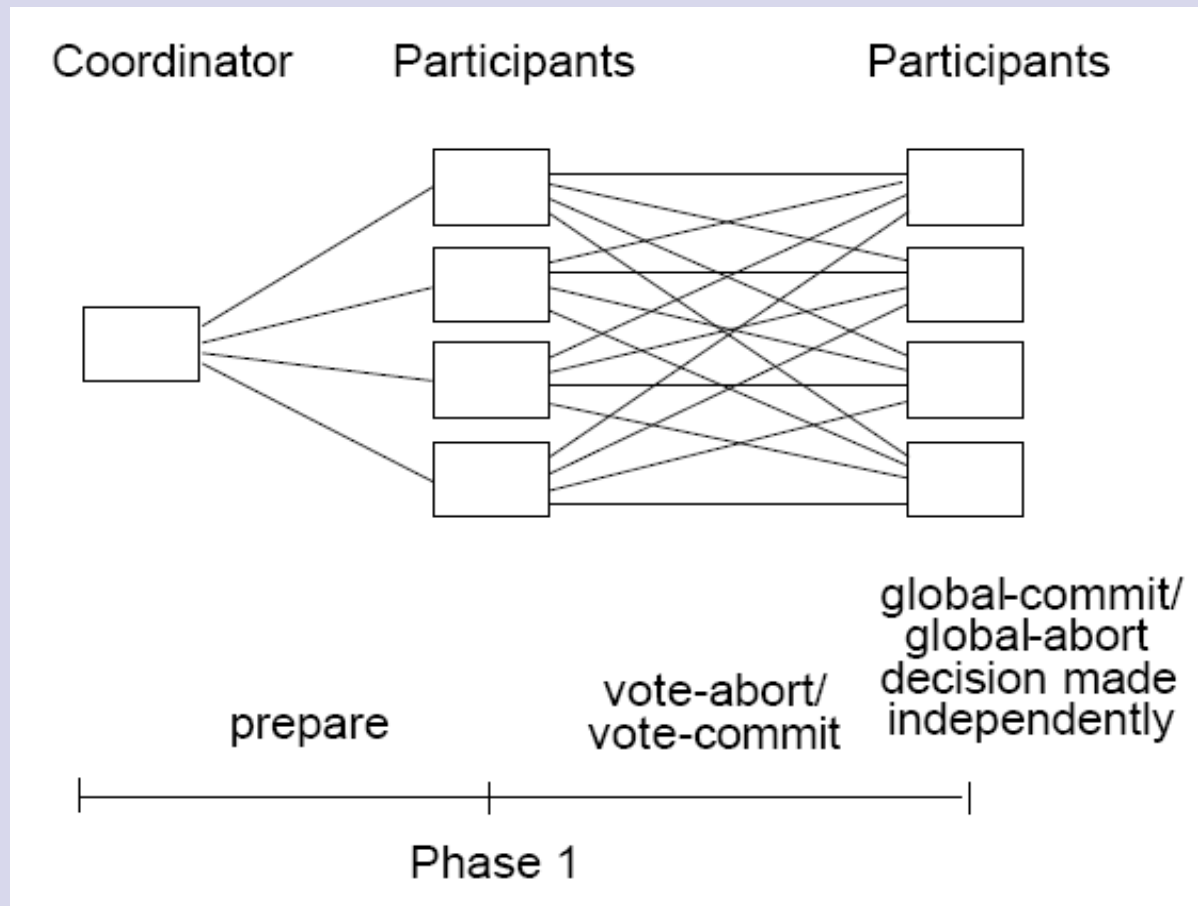
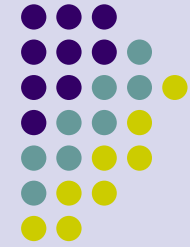
Akce 2PC protokolu



