Univerzita Hradec Králové
Fakulta informatiky a managementu

Zablokování (Deadlock)

Mgr. Josef Horálek

FIM UHK

Deadlock

- V multiprogramovém prostředí si mohou různé prostředky konkurovat v získaní konečného počtu zdrojů
  - může se tedy stát, že čekající proces svůj stav již nikdy nezmění, protože zdroje na které čeká drží jiné čekající procesy;
  - tato situace se nazývána deadlock (zablokování);

Deadlock

- Ilustrace
  - Jestliže se k sobě vzájemně blíží dva vlaky na křížení kolejí, oba musí zcela zastavit a žádný se nesmí opětovně rozjet dokud druhy neodjede;

Model systému

- Systém obsahuje konečný počet zdrojů, které mohou být distribuovány mezi konkurující si procesy;
- Zdroje jsou rozdělené do tříd dle typu
  - každá třída obsahuje určitý počet identických instancí;
    - paměťový prostor;
    - cykly CPU;
    - soubory;
    - I/O zařízení;

Model systému

- Jestliže proces požaduje instanci určitého typu zdroje, jeho žádost uspokojí alokace jakékoliv instance daného typu
  - pokud tomu tak není, instance nejsou identické a třída zdroje není definována korektně;
- Proces musí o zdroj požádat před jeho užitím a musí ho po užití uvolnit
- Proces může požadovat tolik zdrojů, kolik jich vyžaduje uskutečnění zpracovávané úlohy.
  - počet požadovaných zdrojů většinou nepřekročí celkový počet zdrojů v systému;

FIM UHK

Model systému

- Při běžném vykonávání operace může proces užít zdroj pouze dle následující sekvence:
  - dotaz
  - užití
  - uvolnění
- Říkáme, že množina procesů je ve stavu zablokována, jestliže každý proces v množině čeká na událost, kterou může vyvolat pouze jiný proces z téže množiny
    - událost, se kterou se budeme hlavně zajímat je obsazení zdroje a jeho uvolnění

Model systému

- Deadlock může být vyvolán i různými typy zdrojů
  - např. systém s jednou tiskárnou a jednou magnetopáskovou mechanikou;
  - nechť proces Pi užívá magnetopáskovou mechaniku a proces Pj tiskárnu;
  - jestliže Pi požádá o tiskárnu a Pj o páskovou mechaniku nastává deadlock;

FIM UHK

Charakteristika deadlocku

- Deadlock je nežádoucí
  - procesy v deadlocku nemohou dokončit své spuštění;
  - systémové zdroje jsou obsazené;
  - ani jiné procesy nemohou tyto zdroje alokovat;
- Deadlock může nastat, jestliže jsou v systému současně splněny následující 4 podmínky:
  - vzájemná jedinečnost;
  - drží a čeká;
  - nepreemptivnost;
  - kruhové čekání;

FIM UHK

Charakteristika deadlocku

- Všechny podmínky pro vznik deadlocku musí nastat současně
  - podmínka kruhového čekání implikuje podmínku drž a čekej;
  - podmínky tedy nejsou na sobě navzájem nezávislé;

FIM UHK

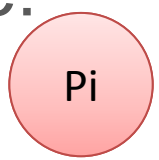
Charakteristika deadlocku

- Graf alokace zdrojů:
  - deadlock může být popsán pomocí orientovaného grafu (graf alokace zdrojů);
  - tento graf sestává z množiny hran H;
  - z množiny vrcholů V;
  - množina vrcholů sestává ze dvou podmnožin:
    - P = {P1, P2, ..., Pn} - množina všech procesů v systému;
    - R = {R1, R2, ..., Rm} - množina všech zdrojů v systému;

FIM UHK

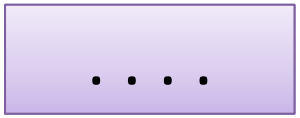
Charakteristika deadlocku

- Graficky prezentujme:
  - proces Pi jako kruh;



  - zdroj Rj jako obdélník;

  - protože každý zdroj Rj může obsahovat i více instancí, každou instanci definujme jako tečku uvnitř obdélníku;

