

PDV 10 2019/2020

Výpočet globálního stavu

Michal Jakob


michal.jakob@fel.cvut.cz

Centrum umělé inteligence, katedra počítačů, FEL ČVUT





Globální Stav

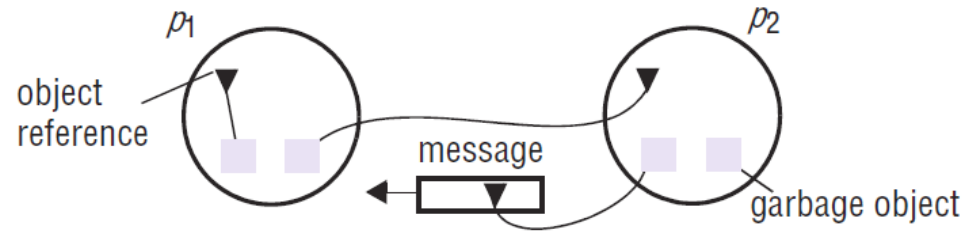


Globální stav: množina lokální **stavů procesů** v DS a **stavů** všech **komunikačních kanálů** v jednom okamžiku*.

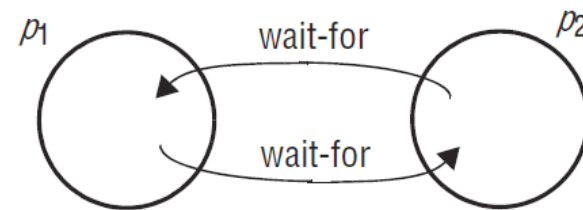
Globální snapshot: záznam globální stavu.

Příklady

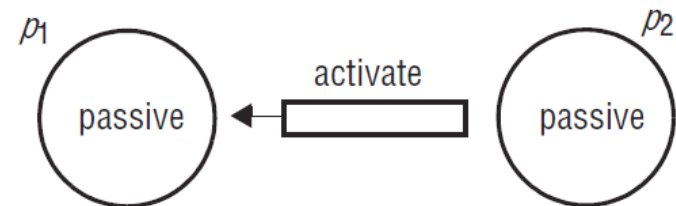
Garbage collection: nutnost identifikovat objekty, na které není *globálně* žádná reference.



Detekce uváznutí (deadlock): nutnost identifikovat cykly *globálním* wait-for grafu.



Detekce ukončení výpočtu: nutnost zajistit, že *všechny* procesy jsou pasivní a v *žádném* kanálu přenosu není žádná potenciálně aktivační zpráva.



Checkpointing za účelem obnovení globálního stavu systému.

...

Globální stav

Pokud bychom měli **globální hodiny**, tak zaznamenat globální stav jednoduché: všechny procesy by zaznamenaly stav v dohodnutý čas, tj. v jednom **fyzickém okamžiku**.

Jak zaznamenat globální stav **bez** globálních hodin?

