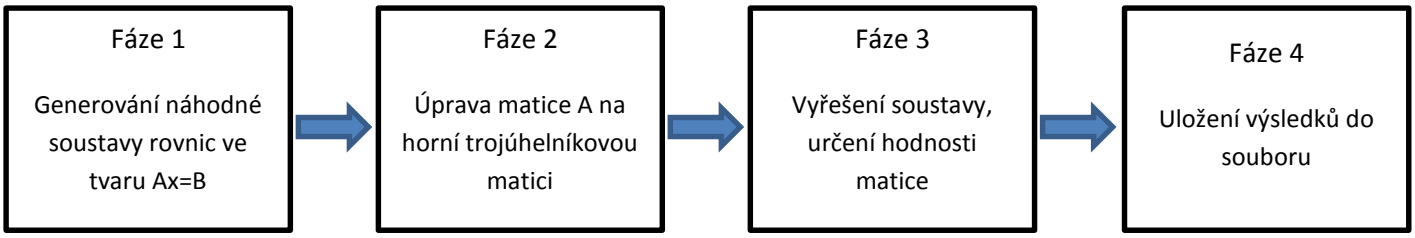
HW 1: Řešení soustav lineárních rovnic pomocí POSIXových vláken

Bodů: 6 Termín odevzdání: do čtvrtka 3. 11. 2015 (12:00)

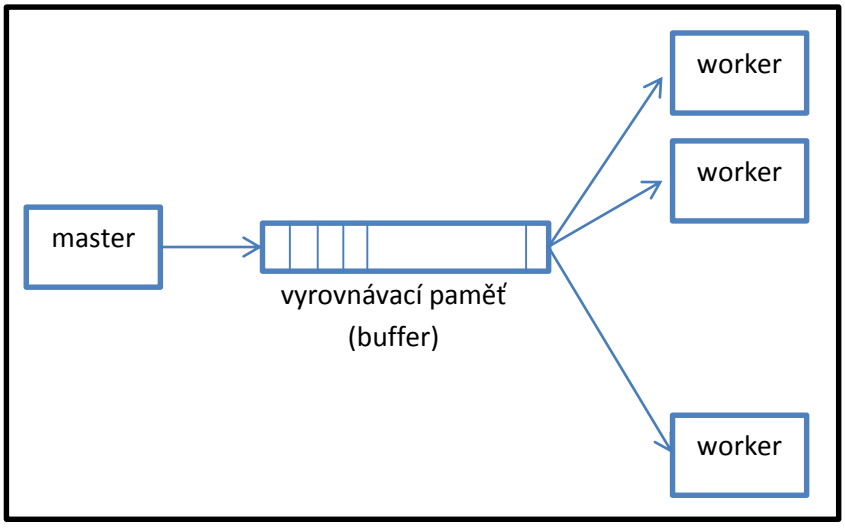
Implementujte řešení soustavy lineárních rovnic pomocí Gaussovy eliminační metody. Celý výpočet rozdělte do několika fází:

1. Generování náhodných čtvercových matic **A** a vektorů pravých stran **B**, které budou definovat rozšířenou matici příslušné soustavy lineárních rovnic ve tvaru Ax=B. Velikost (dimenze) matice bude dána konstantou MATRIX_DIM.
2. Úprava matice **A** na horní trojúhelníkovou (pod hlavní diagonálou obsahuje nulové prvky) včetně úpravy příslušných pravých stran tj. vektoru **B**.
3. Určení hodnosti matice **A** i rozšířené matice soustavy (**A**|**B**) a následné vyřešení soustavy rovnic pomocí zpětného dosazování.
4. Zápis výsledku do souboru (ukládejte koeficienty soustavy rovnic ve formě matice, hodnost matice **A** i rozšířené matice soustavy (**A**|**B**) a nalezené řešení soustavy).



Vzájemné propojení jednotlivých bloků je patrné z výše uvedeného obrázku. Fáze 1, 3 a 4 implementujte pomocí samostatných vláken. Fáze 2 bude implementována jako thread pool (viz obrázek níže), vlákno nazvané master bude přijímat od předchozího bloku náhodně vygenerované soustavy rovnic a bude je ukládat do vyrovnávací paměti, která bude mít omezenou velikost STAGE2_BUFFER_SIZE. Z vyrovnávací paměti si budou jednotlivá vlákna (worker) načítat matice pro zpracování. Počet vláken ve fázi 2 (bloky označené na obrázku jako worker) bude dán konstantou STAGE2_WORKERS_COUNT. Pro synchronizaci přístupu k vyrovnávací paměti použijte klasické synchronizační schéma producent-konzument, kde semafory full a empty nahradíte pomocí condition variables (budete tedy potřebovat 1x mutex a 2x condition variable).

Fáze 1 a 2 a fáze 3 a 4 budou propojeny přímo, tzn. nebude zde implementována vyrovnávací paměť a příslušná vlákna si budou předávat zpracovávané matice přímo. Pro synchronizaci opět použijte condition variables (princip je stejný jako u schéma produ-



cent-konzument, akorát zde není žádná vyrovnávací paměť, tj. pokud má již první blok připravena data k poslání a druhý blok je zaneprázdněn výpočtem, pak musí první blok čekat – být uspán a probuzen druhým blokem a obráceně, pokud druhý blok potřebuje data a první blok je ještě nemá připravená, tak musí také dojít k uspání druhého bloku s následným probuzením signálem v okamžiku kdy první blok bude mít data připravena k poslání). Fáze 2 a 3 budou propojeny přes vyrovnávací paměť, do které budou přímo zapisovat jednotlivá vlákna použitá ve fázi 2 tj. všechna vlákna označená jako worker.

Celou aplikaci navrhněte tak, aby po vygenerování daného počtu matic (bude dáno konstantou MATRIX_COUNT) a jejich zpracování došlo k automatickému ukončení celé aplikace (tzn. vlákno generující matice zašle ostatním vláknům v pipeline pokyn k ukončení). Součástí aplikace nechť jsou i kontrolní výpisy, ze kterých bude patrné, které vlákno zpracovává zrovna kterou matici (stačí vypisovat identifikátory matic – např. číslo udávající pořadí v jakém byla matice vygenerována).

Forma odevzdáni: emailem na adresu sloup@fel.cvut.cz se subjectem: "**A4M39GPU – HW1**".