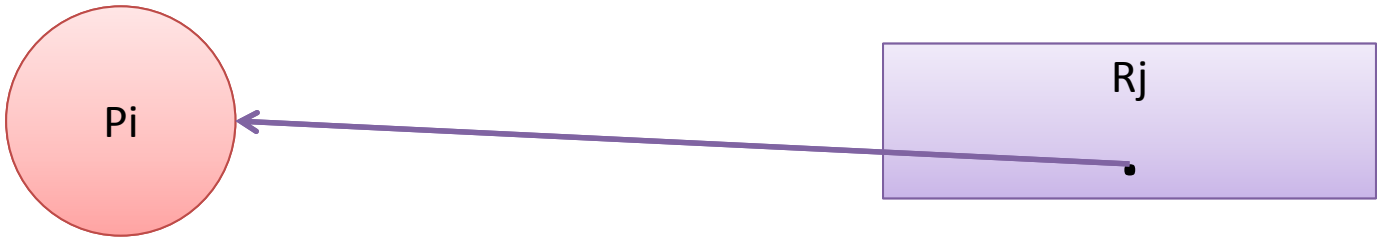


Charakteristika deadlocku

- Orientovaná hrana od procesu Pi ke zdroji Rj
  - znamená, že proces Pi požaduje instanci zdroje Rj a čeká na její přidělení;
  - nazývá se hrana dotazu;
    - hrana dotazu tedy ukazuje na celý obdélník Rj,