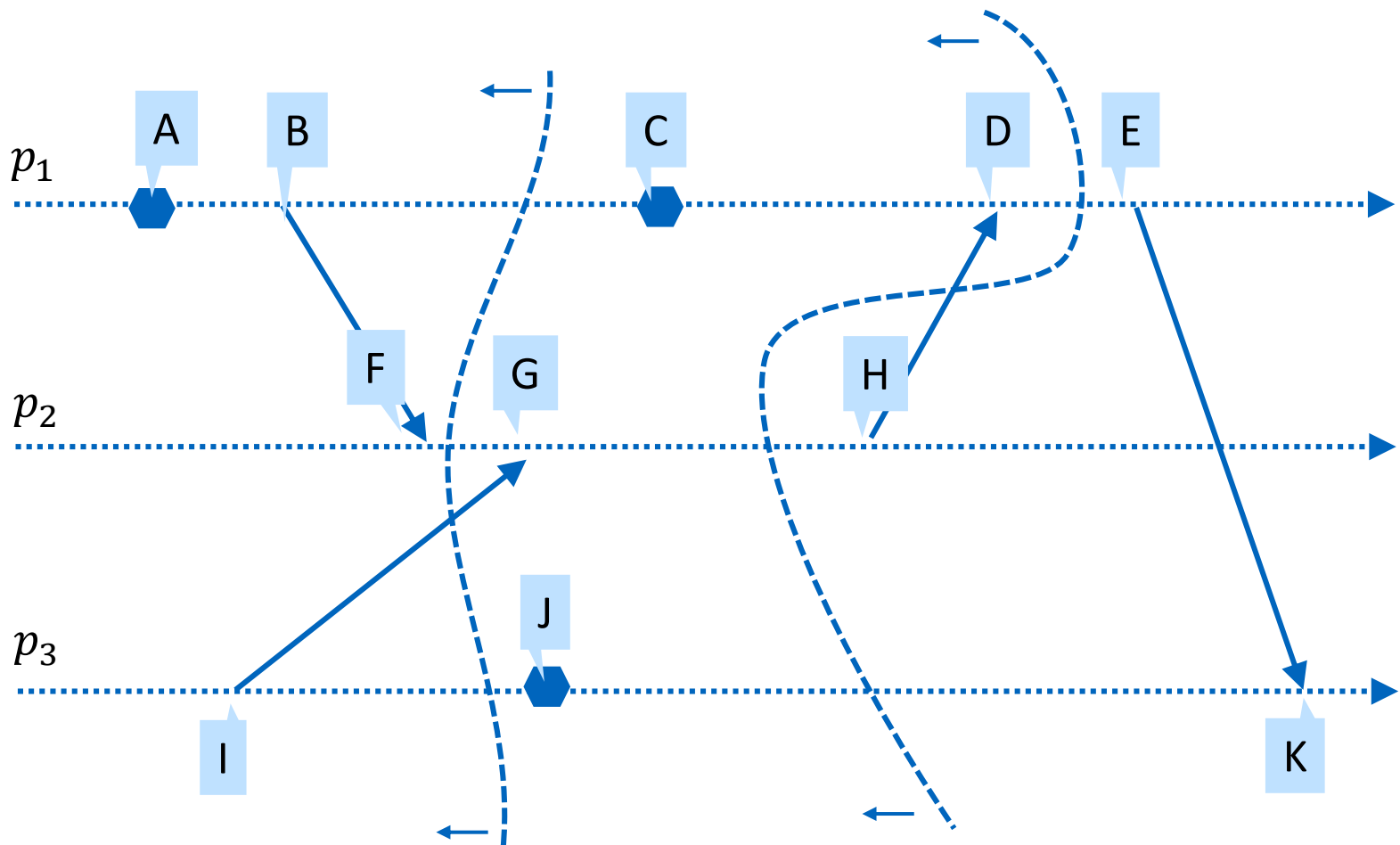
→ **Logický okamžik**

Řez distribuovaného výpočtu

Řez: časová hranice v každém procesu a v každém komunikačním kanále.

- události, které nastanou před řezem, jsou **v řezu**
- události, které nastanou po něm, jsou **mimo řez**.

Řez: Příklad



Konzistentní řez

Definice (optimistická)

Řez R je konzistentní pokud splňuje kauzalitu, tj. pokud pro každý pár událostí e, f v systému platí:

