Semestralni projekt MI-PDP 2019/2020:

Paralelni algoritmus pro reseni problemu

Zuzana Fílová

magisterske studium, FIT CVUT, Thakurova 9, 160 00 Praha 6

6. května 2020

1 Definice problému a popis sekvenčního algoritmu

Tato semestrální práce se zbývá řešením problému hledání minimálního hranového řezu grafu rozděleného na části velikosti a a n-a. Cílem je najít rozdělené množiny uzlů V do dvou disjunktních podmnožin X, Y takových, že množina X obsahuje a uzlů, množina Y obsahuje n-a uzlů a velikost hranového řezu mezi X a Y je minimální možná.

Vstupem algoritmu je graf G(V,E) = jednoduchý souvislý neorientovaný hranově ohodnocený graf o n uzlech a průměrném stupni k. Ohodnocení hran grafu jsou desetinná čísla z intervalu (0 … 1) a číslo a.

Výstupem je vektor 0 a 1, který pro každý vrchol určuje jeho přítomnost do množiny X nebo Y. Dále také součet ohodnocení všech hran hranového řezu mezi množinami X a Y.

Sekvenční algoritmus je založen na rekurzivním prohledávání do hloubky (DFS). Algoritmus systematicky přiřazuje uzly do množiny X/množiny Y. Po každém přiřazení spočítá aktuální hodnotu hranového řezu (mezi vrcholy již rozřazenými do množin X a Y nebo pro celý graf, pokud již má množina X zadaný počet prvků). Tato hodnota je porovnána s dosud nejlepším nalezeným řešením. V případě, že je aktuální hodnota horší než nejlepší řešení, dochází k ořezání stavového prostoru a prohledávání tímto směrem již neprobíhá. Prohledávání v dané větvi je ukončeno také v případě, že se naplní potřebný počet (a) prvků množiny X.

2 Popis paralelniho algoritmu a jeho implementace v OpenMP - taskovy paralelismus

Implementace task paralelismu je velmi podobná sekvenčnímu algoritmu pouze s několika málo rozdíly. Pomocí OpenMP direktiv jsou v kódu vytvořeny paralelní regiony, ve kterých je výpočet zpracováván pomocí více paralelně běžících vláken. Paralelní prohledávání je řešeno pomocí konstrukce task (#pragma omp task). Všechna vlákna pracují se sdílenou pamětí, proto je potřeba při aktualizaci globálních proměnných (v tomto případě nejlepšího výsledku) využívat kritickou sekci (#pragma omp critical), aby nedocházelo k časově závislým chybám. Program vrátí řešení, až poslední vlákno dokončí výpočet.

3 Popis paralelniho algoritmu a jeho implementace v OpenMP - datovy paralelismus

Dalším způsobem paralelismu je datový paralelismus. Jeho implementace je opět podobná sekvenčnímu algoritmu s několika rozdíly. Nejprve je z počátečního stavu vygenerována fronta stavů (která je poté převedena do vektoru). Vektor se dá rozdělit na nezávislé části, které mohou zpracovávat jednotlivá vlákna. Jednotlivé stavy (a navazující podstromy) z tohoto vektoru jsou poté paralelně zpracovávány v paralelním for cyklu (#pragma omp parallel for). Aktuální nejlepší řešení je aktualizováno v kritické sekci. Na konci paralelního cyklu je implicitně bariéra. Program vrátí řešení, až poslední vlákno dokončí výpočet.

4 Popis paralelniho algoritmu a jeho implementace v MPI

Knihovna MPI dovoluje práci s distribuovanou pamětí (využití více procesů). Vzhledem k neexistenci sdílené paměti musí sdílení informací mezi jednotlivými procesy probíhat pomocí zasílání zpráv. Procesy se dělí na jeden Master proces a Slave procesy. Master proces se stará o distribuci práce mezi Slave

procesy a průběžné hodnocení výsledku. Kód je rozdělen na část, kterou vykonává Master proces (má rank 0) a část, kterou vykonávají ostatní procesy. Master proces nejprve z počátečního stavu vytvoří frontu stavů, z níž postupně přiděluje práci Slave procesům. Slave procesy spočítají výsledek přidělené práce a pomocí zprávy jej vrací Master procesu. Ten získané mezivýsledky vyhodnotí a vrátí finální výsledek.

5 Naměřené výsledky a vyhodnocení

V následujících tabulkách a grafech jsou zobrazeny výsledky měření. Byla zkoumána doba výpočtu pro výše uvedené algoritmy (sekvenční řešení, task paralelismus, data paralelismus, MPI řešení). V případě MPI byly využity 3 procesy (2 Slave + 1 Master). Tabulky 1, 4, 3 a 2 ukazují naměřené časy pro různé typy výpočtu (paralelní/sekvenční) pro různé zadané soubory (mhr_??_??_?.txt). V tabulce 5 je srovnání výpočetních časů pro různé typy výpočtu u souboru mhr_37_15_17.txt.