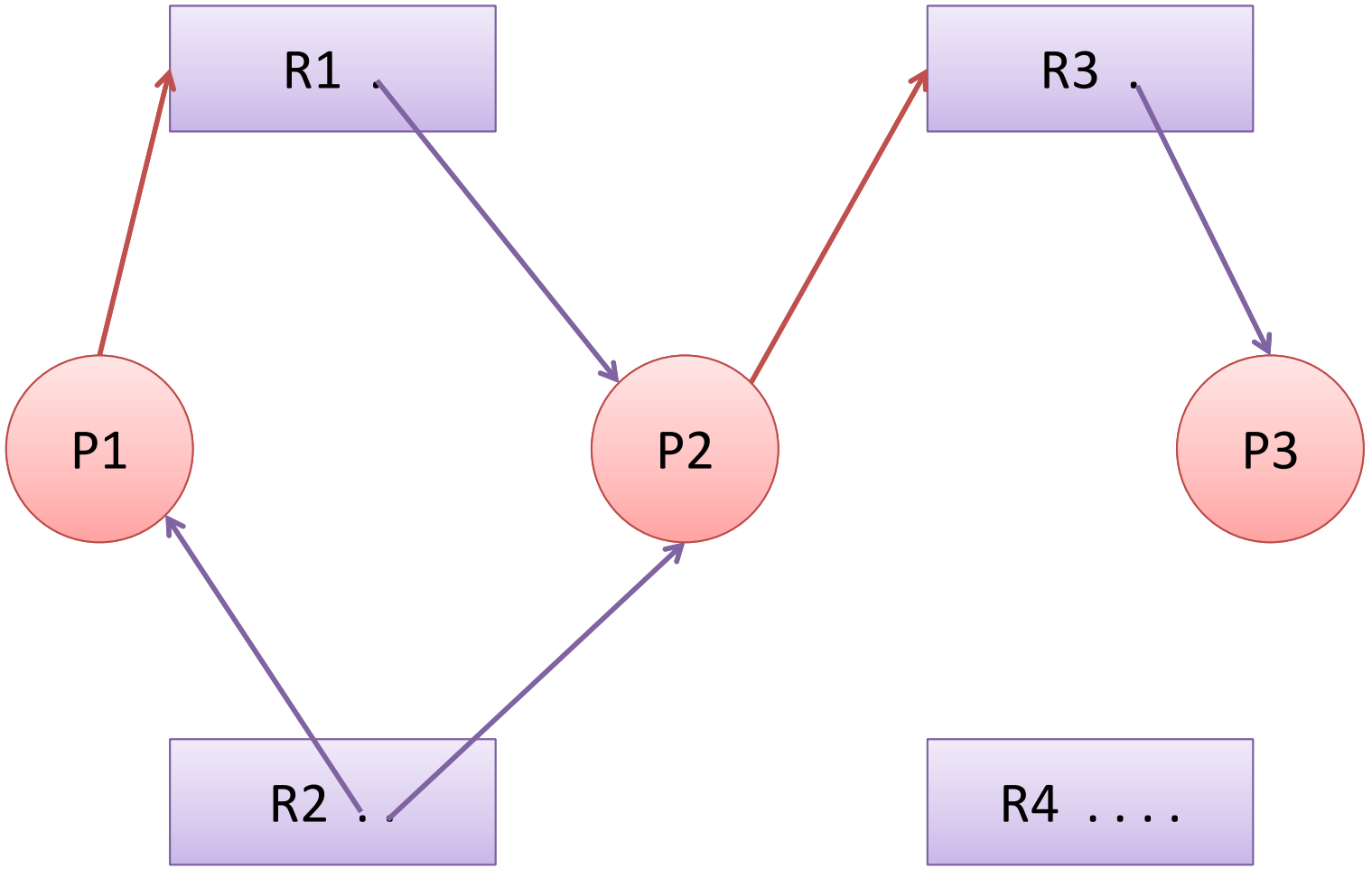
Charakteristika deadlocku

- Orientovaná hrana od zdroje Rj k procesu Pi
  - znamená, že instance zdroje Rj byla alokována procesu Pi;
  - nazývá se hrana přidělení;
  - hrana přidělení musí ukazovat na konkrétní instanci (tečku uvnitř obdélníku);