$$f \in R \wedge e \rightarrow f \Rightarrow e \in R$$

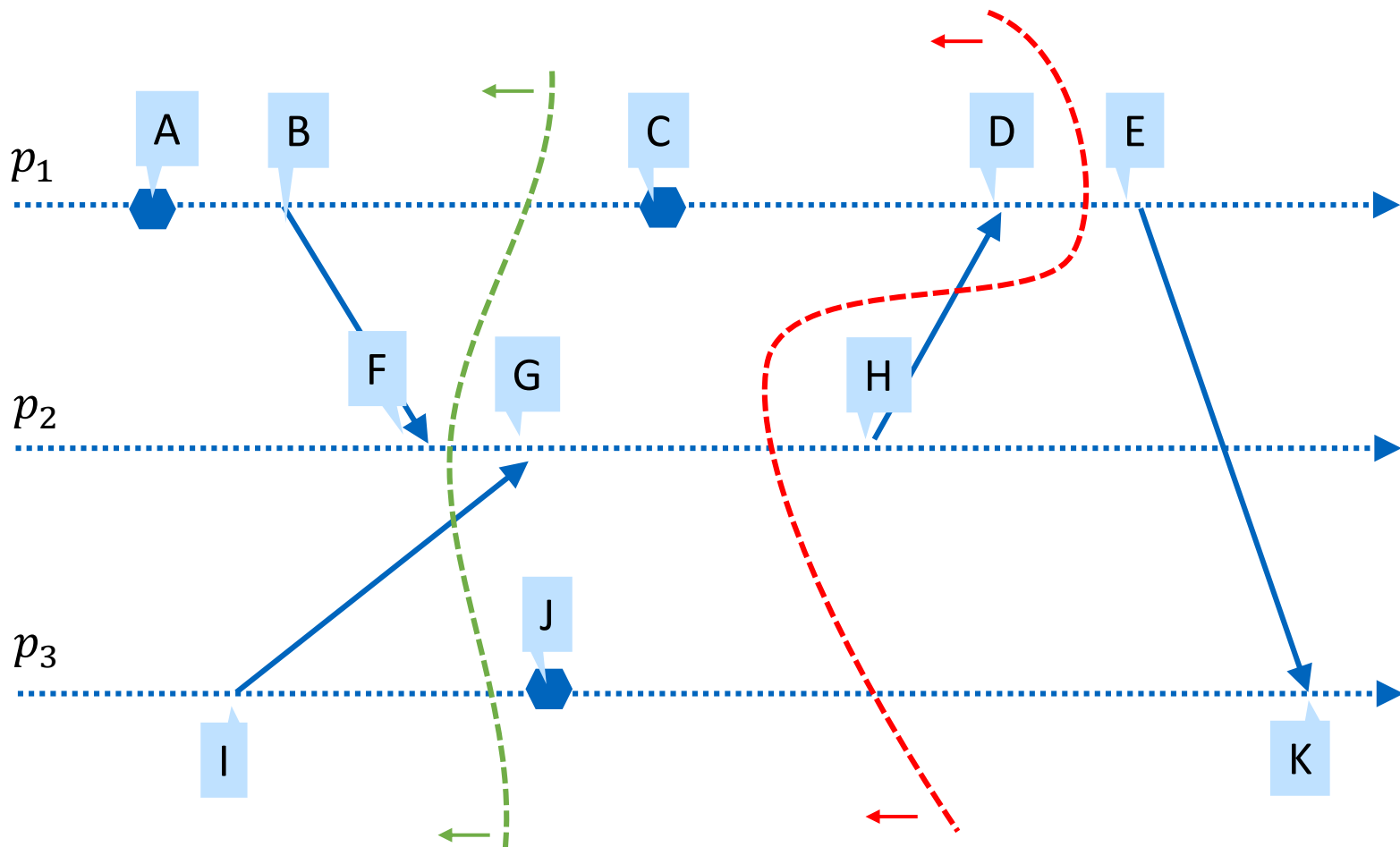
tj. pokud řez obsahuje nějakou událost, obsahuje i všechny, které ji předcházejí dle relace **stalo se před** (tj. nelze, aby v řezu byl „**důsledek**“ a nebyla tam „**příčina**“.)

Konzistentní řez = **logický okamžik**

Konzistentní globální stav odpovídá konzistentnímu řezu.

- Globální stav je konzistentní, pokud by mohl být pozorován externím pozorovatelem.

Řez: Příklad



konzistentní řez
(mohl, ale *nemusel*
být pozorován)

nekonzistentní řez
(nikdy nemohl být pozorován)



Výpočet konzistentního globálního stavu

Problém výpočtu globálního stavu

Cíl: Zaznamenat **globální snapshot**, tj. stav pro každý proces a každý komunikační kanál.

Požadavek: Zaznamenání snapshotu by neměla **interferovat** s během distribuované aplikace a neměla by vyžadovat po aplikaci zastavení posílání aplikačních zpráv.

Model

(Existují i algoritmy se slabšími předpoklady)

- skupina N procesů p_1, \dots, p_N
- **FIFO perfektní komunikační kanál** mezi každým párem procesů, tj. zprávy se neduplikují, nevznikají, neztrácejí a jsou doručovány v pořadí odeslání (značení: $C_{i,j}$ kanál přenáší zprávy z procesu p_i do procesu p_j)
- **asynchronní systém:** neznáma, ale **konečná latence**

Předpoklad: Každý proces je schopen zaznamenat svůj vlastní **aplikační stav** (případně low-level systémový stav).

Chandy-Lamport algoritmus pro distribuovaný globální snapshot

(Vytváření snapshotu je distribuované.)

Speciální zpráva: **ZNAČKA** ■

Jeden (libovolný) z procesů iniciuje vytvoření snapshotů.

Procesy reagují na příjem zprávy **ZNAČKA** ■ .

Chandy-Lamport algoritmus pro globální snapshot

Zahájení tvorby snapshotu

Iniciující **proces** p_i odešle ZNAČKU ■ všem ostatním procesům (i sobě)

