Semestrální projekt MI-PPR 2015/2016:

Paralelní algoritmus pro řešení problému Zobecněné bisekční šířky

Petr Elexa
Adam Kučera
magisterské studium, FIT ČVUT, Thákurova 9, 160 00 Praha 6

December 22, 2015

1 Definice problému a popis sekvenčního algoritmu

1.1 Zobecněná bisekční šířka

1.1.1 Vstupní data

- a = přirozené číslo
- n = přirozené číslo představující počet uzlů grafu G, $n \geq 5$
- m = přirozené číslo představující počet hran grafu G, $m \geq n$
- k = přirozené číslo řádu jednotek představující průměrný stupeň uzlu grafu G, $n \geq k \geq 3$ $G(V, E)$ = jednoduchý souvislý neorientovaný neohodnocený graf o n uzlech V a m hranách E (nepovinný, při jeho absenci je graf s danými parametry vygenerován).

1.1.2 Úkol

Nalezněte rozdělení množiny n uzlů grafu G do dvou disjunktních podmnožin X a Y tak, že podmnožina X obsahuje a uzlů, podmnožina Y obsahuje $n-a$ uzlů a počet všech hran $\{u, v\}$ takových, že u je z X a v je z Y, je minimální.

1.1.3 Výstup algoritmu

Výpis disjuktních množin uzlů X a Y a počet hran tyto množiny spojující.

1.1.4 Sekvenční algoritmus

Řešení existuje vždy. Vždy lze sestrojit zobecněný bisekční řez grafu. Sekvenční algoritmus je typu BB-DFS s hloubkou prohledávaného prostoru omezenou na $|a|$. Přípustný mezistav je definovaný rozdělením množiny uzlů na dvě disjunktní podmnožiny X a Y. Přípustná koncová řešení jsou všechna zkonstruovaná rozdělení množiny uzlů grafu G do množin X a Y. Cena, kterou minimalizujeme, je počet hran spojující X a Y.

Těsná dolní mez je rovna 1.

Triviální horní mez je rovna m.

1.2 Naše implementace problému

1.3 Formát výstupních dat

Algortitmus vypíše množinu X o velikosti a jako seznam identifikačních čísel uzlů podle toho, jak byly zadány ve vstupní matici grafu. Zároveň vypíše, kolik hran mezi množinou X a zbytkem grafu (množinou Y) existuje.

1.4 Experimentálně naměřená doba výpočtu

a	n	m	k	Naměřený čas
10	30	120	10	153,20 s
14	30	120	10	359,77 s

parA 10 parN 30 parM 120 parK 10 ... 5 min parA 10 parN 35 parM 140 parK 10 ... 15 min

2 Popis paralelního algoritmu a jeho implementace v MPI

2.1 Paralelizace sekvenčního řešení

Hlavním procesem řídící celý výpočet je proces s ID rovno 0 (p_0). Ten na začátku přečte vstupní parametry a případně vygeneruje graf, nad kterým

se bude počítat. Tento graf i parametry poté rozešle všem ostatním procesům. Poté jsou všechny procesy připraveny, všechna práce k vypočtení zatím zůstává procesu p_0.

Kostrou parelelního algoritmu je hlavní komunikační smyčka, která provádí výpočty a vždy po určitém počtu kroků zkontroluje, zda daný proces nedostal nějaké zprávy od jiných procesů. Tyto zprávy se používají k distribici práce k jednotlivým procesům a ke globálnímu ukončení výpočtu.

Pokud nějaký proces nemá práci (jako např. všechny procesy kromě p_0 na začátku), odešle žádost o práci jinému procesu. Ten pokud práci má, rozdělí svou práci a odešle žádajícímu procesu počateční a konečný prefix kombinací, které má počítat. V opačném případě odešle zprávu o chybě. Každý proces p_i úplně na začátku žádá o práci proces p_0, v každé další žádosti pak $p_{(i+1) \bmod p}$. Pokud žádost selže, obrátí se na další proces v pořadí.

Pokud nultému procesu dojde práce, rozešle tzv. End Token, pomocí kterého zjišuje, jestli ostatní procesy ještě pracují. Pokud se mu tento token vrátí jako TRUE, ukončí celý výpočet. Ostatní procesy token přijmou, nastavý na TRUE, pokud už nemají co počítat nebo na FALSE, pokud mají a přepošlou ho dalšímu procesu. Proces p_{p-1} posílá token zpět procesu p_0.

Jakmile proces p_0 detekuje, že všechny procesory již skončili, odešle tzv. Finish Token. Každý proces po jeho přijetí procesu p_0 pošle svou nalezenou minimální zobecněnou bisekční šířku a také množinu uzlů o velikosti a, pro kterou tuto šířku naměřil. Proces p_0 přijme všechny tyto zprávy a provede nad nimi minimální redukci, čímž se dostane ke globálnímu minimu, které poté vypíše na výstup a ukončí běh programu.

2.1.1 Konstanty a parametry paralelního výpočtu

3 Naměřené výsledky a vyhodnocení

seq problémy, co trvají 5, 10 a 15 minut

nechat je bezet na 2 až max procesorech

grafy zrychlení $S(n, p)$. superlinearni zrychleni?

Vyhodno?te komunika?ní složitost dynamického vyvažování zát?že a posu?te vhodnost vámi implementovaného algoritmu pro hledání dárce a d?lení zásobníku pri ?ešení vašeho problému. Posu?te efektivnost a škálovatelnost algoritmu. Popište nedostatky vaší implementace a navrhn?te zlepšení.

Empiricky stanovte granularitu vaší implementace, tj., stupe? paralelismu pro danou velikost ?ešeného problému. Stanovte kritéria pro stanovení mezí,

za kterými již není u?inné rozkládat výpo?et na menší procesy, protože by komunika?ní náklady převážily urychlení paralelním výpo?tem.

4 Závěr

Celkové zhodnocení semestrální práce a zkušenosti získaných b?hem semestru.

5 Literatura

Zobecněná bisekční šířka - implementace. Github [online]. [cit. 2015-12-20]. Dostupné z: https://github.com/Wrent/ppr$_z bs$