FIM UHK

Graf alokace zdrojů



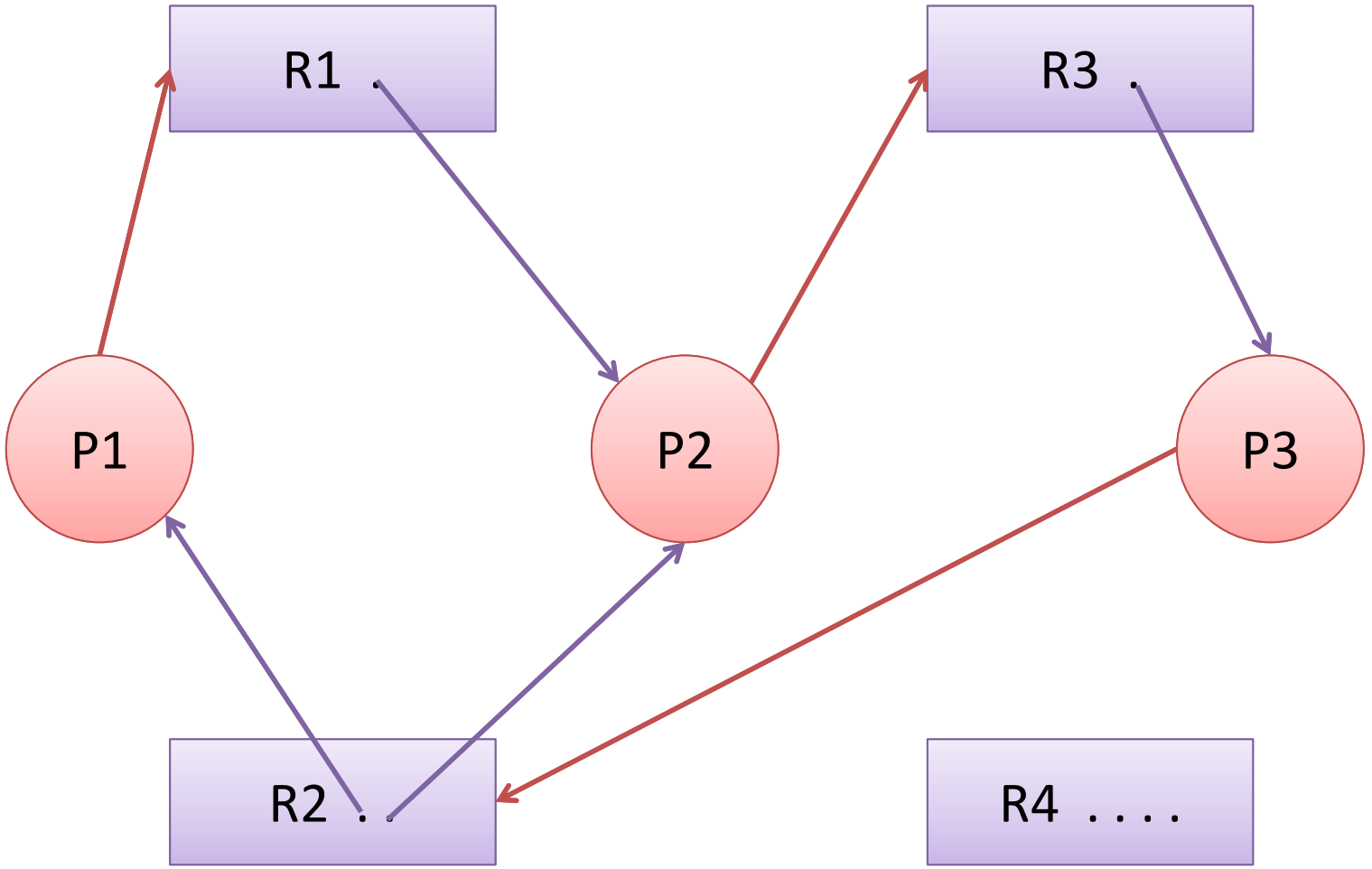
FIM UHK

Charakteristika deadlocku

- Pokud graf neobsahuje žádnou kružnici, není žádný proces zablokován
- Jestliže graf kružnice obsahuje, deadlock může nastat
  - pokud by každý typ zdroje obsahoval právě jednu instanci, kružnice v grafu by automaticky znamenala deadlock;
  - jestliže každý zdroj obsahuje několik instancí, potom kružnice v grafu nemusí nutně znamenat deadlock;