V tabulce 6 a 7 jsou zobrazeny hodnoty paralelního zrychlení a efektivnosti vypočítané z naměřených hodnot při zpracování souboru mhr_34_10_17.txt jehož sekvenční zpracování trvalo 14 s. Paralelní zrychlení $S(n,p)$ je definováno jako $S(n,p) = \frac{SU(n)}{T(n,p)}$ a paralelní efektivnost $E(n,p)$ jako $E(n,p) = \frac{S(n,p)}{p}$.

Naměřené a vypočtené hodnoty odpovídají očekávanému chování a ukazují, že paralelní zpracování je vhodné pro řešení zadaného problému. V ideálním případě by bylo dosaženo lineárního zrychlení, to se nestalo, ale i tak jsou výsledky uspokojivé. Nejlepší zrychlení výpočtu nastalo při využití datového paralelismu.

6 Závěr

V rámci semestrální práce byl řešen problém minimálního hranového řezu. Bylo implementováno sekvenční řešení, task paralelismus a data paralelismus s OpenMP a také paralelní algoritmus v MPI. Ukázalo se, že využití paralelního výpočtu (ať už s pomocí task/data paralelismu OpenMP nebo MPI) je efektivnější než pouhé sekvenční řešení a lze pomocí něj zkrátit výpočetní čas potřebný k řešení úlohy.

7 Literatura

Cvičení a přednášky z předmětu MI-PDP.

soubor	čas (s)
mhr_20_10_5.txt	0.005
mhr_30_10_10.txt	1.324
mhr_30_10_15.txt	3.056
mhr_34_10_15.txt	8.987
mhr_34_10_17.txt	14.395
mhr_37_15_17.txt	460.304

Tabulka 1: Naměřené časy sekvenčního algoritmu

soubor	čas (s)
mhr_20_10_5.txt	0.025
mhr_30_10_10.txt	0.165
mhr_30_10_15.txt	0.468
mhr_34_10_15.txt	1.511
mhr_34_10_17.txt	1.687
mhr_37_15_17.txt	46.846

Tabulka 2: Naměřené časy paralelního algoritmu (task paralelismus)

soubor	čas (s)
mhr_20_10_5.txt	0.033
mhr_30_10_10.txt	0.277
mhr_30_10_15.txt	0.628
mhr_34_10_15.txt	1.729
mhr_34_10_17.txt	1.894
mhr_37_15_17.txt	38.124

Tabulka 3: Naměřené časy MPI

soubor	čas (s)
mhr_20_10_5.txt	0.021
mhr_30_10_10.txt	0.463
mhr_30_10_15.txt	0.297
mhr_34_10_15.txt	1.605
mhr_34_10_17.txt	1.136
mhr_37_15_17.txt	34.590

Tabulka 4: Naměřené časy paralelního algoritmu (data paralelismus)

typ výpočtu	čas (s)
sekvenční	460.304
task paralelismus	46.846
data paralelismus	34.590
MPI (6 vláken)	52.801
MPI (10 vláken)	38.124
MPI (20 vláken)	40.735

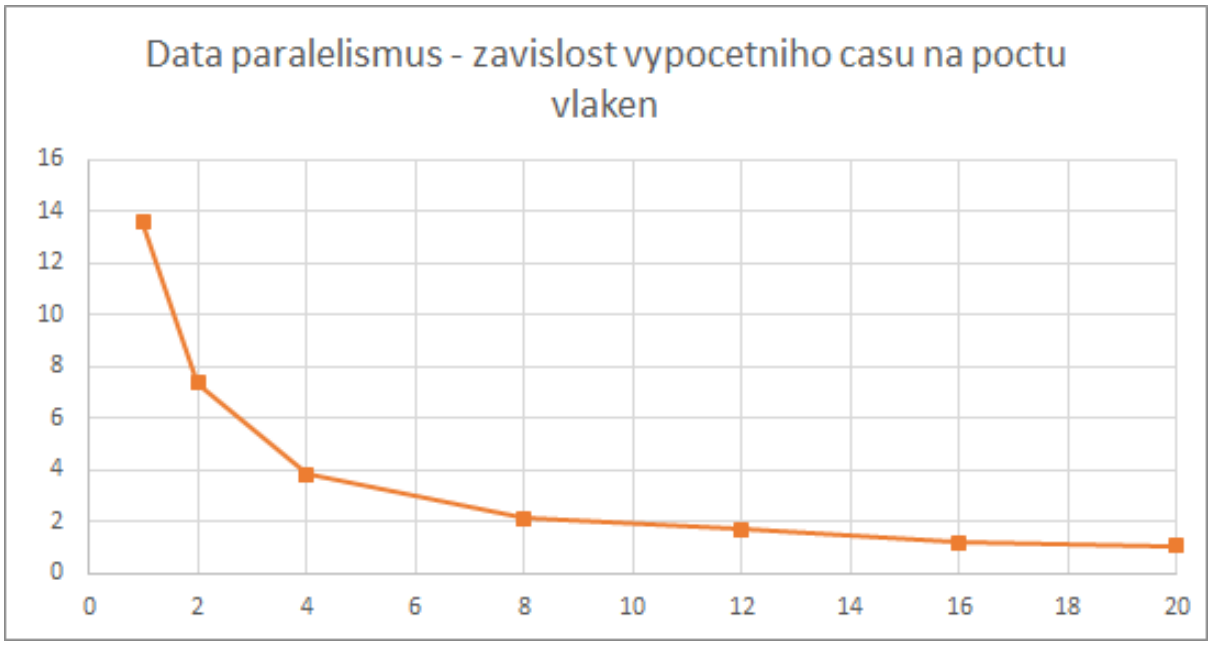
Tabulka 5: Naměřené časy pro různé typy výpočtu - mhr_37_15_17.txt