Příjem ZNAČKY ■ procesem p_i kanálem $C_{m,i}$

if (p_i dosud nezaznamenal svůj stav) **then**

p_i **zaznamená** svůj stav (a vykoná pravidlo pro odeslání ZNAČKY ■);

p_i **zaznamená** stav kanálu $C_{m,i}$ jako prázdnou množinu;

p_i každým odchozím **kanálem** $C_{i,j}$ **odešle** jednu ZNAČKU ■ (předtím než skrze $C_{i,j}$ pošle jakoukoliv jinou zprávu);

p_i **zapne zaznamenávání** zpráv doručených skrze všechny ostatní příchozí kanály $C_{j,i}$ kromě $C_{m,i}$

else

p_i **zaznamená** stav kanálu $C_{m,i}$ jako množinu všech zpráv, které p_i obdržel skrze $C_{m,i}$ od doby, kdy zahájil záznam $C_{m,i}$;

p_i záznam **ukončí**;

end if

Ukončení

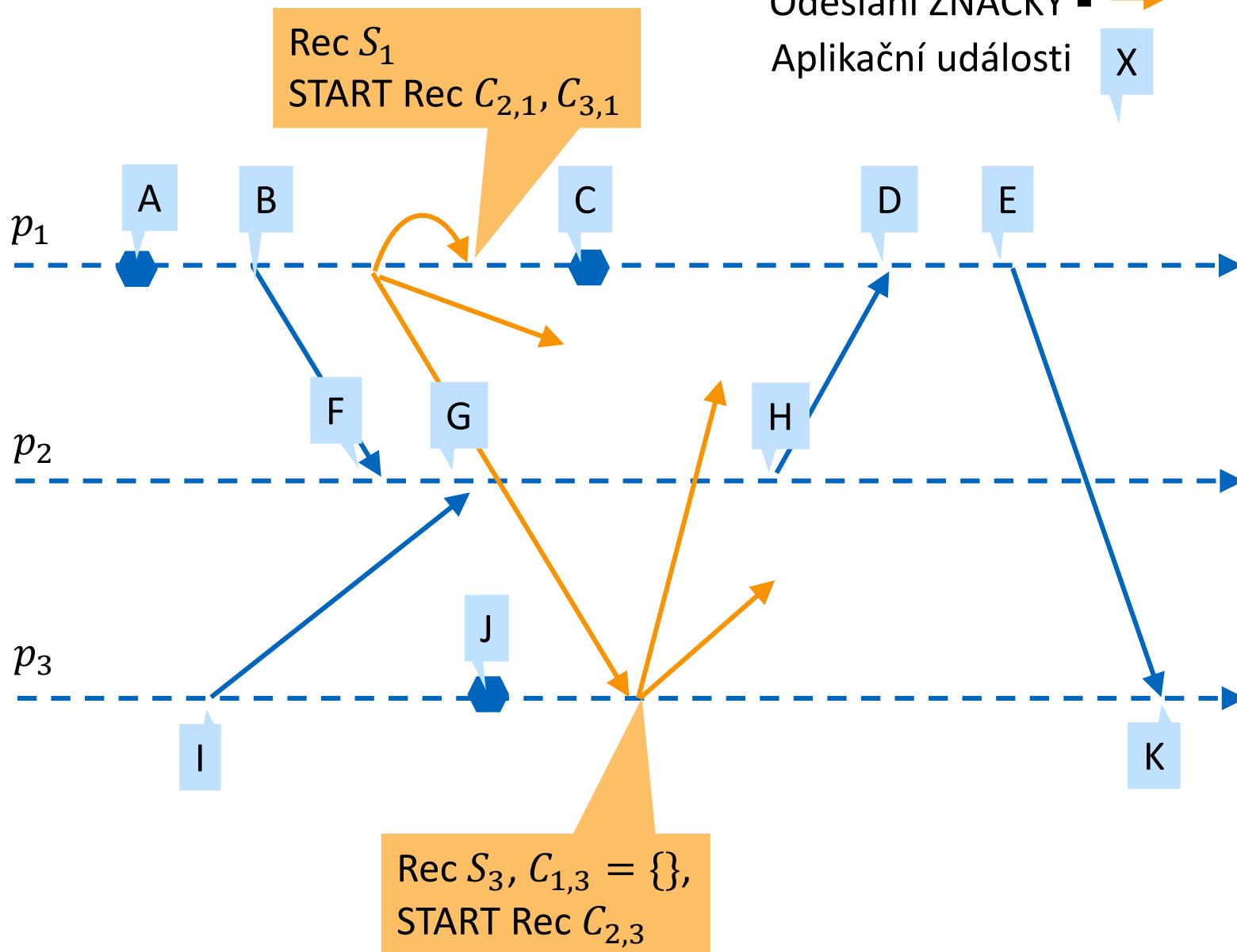
Algoritmus končí jakmile:

