Paralelní a distribuované algoritmy (PRL)
Mesh Multiplication

Daniel Konečný (xkonec75)
Vysoké učení technické v Brně
Fakulta informačních technologií

3. května 2021

1 Algoritmus Mesh Multiplication

Mesh Multiplication je paralelní algoritmus určený k násobení matic. Vstupem tohoto algoritmu jsou dvě matice $m_1 \times m_2$ a $n_1 \times n_2$, které musí splňovat základní požadavek $m_2 = n_1$. Značení velikost matic si tedy můžeme zjednodušit na $m \times k$ a $k \times n$, u kterého bude mít výsledná matice velikost $m \times n$. Právě takovou velikost je potřeba uvažovat i pro mřížku použitých procesorů pro výpočet. Každý procesor se totiž stará o jednu hodnotu výsledné matice.

Protože násobení matic není komutativní operace, musíme uvažovat i správné pořadí matic. Uvažujme mřížku $m \times n$ procesorů. Nalevo od ní si umístíme první z násobených matic, využijeme však navíc ještě posunutí jednotlivých řádků matice doleva tak, že první řádek není posunutý, druhý řádek je posunutý o jeden prvek doleva, třetí o dva prvky a tak dále. Vznikne nám tímto určitá schodovitá podoba matice. Analogický postup aplikujeme na druhou matici, která se umístí nad mřížku procesorů. První sloupec matice je nechaný na původním místě, druhý sloupec je posunutý o jeden prvek nahoru a podobně dále.

Dále je proveden určitý počet kroků (označme ho s, blíže popsaný bude v následující podsekci), ve kterých je vždy celá první matice posunuta o jeden prvek doprava a druhá matice o jeden prvek dolů. Tím se vždy dva prvky setkají na pozici určitého procesoru. Daný procesor provede násobení těchto prvků a přičte si je už k předchozím získaným součinům. Pokud na daném místě není prvek, procesor neprovádí žádnou operaci. Po k přičtených součinech na každém procesoru jsou zpracovány všechny prvky obou matic a mřížka výsledné součty v jednotlivých procesorech reprezentují matici $m \times n$ ve své výsledné podobě.

Složitost algoritmu

Uvažujme matice dříve uvedené, tedy o velikostech $m \times k$ a $k \times n$. Prostorová složitost algoritmu je zřejmá, jedná se o velikost výsledné matice, tedy:

$$p(m, n, k) = m * n.$$

Pro kompletní výpočet je potřeba provést s kroků, kde jeden krok je posunutí první matice doprava a druhé matice dolů vždy o jeden prvek. Počet kroků s je možné spočítat jako $k + max(m, n) - 1$. Tento počet kroků odpovídá časové složitosti algoritmu, tedy:

$$t(m, n, k) = k + max(m, n) - 1.$$

Cena algoritmu poté je následující:

$$\begin{aligned} c(m, n, k) &= t(m, n, k) * p(m, n, k) \\ &= m * n * (k + max(m, n) - 1) \\ &= m * n * k + m * n * max(m, n) - m * n. \end{aligned}$$