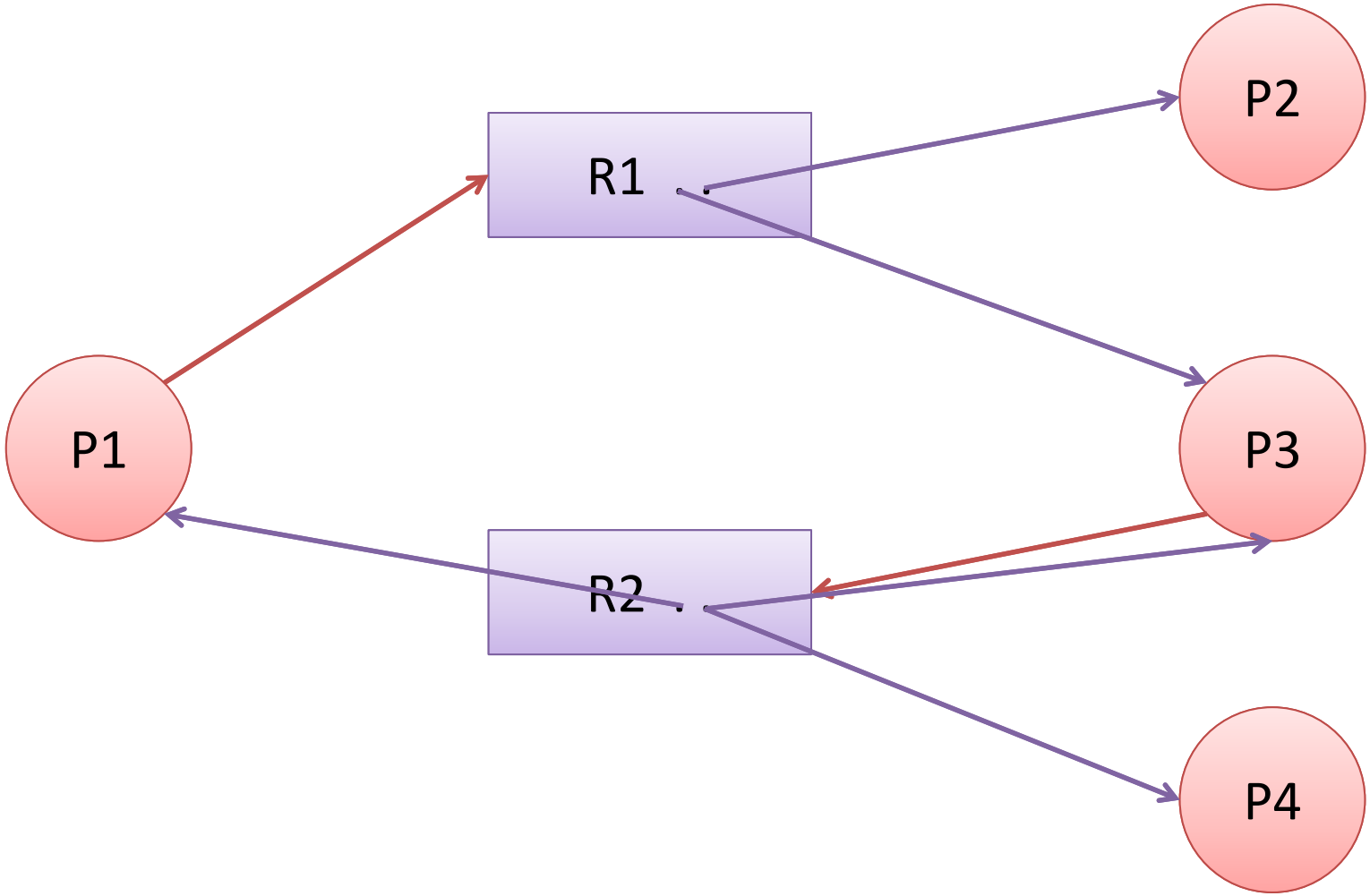
FIM UHK

Graf alokace zdrojů s deadlockem



FIM UHK

Graf alokace zdrojů s kružnicí bez deadlocku



FIM UHK

Charakteristika deadlocku

- Shrňme zjištěné poznatky:
  - pokud graf alokace zdrojů neobsahuje žádnou kružnici, potom v systému není žádný deadlock;
  - pokud v grafu kružnice je, v systému deadlock být;

Metody obsluhy deadlocku

- Principiálně existuji 3 metody řešení deadlocku:
  - můžeme v systému užít protokol, který zajistí, že nikdy deadlock nenastane;
  - můžeme systému dovolit, aby deadlock nastal a potom ho vyřešit;
  - můžeme celý problém deadlocku ignorovat a předstírat, že k němu nikdy nedojde;
    - toto řešení zatím užívá většina OS, včetně Unixu;

Charakteristika deadlocku

- K zajištění, že deadlock v systému nikdy nenastane může systém užít jedno z schémat:
  - deadlock-prevention
    - je množina metod, které zajišťují, že minimálně jedna z nutných podmínek nenastane;
    - tato metoda spočívá v omezení možnosti vytváření žádosti o zdroje;
  - deadlock-avoidance
    - požaduje, aby OS měl k dispozici další rozšiřující informace o zdrojích, které proces požaduje a používá;
      - systém může rozhodnout o bezpečném přidělení;
    - k bezpečnému vyřešení žádosti o přidělení zdroje potřebuje:
      - systémové informace o volných instancích zdroje;
      - systémové informace o instancích alokovaných jiným procesům;
      - systémové informace o budoucích žádostech;
      - systémové informace o instance tohoto zdroje;
      - systémové informace o budoucích uvolněních těchto instancí;

FIM UHK

Charakteristika deadlocku

- Nevyužívá-li systém schéma deadlock-prevention nebo schéma deadlock-avoidance obecně může dojít k deadlocku
- V takovémto prostředí může systém provádět detekci možného deadlocku nastane-li, pak ho vyřešit

FIM UHK

Charakteristika deadlocku

- Pokud systém nemá žádné zabezpečení proti deadlocku, může nastat situace, kdy se systém do deadlocku dostane a nemá žádný mechanismus k tomu, aby zjistil, co se stalo
  - nedetekovaný deadlock vede k snížení výkonu systému;
  - zdroje jsou drženy procesy, které nemohou být ukončeny;
    - další a další procesy, které nárokují blokované zařízení se dostávají do deadlocku;
    - nakonec může systém zcela přestat fungovat a vyžaduje manuální restartování;