počet vláken	paralelní zrychlení S(n,p)	paralelní efektivnost E(n,p)
2	1.84	0.92
4	3.76	0.94
8	4.85	0.61
12	3.62	0.30
16	6.84	0.43
20	2.99	0.15

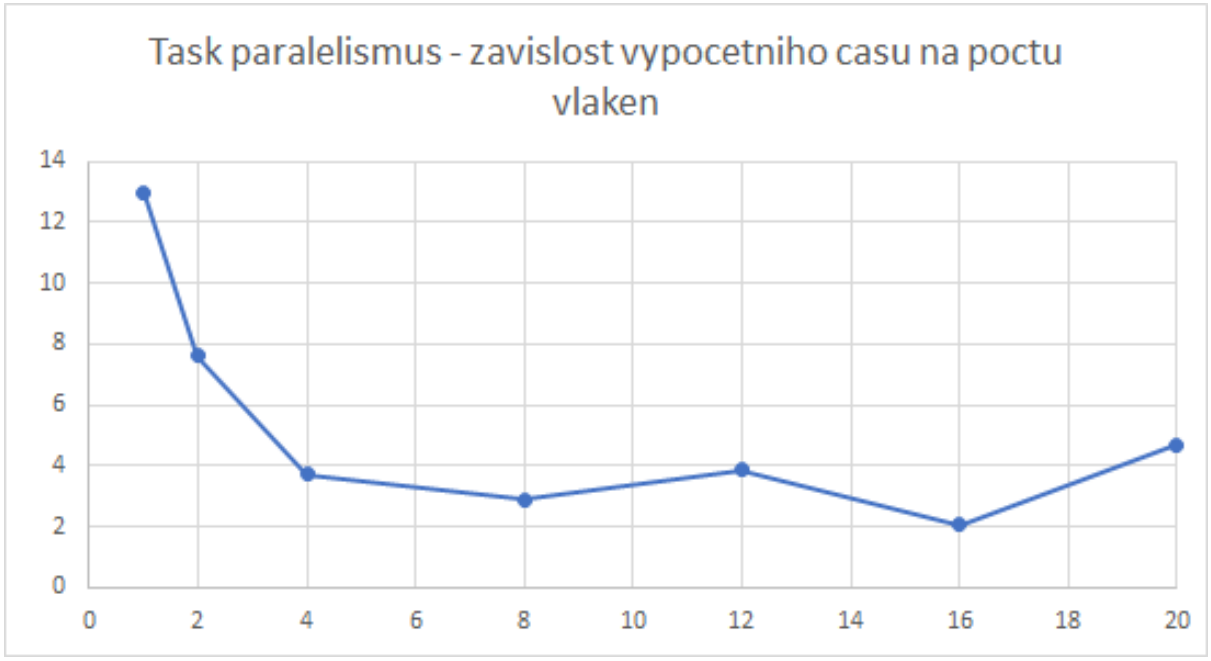
Tabulka 6: Paralelní zrychlení a efektivnost (task paralelismus)

počet vláken	paralelní zrychlení S(n,p)	paralelní efektivnost E(n,p)
2	1.89	0.95
4	3.65	0.91
8	6.47	0.81
12	8.24	0.69
16	11.54	0.72
20	13.12	0.66

Tabulka 7: Paralelní zrychlení a efektivnost (data paralelismus)



Obrázek 1: Data paralelismus - výpočetní čas (s) pro různý počet vláken (osa x) u souboru mhr_34_10_17.txt.



Obrázek 2: Task paralelismus - výpočetní čas (s) pro různý počet vláken (osa x) u souboru mhr_34_10_17.txt.