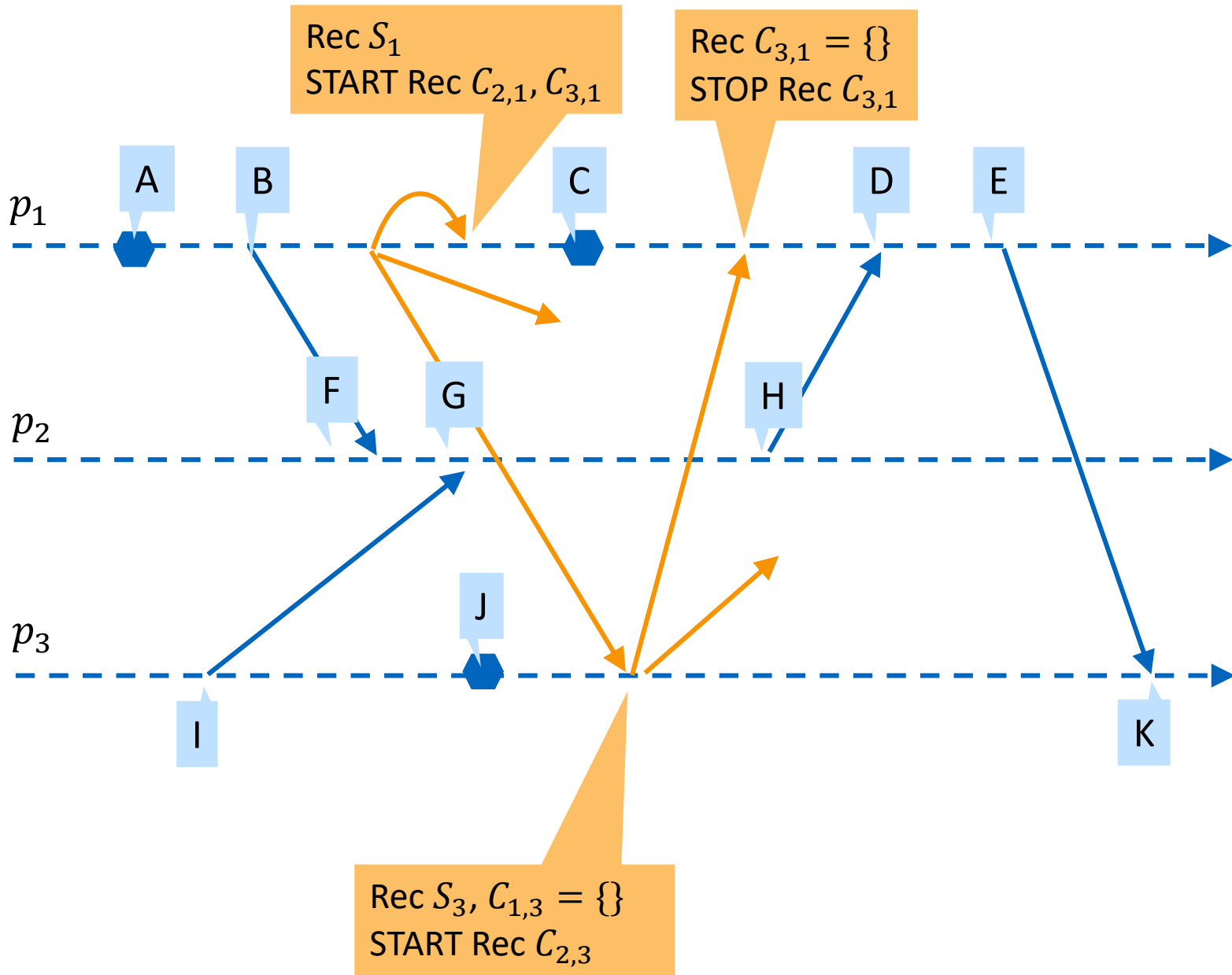
- všechny procesy přijaly ZNAČKU ▪ (aby zaznamenaly svůj stav) a
- všechny procesy přijaly Značku ▪ na všech $N - 1$ svých příchozích kanálech (aby zaznamenaly stav na všech kanálech)