Detekce deadlocku

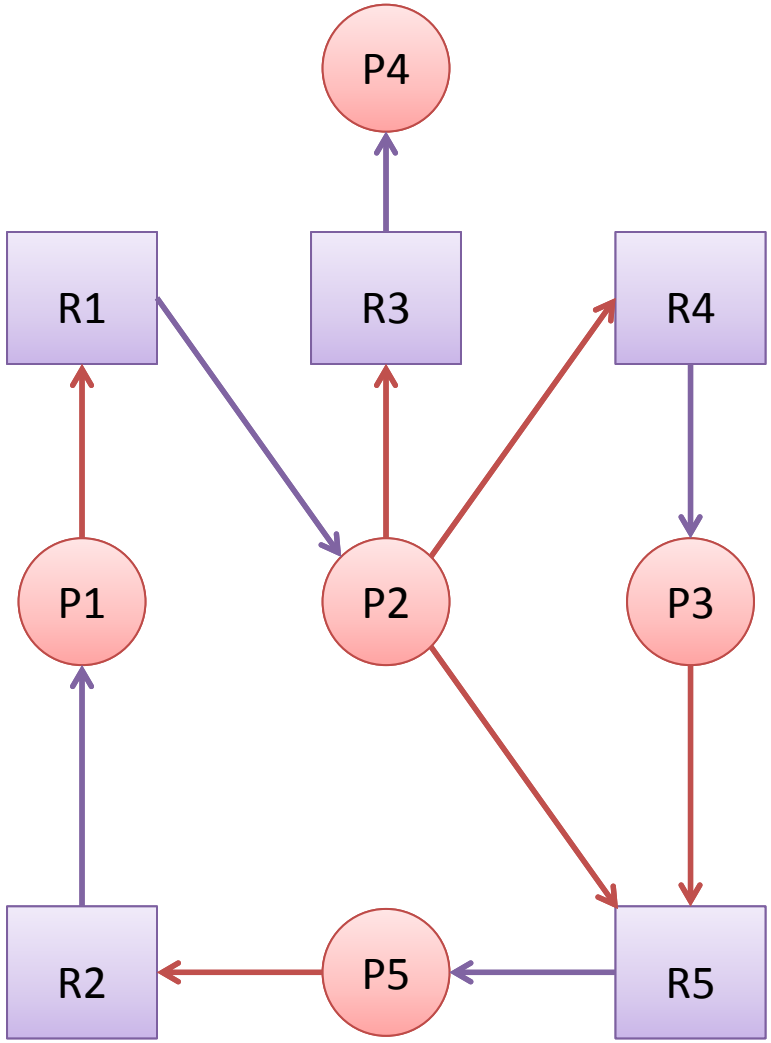
- Jestliže systém nevyužívá žádného mechanismu pro prevenci deadlocku, může deadlock v systému nastat
  - v takovém prostředí musí systém provádět:
    - algoritmus ohodnocující stav systému, který je schopen odhalit nastalý deadlock;
    - algoritmus, který nastalý deadlock vyřeší;
- Odhalení a vyřešení deadlocku vyžaduje režii
  - ta neobsahuje jen čas systému nutný k obsluze datových struktur a spuštění algoritmu detekce, ale také potenciální ztráty vzniklé při nápravě deadlocku;