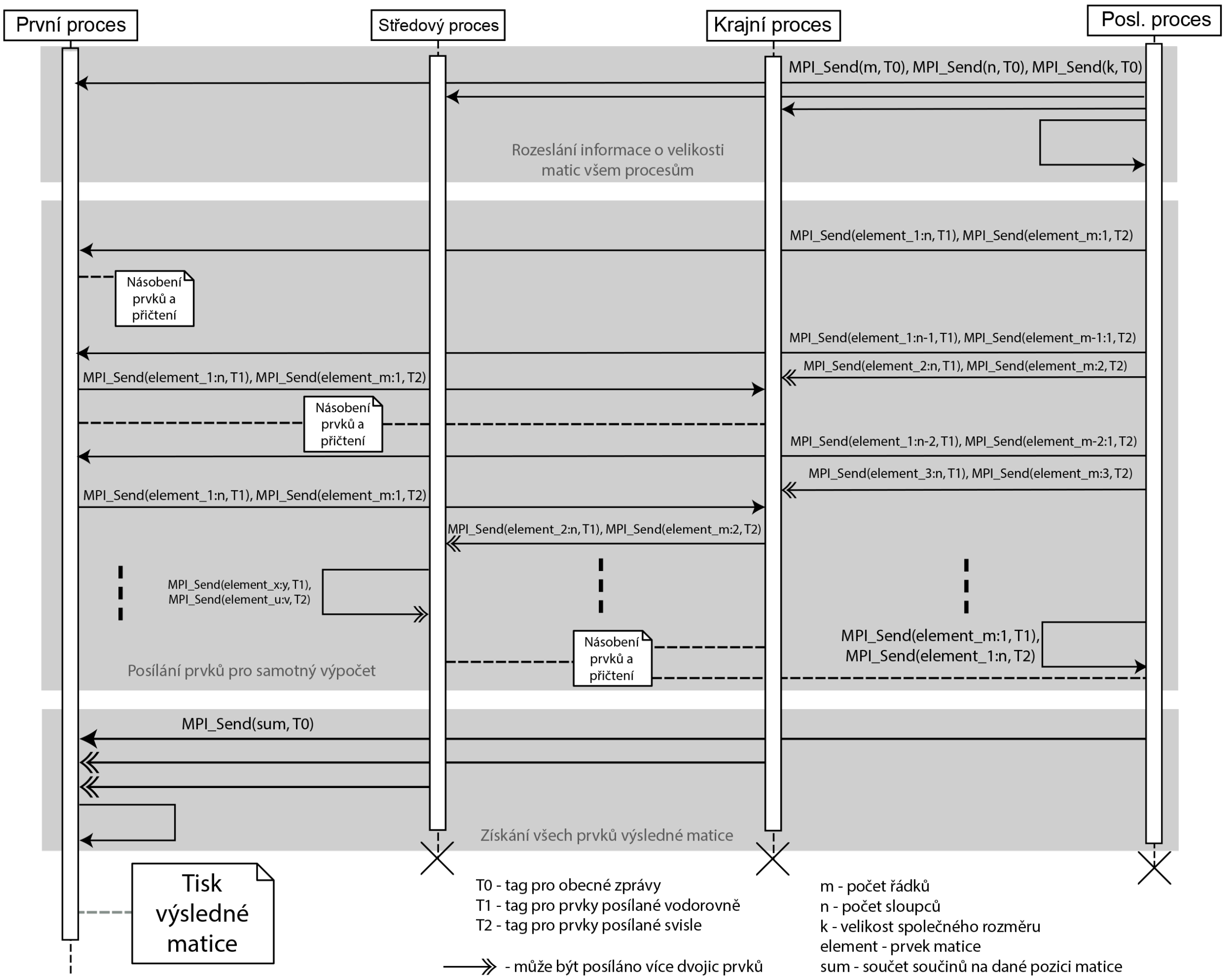
Pro jednoznačnější uvedení složitost algoritmu můžeme uvažovat čtvercové matice. Bude tedy dostatečná pouze jedna proměnná n.

$$p(n) = n * n = \mathcal{O}(n^2)$$

$$t(n) = n + n - 1 = \mathcal{O}(n)$$

$$c(n) = n * n * (2 * n - 1) = \mathcal{O}(n^3)$$

2 Komunikační protokol



3 Experimenty

Bylo provedeno 15 měření času výpočtu. Jako vstup byly použity matice velikostí $m \times 1$ a $1 \times m$, tedy vektory, o velikostech 1 až 15 prvků. Výsledkem je vždy matice $m \times m$. Jako měřící metrika byl využit nástroj `time` a součet naměřených časů `user` a `sys`, dále už jen $t(m)$.

4 Závěr

Algoritmus Mesh Multiplication byl plně naprogramován a dokáže násobit matice různých délek. Jeho teoretická časová složitost by měla být lineární. Dle experimentů však lze poznat, že se zvyšující se velikostí vstupu a tedy zvyšujícím se počtem potřebných procesorů, se doba běhu zvyšuje rychleji než lineárně. To bude patrně důvodem zvyšující se režie při přepínání kontextu, neboť stroj, na kterém byly experimenty prováděny, zvládne pouze 4 běžící vlákna. Jakmile počet potřebných procesorů překročil číslo 4, začala doba běhu narůstat výrazně rychleji.