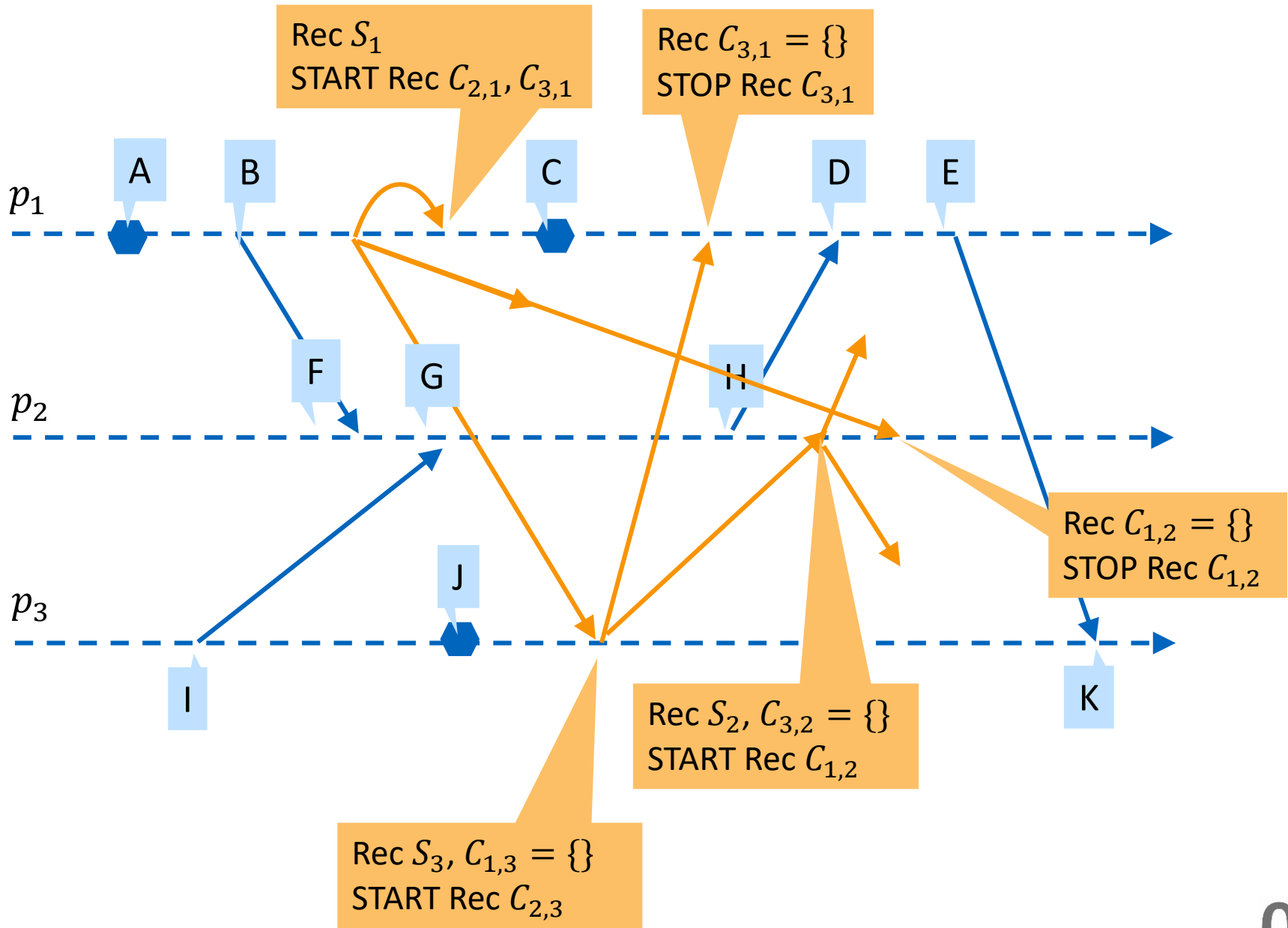
Následně (je-li potřeba) mohou být jednotlivé fragmenty globálního stavu posbírány centrálním serverem a poskládán **plný globální snapshot**.