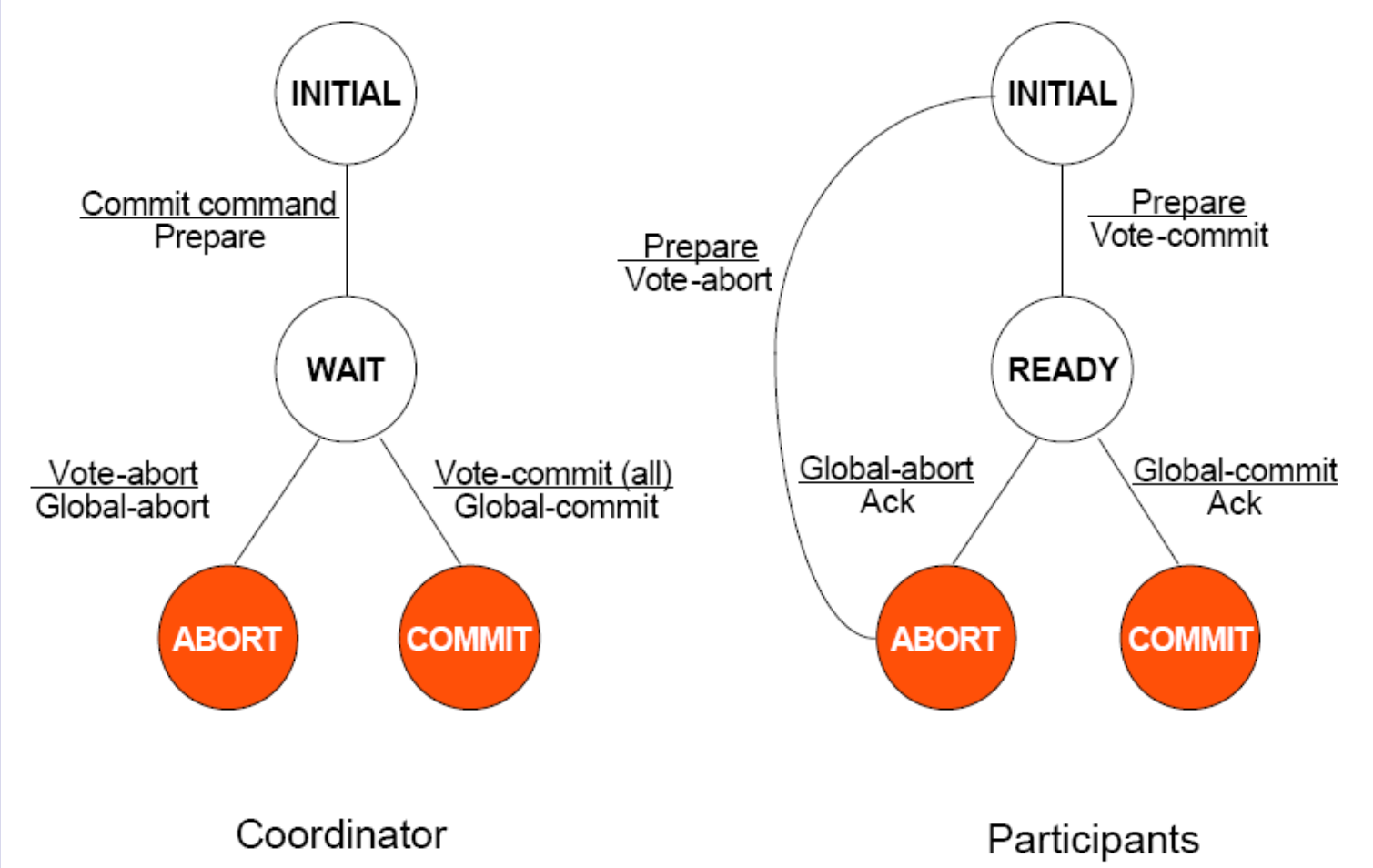
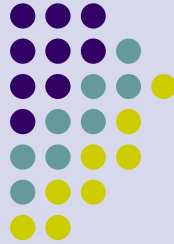
Centralizovaný 2PC



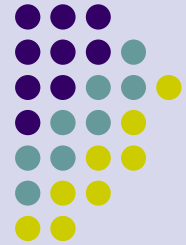
Distribovaný 2PC



Stavové přechody 2PC

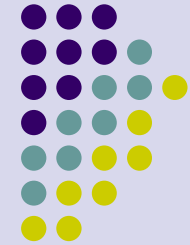


Problémy s 2PC



- blokování
 - Ready znamená, že účastník (participant) čeká na koordinátora
 - Pokud koordinátor skončí chybou, čeká participant na obnovu
 - Blokování redukuje dostupnost
- Není možná nezávislá obnova
- Hledáme protokol pro obnovu, který by neblokoval při výpadku účastníka nebo koordinátora
- Takovým protokolem je – 3PC

3-fázový commit (3PC)