Detekce deadlocku

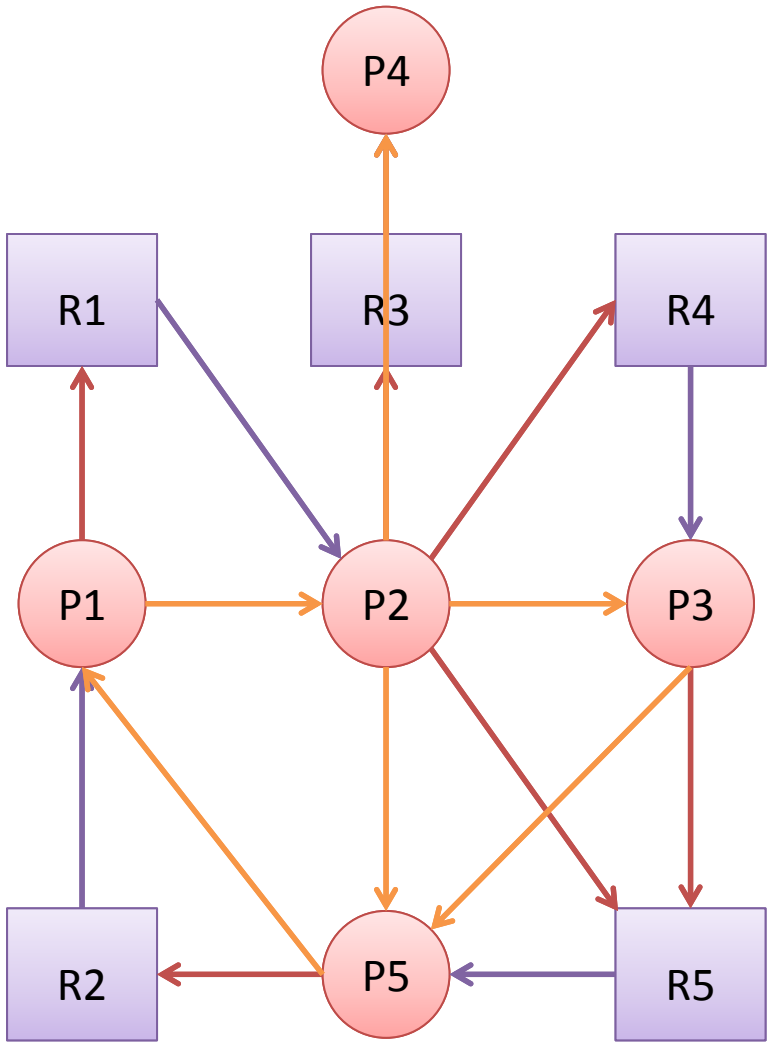
- Jedna instance v každé třídě zdroje:
  - jestliže všechny třídy zdrojů mají pouze jednu instanci
    - k detekci deadlocku použijeme speciální tvar grafu alokace zdrojů = **graf čekání**;
    - tento graf můžeme získat z grafu alokace zdrojů odebráním všech uzlů zdrojů a příslušných hran;

FIM UHK

Stav systému



Graf alokace zdrojů



Graf alokace čekání

FIM UHK

Detekce deadlocku

- Deadlock je v systému tehdy a jen tehdy, obsahuje-li adekvátní graf čekání kružnici
  - pro detekci deadlocku musí systém udržovat graf čekání;
  - musí periodicky spouštět algoritmus, který hledá v grafu kružnici;
    - algoritmus detekující kružnici v takovémto grafu vyžaduje sekvenci n^2 operací, kde n je počet vrcholů grafu;

Detekce deadlocku

- Užití algoritmu detekce deadlocku:
  - kdy má být spuštěn algoritmus detekce deadlocku?
    - odpověď je podmíněna dvěma faktory:
    - jak často k deadlocku pravděpodobně dochází?
    - kolik procesů bude deadlockem postiženo v případě, že nastane?
    - pokud k deadlocku dochází často:
      - algoritmus detekce je třeba spouštět často;
      - zdroje alokované procesy v deadlocku budou zablokované dokud nebude deadlock odstraněn;
      - počet procesů v deadlocku se může časem zvětšovat;

FIM UHK

Detekce deadlocku

- Deadlock může nastat pouze v případě, že některý proces vytvoří žádost, která nemůže být okamžitě vyřízena
- Je-li v systému mnoho různých tříd zdrojů, může jedna žádost vyvolat více kružnic v grafu alokace zdrojů
- Spouštění algoritmu detekce deadlocku po každé žádosti může vyvolat značné nároky na výpočetní čas

Detekce deadlocku

- Je-li algoritmus detekce spouštěn v libovolném čase
  - může v grafu alokace zdrojů existovat velké množství kružnic;
  - obecně pak nejsme schopni určit, který z množství zablokovaných procesů deadlock vyvolal;