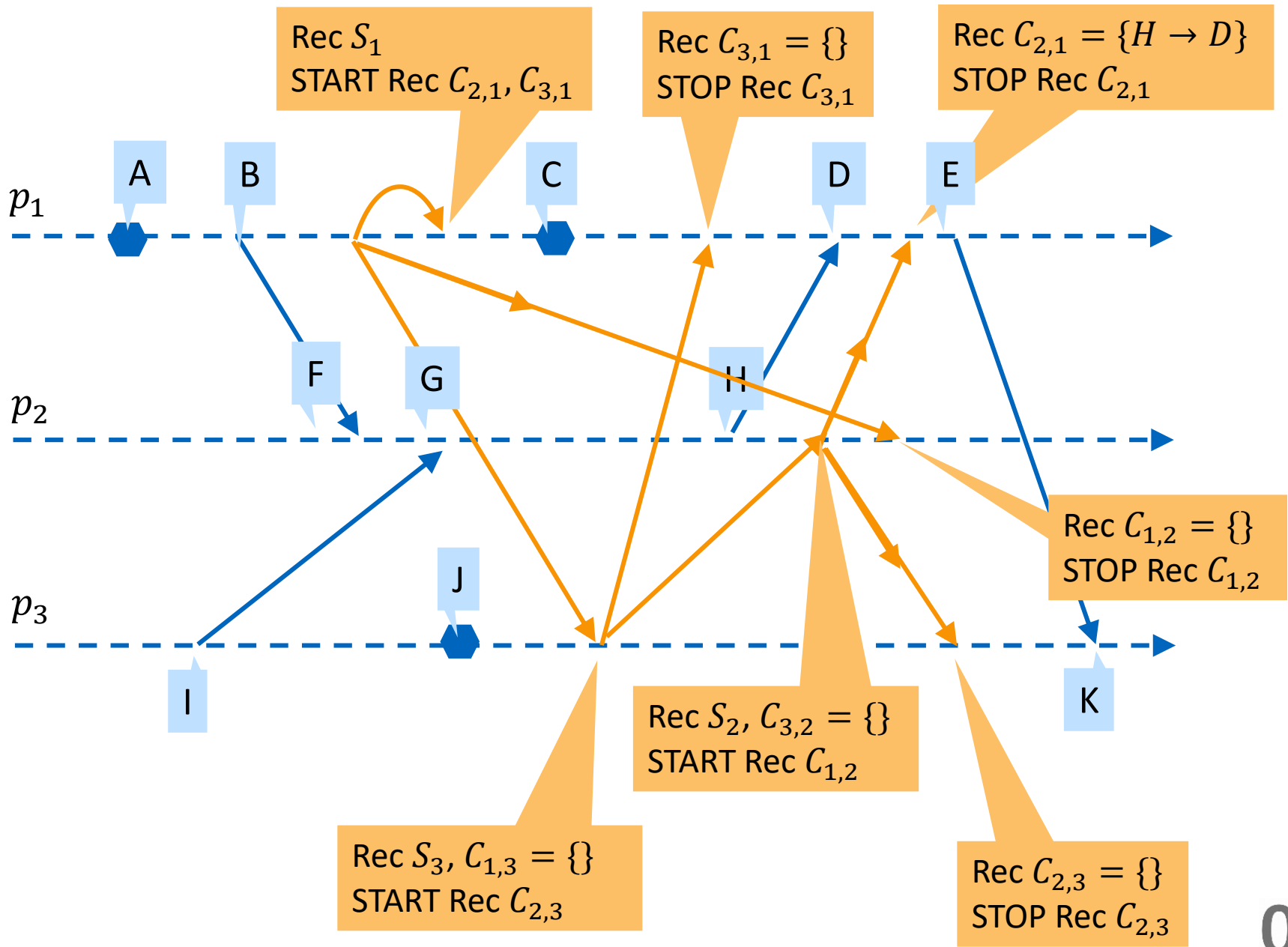
Příklad

Odeslání ZNAČKY ■ →
Aplikační události X

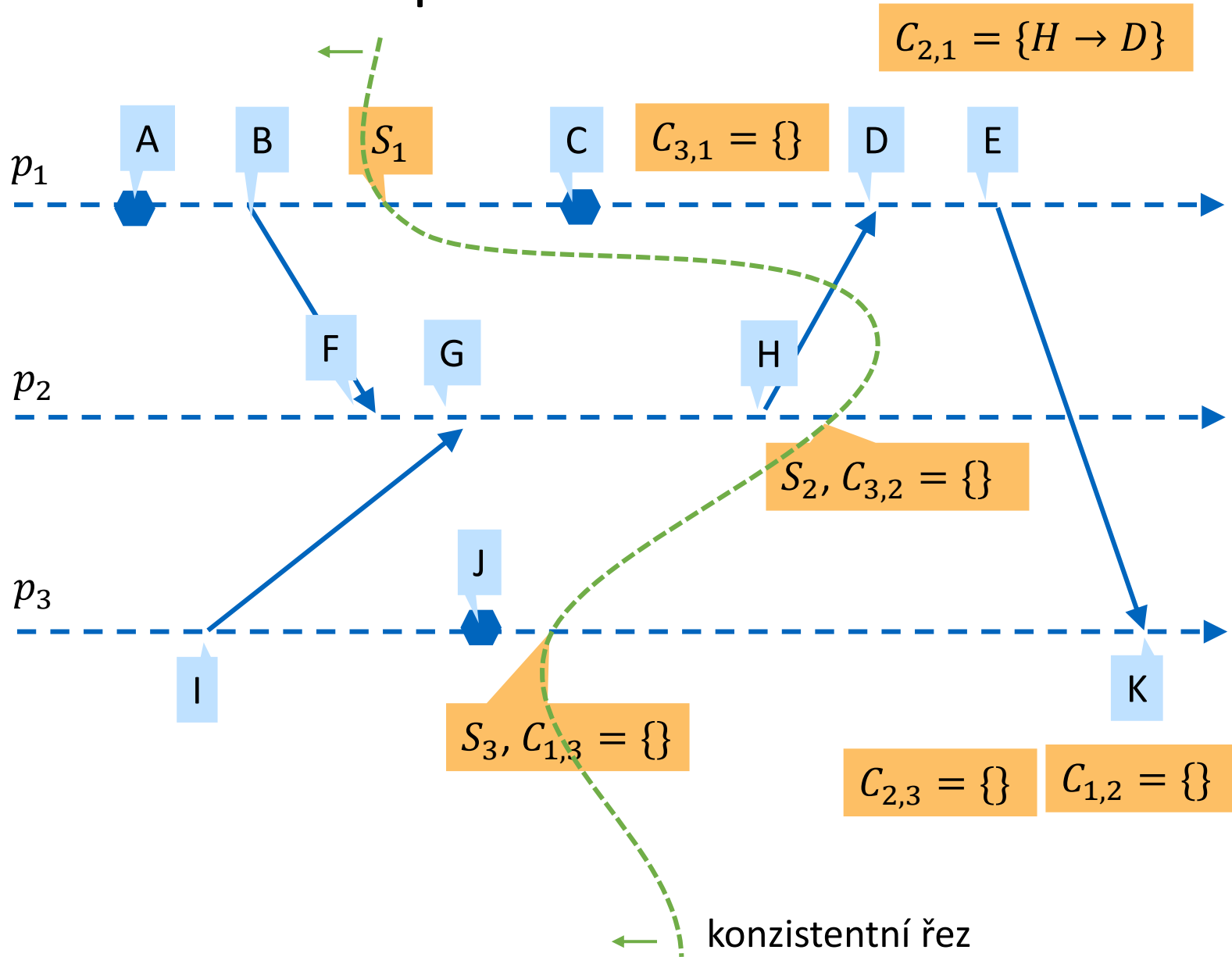








Finální Snapshot





Vlastnosti

Vlastnosti Chandy-Lamport

Výsledkem Chandy-Lamport algoritmu pro výpočet globální snapshotu je **konzistentní řez**.

Důkaz:

- necht' R je výsledný řez a e_i a e_j jsou události v procesech P_i a P_j tak, že $e_i \rightarrow e_j$
- pro konzistenci třeba dokázat implikaci: $e_j \in R \Rightarrow e_i \in R$
- tj. pokud $e_j \rightarrow \langle P_j \text{ zaznamená svůj stav} \rangle$ pak $e_i \rightarrow \langle P_i \text{ zaznamená svůj stav} \rangle$
- Předpokládejme, že tato implikace neplatí, tj. $e_j \rightarrow \langle P_j \text{ zaznamená svůj stav} \rangle$ a $\langle P_i \text{ zaznamená svůj stav} \rangle \rightarrow e_i$
- Uvažujme cestu aplikačních zpráv z e_i do e_j (přes další procesy)
- Vzhledem k FIFO kanálu musí ZNAČKY na všech spojích výše uvedené cesty předcházet aplikační zprávy
- Tedy, protože $\langle P_i \text{ zaznamená svůj stav} \rangle \rightarrow e_i$, tak musí platit, že P_j obdržel svoji ZNAČKU před e_j
- A tedy $e_j \notin R \rightarrow$ spor

Vyhodnocení stabilních vlastností