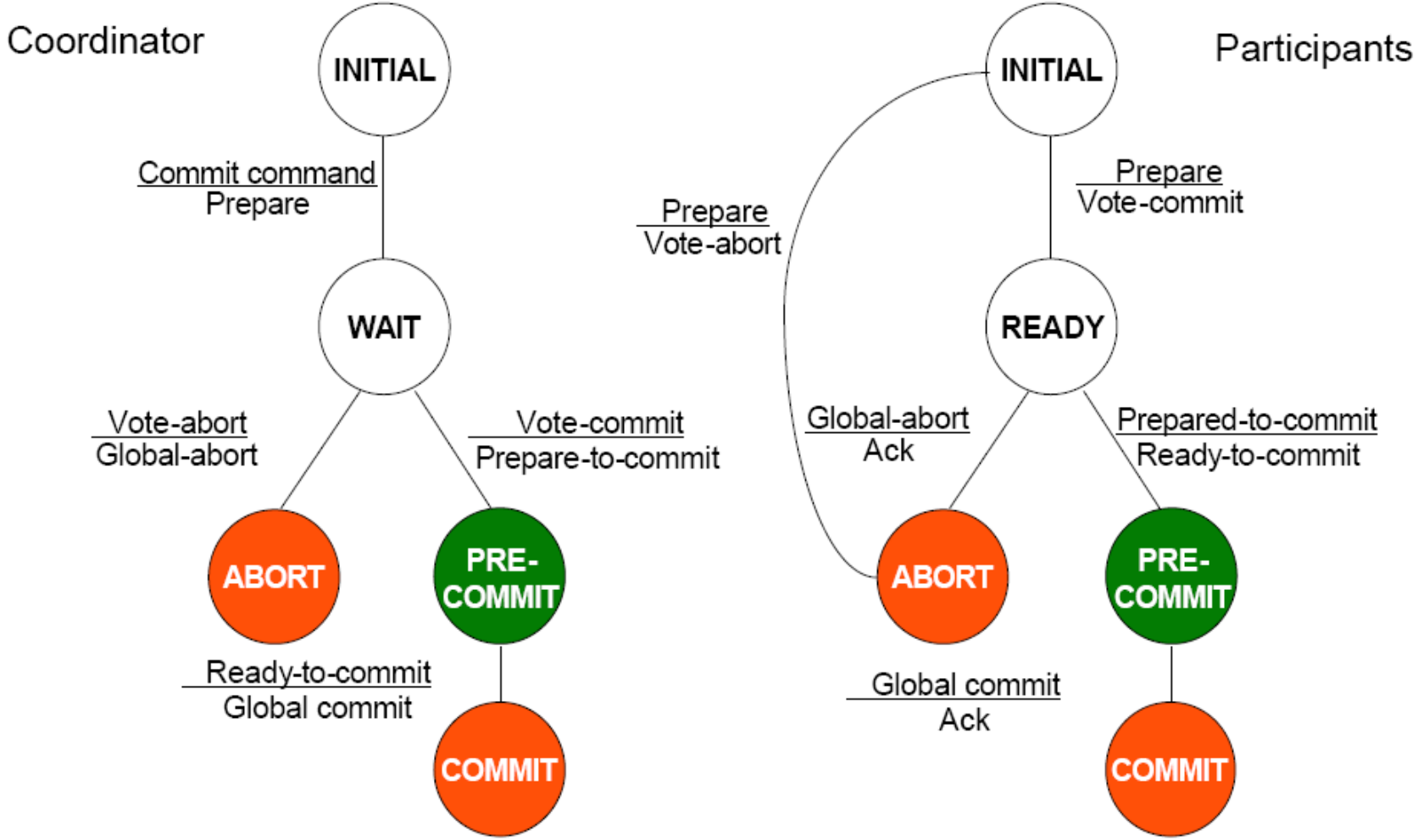
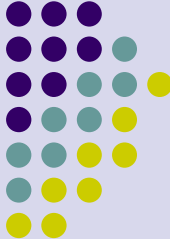
- 3PC je neblokující.
- Protokol ukončení (commit) je neblokující jestliže
 - Je synchronní během jednoho přechodu
 - Jeho diagram přechodů zahrnuje
 - Žádný stav, který by sousedil s oběma commit i abort stavy
 - Žádné nespáchatelné (non commit) stavy, sousedící s commit stavem
- sousedství: možnost přejít z jednoho stavu do druhého jedním přechodem
- spáchatelný: všechny strany volily commit transakci



3-fázový commit (3PC)

- Fáze hlasování, fáze rozhodování, fáze provedení (commit).
- Fáze hlasování
 - Koordinátor posílá požadavek hlasování
 - Všichni odpoví commit nebo abort
- Fáze rozhodování
 - Po příjmu všech odpovědí rozhodne koordinátor o výsledku – commit nebo abort
 - Koordinátor posílá abort nebo prepare-to-commit
 - Všichni odpovídají ACK
- Fáze provedení (commit)
 - Po získání všech ACK posílá koordinátor commit
 - Všichni potvrzují ACK

Stavy ve 3PC



3PC - komunikace