FIM UHK

Náprava deadlocku

- Co když algoritmus detekce zjistí v systému přítomnost deadlocku ?
  - informovat správce systému, že nastal deadlock a nechat ho odstranit deadlock manuálně;
  - ponechat automaticky nápravu na systému;
    - automatická náprava může být provedena dvojím způsobem:
      - nejjednodušší možností je ukončit jeden nebo více cyklicky čekajících procesů;
      - druhá možnost je preemptivně ukončit alokaci nějakého zdroje deadlockovaným procesem;

Náprava deadlocku
Ukončení procesu

- K eliminaci deadlocku ukončením některého zablokovaného procesu je možno použít jednu ze dvou metod:
  - v obou případech systém uvolní a získá všechny zdroje alokované ukončovaným procesům;
  - ukončení všech deadlockovaných procesů;
    - ukončit všechny procesy v kruhu cyklického čekání je jednoduché, ale drahé;
    - výpočet procesu mohl trvat dlouho než se proces dostal do deadlocku, a celá tato výpočetní doba je ztracena a výpočet procesu musí začít znovu;
  - ukončení jednoho nebo více procesů, dokud není deadlock eliminovaný;
    - vyžaduje značnou režii
    - po každém ukončení procesu musí být spuštěn algoritmus detekce deadlocku, nachází-li se systém stále v deadlocku;.

Náprava deadlocku
Ukončení procesu

- Faktorů pro výběr procesu:
  - jakou má proces prioritu;
  - jak dlouho je již proces zpracováván systémem a jaká doba je ještě třeba k jeho dokončení;
  - kolik zdrojů a jakého typu má proces alokováno;
    - (nedají-li se tyto zdroje preemptivně uvolnit);
  - kolik dalších zdrojů bude proces ještě potřebovat ke svému dokončení;
  - kolik procesů je třeba ukončit;
  - je proces interaktivní nebo dávkový;

Náprava deadlocku
Preemptivní uvolnění zdroje

- Při eliminaci deadlocku preemptivním uvolněním zdroje:
  - postupně uvolňujeme zdroje alokované zablokovanými procesy;
  - přidělujeme je jiným, dokud není deadlock odstraněn;
  - chceme-li tuto metodu užít, je třeba vyřešit tři problémy:
    - vybrat oběť;
    - zabezpečení;
    - umoření;
      - často využívaná možnost je kombinovat výše zmíněné tři základní metody a pro každou třídu zdrojů užít optimální metodu;

Náprava deadlocku
Kombinovaný přístup k řešení

- K ilustraci popisovaného přístupu užijeme systém obsahující následující třídy zdrojů:
  - interní zdroje:
    - zdroje užívané systémem (např. process control block);
  - centrální paměť:
    - paměť vyhrazená pro uživatelské úlohy;
  - zdroje procesu:
    - alokované zdroje (např. pásky, tiskárny apod.) a soubory;
  - swapovací prostor:
    - úložný prostor pro uživatelské procesy;

Náprava deadlocku
Kombinovaný přístup k řešení

- Kombinované řešení deadlocku pro výše uvedené pořadí tříd zdrojů užívá následujících postupů pro jednotlivé třídy:
  - interní zdroje;
    - deadlock prevention popření podmínky cyklického čekání užitím uspořádání tříd zdrojů, nedochází k rozhodování mezi více žádostmi v jednom okamžiku;
  - centrální paměť;
    - deadlock prevention užitím preemptivní prevence, protože každá úloha může být vyswapována z paměti a ta může být preemptivně přidělena jiné úloze;
  - zdroje procesu;
    - deadlock avoidance, neboť potřebné rozšiřující informace je možno získat;
  - swapovací prostor;
    - předběžně přidělení (preallocation), protože maximální požadavky jsou předem známy;
    - tento příklad ilustruje, jak mohou být základní metody obsluhy deadlocku kombinovány v uspořádané struktuře zdrojů, aby bylo docíleno efektivního řešení deadlocku;

Univerzita Hradec Králové
Fakulta informatiky a managementu

Děkuji za pozornost…