Uvažujme **vlastnost distribuovaného výpočtu** jako logickou formuli definovanou nad globálním stavem distribuovaného výpočtu.

Stabilní vlastnost je taková vlastnost, že jakmile je ve výpočtu **jednou** splněna, zůstává splněna **navždy**.

- příklad stabilní vlastnosti živosti: výpočet skončil
- příklad stabilní vlastnosti porušující bezpečnost: nastalo uváznutí, objekt je sirotek (neukazuje na něj žádná reference)

Chandy-Lamport algoritmus lze použít pro detekci **stabilních** globálních vlastností.

(Lze ukázat, že pokud nějaká stabilní vlastnost splněna v globálním snapshotu zachyceným snapshot algoritmem, bude splněna i výpočtu splněna i ve fyzickém okamžiku doběhnutí snapshotů algoritmu. Naopak, pokud ve vypočteném globálním snapshotu nějaká stabilní vlastnost splněna není, nemohla být ve výpočtu splněna v okamžiku zahájení snapshotů algoritmu).

Shrnutí

Schopnost **zachytit globální stav** distribuovaného výpočtu je důležitá.

Vytvoření snapshotů by nemělo nijak omezovat probíhající výpočet.

Chandy-Lamportův algoritmus vypočte globální snapshot.

Vypočtený globální snapshot odpovídá **konzistentnímu řezu**.

Globální snapshot může být využit k **detekci stabilních vlastností** výpočtu.