Paralelní algoritmy

2. ledna 2013

Obsah

1 Model PRAM **2**

1.1 Konflikty 2

1.2 Synchronizace 2

2 Složitosti **3**

2.1 Časová 3

2.2 Paměťová 3

3 Aktivace **3**

4 Instrukční sady **3**

5 Řešení konfliktů **3**

6 Sekvenční modely **4**

7 Paralelní algoritmy pro vybrané funkce **4**

7.1 Logaritmus 4

7.2 Nalezení maxima 4

7.3 Sčítání n-bitových čísel 4

7.4 Násobení 5

1 Model PRAM

Staví na modelu RAM. Ten je sekvenční, pevný program. Je paměť indexovaná číslem, a každá buňka obsahuje libovolně velké přirozené číslo.

PRAM funguje tak, že je mnoho RAMů, na začátku se vybere počet těchto RAMů, které se aktivují. Každý má své číslo (PID). Dále dostanou kromě vlastní paměti ještě globální (sdílenou) paměť, kterou smějí sledovat/ovlivňovat všichni.

1.1 Konflikty

Může se stát, že by došlo ke konfliktu při komunikaci s globální paměti.

Při čtení může být buď Exclusive read (nesmí se paralelně číst) nebo Concurrent read (smí se číst zároveň).

Při zápisu může být buď Exclusive write, Concurrent write je třeba ještě rozdělit.

Možnosti jsou prioritný (např. podaří se tomu s nejmenším PID), arbitrary (někdo z nich zapíše, neví se kdo), collision (zapíše se značka, že nastala kolize), common (smějí, ale musí zapisovat totéž) a W-PRAM (smějí zapisovat zároveň, ale všichni musí zapisovat číslo 1).

1.2 Synchronizace

- **SIMD** – shodné programy nad více dat. Je to jednodušší, ale problémy s podmínkami. Může se řešit tak, že má každý procesor bit, který říká, jestli zapisuje výsledky, nebo ne.
- **Uniformně** – mají stejné programy, za jeden tick jednu instrukci, ale programové čítače můžou být různé (tedy, každý může počítat jinak).
- **MIMD** – různé programy na různých procesorech.

Simulovat dolů jdou jednoduše.

Uniformní na SIMD modelovat tak, že v každém kole si každý procesor zkontroluje, že má zrovna simulovat tu a tu instrukci (tedy, napřed se kontroluje „simuluju první instrukci?“, potom „simuluju druhou?“, etc. . .), zpomalení podle délky kódu.

2 Složitosti

2.1 Časová

Potřebujeme říct, jak dlouhé jsou instrukce. Možnost buď jednotková a nebo logaritmická (tedy, podle počtu bitů), ale v tom případě je problém se synchronizací procesorů.

2.2 Paměťová

Těch je více. Jednak udává počet spotřebovaných registrů celkem (tedy, součet přes všechny procesory) – $S(n)$.

Potom tu je $P(n)$ – počet aktivovaných procesorů.

$W(n)$ je šířka slova – tedy, kolikabitová čísla budou stačit.

3 Aktivace

První možnost je, že se probudí všechny naráz. To je problematické.

Druhou možností je, že se probudí první a ten budí další (ty taky mohou budit).

4 Instrukční sady

Můžeme používat různě omezené instrukční sady (např. jen sčítání, odčítání a posun doleva o 1).

5 Řešení konfliktů

Pokud se chceme vyhnout konfliktům, tak lze postavit strom, kterým se to slučuje/rozděluje.

Toto dává logaritmické zpomalení.

Potřebujeme trochu rozdělit, co dělat, když každý zapisuje jinam, kde sehnat pomocné procesory, etc.

Lze vystačit jen s globální pamětí (tu rozstrkat).

6 Sekvenční modely

Různé turingovy stroje, Markovovy algoritmy, částečně rekurzivní funkce. Všechny stejně silné, rychlostí se liší jen polynomiálně (pokud jsou „rozumné“).

PRAM se dá simulovat na nich.

Všechny „rozumné“ paralelní modely se liší jen polynomiálně a jsou vázané na paměťovou složitost sekvenčních algoritmů.

Zatím je otevřené, jestli když je problém v polylog-space, pak by se dal paralelně vyřešit exponenciálně rychleji. Zatím se nevím, jestli každý problém z P je z polylog-space.

Třída NC je taková, kde k vyřešení v polylog čase a stačí k tomu polynomiální množství procesorů.

Pokud je zrychlení polylogaritmické a práce je stejně (tedy, počet procesorů $\times$ čas), pak je optimální. Efektivní je, pokud je práce nejvýše polylogaritmicky více. Toto je stejné ve všech rozumných modelech.

7 Paralelní algoritmy pro vybrané funkce

Jedním principem je postup „později a panuj“, druhý je „práce v týmech“ – každý tým dostane jedno potenciální řešení, on zkontroluje, zda to odpovídá.

7.1 Logaritmus

Princip týmů, jednočlenné, každý procesor ověří, jestli je jeho id správný výsledek.

7.2 Nalezení maxima

Každý tým o n procesorech zkouší, jestli tenhle je maximum, ten co je se zapíše.

7.3 Sčítání n-bitových čísel

Udělám $2^{n(b+\log n)}$ týmů, velikosti čísel zaokrouhlíme na mocniny dvojky. Kontroluje se, jestli to opravdu sedí (uhodnou jak výsledky, tak přenosy, kontrolují).

7.4 Násobení

Fungují podobně, jakoby pod sebou.

Lemma 1 (Parberry,Schnitger) *Nechť $T(n)$ je ws-konstruovatelná. Potom PRAM s omezenou sadou aritmetických instrukcí může simulovat k-páskový deterministických turingův stroj v konstantním čase, pokud ten běží v $O(T(n))$.*

Nagenerují se všechny posloupnosti a ověří.

Věta 1 *Nechť $B(n)$ je ws-kostruovatelná a $T(n)$ časově omezený TS. Potom se dá odsimulovat v čase $O\left(\frac{T(n)}{B(n)}\right)$ s šířkou slova $O(B(n) + \log T(n))$.*

Důsledek 1 *Libovolné polynomiální zrychlení je možné.*