Přiřazení pořadí preorder vrcholům

Jan Krejčí (xkrejc70)

1 Analýza algoritmu

Přiřazení pořadí preorder vrcholům v binárním stromě je speciálním případem průchodu Eulerovy cesty stromem, kdy se počítají pouze uzly, kterými se prochází skrze dopředné hrany. To jsou všechny hrany vedoucí od rodiče k synům, v pořadí otec, levý syn a pravý syn. Průchod začíná v kořeni a výsledný počet vrcholů na výstupu algoritmu je tak roven počtu dopředných hran + 1.

Vstupem je řetězec o n znacích, který reprezentuje uzly binárního stromu. Algoritmus tak k úspěšné paralelizaci vyžaduje $(2n - 2)$ procesorů, což odpovídá celkovému počtu hran v Eulerově cestě. Algoritmus se skládá ze čtyř na sebe navazujících částí, které jsou popsány ve zbylé části této sekce.

1.1 Reprezentace binárního stromu

Nejdříve je zapotřebí zpracovat vstup. To vyžaduje převod pole znaků do struktury reprezentující binární strom, se kterou půjde jednoduše pracovat. V této struktuře je potřeba uchovávat

- název uzlů a hran mezi nimi,
- informaci, zda-li je hrana dopředná či zpětná,
- případné následující hrany.

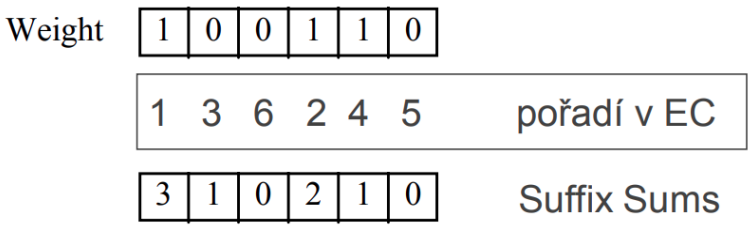
Z této reprezentace binárního stromu bude možné vytvořit Eulerovu cestu.

1.2 Eulerova cesta

Vytvoření Eulerovy cesty znamená vytvoření orientované kružnice v binárním stromě, která umožňuje obecně projít celý strom a všechny jeho hrany právě jednou. Toho docílíme nahrazením každé neorientované hrany mezi uzly dvěma orientovanými hranami, a to zpětnou/reverzní a dopřednou hranou. Vznikne tak Eulerovská kružnice, jejímž speciálním průchodem lze získat různé varianty přiřazení pořadí vrcholů jako například preorder, postorder či inorder.

1.3 Suma suffixů

V další části je každé hraně přiřazena její váha w na základě toho, zda se jedná o dopřednou $(w = 1)$ či zpětnou $(w = 0)$ hranu. Tyto váhy jsou vloženy do pole vah, nad kterým je v pořadí Eulerovské cesty získané v předchozím kroku vypočtena suma suffixů. Tento proces je proveden paralelně. Výsledkem je hodnota u každé z hran udávající počet dopředných hran zbývajících k průchodu stromem preorder. Samotné pořadí vrcholů při tomto průchodu je pak opačné, jak je uvedeno na obrázku 1.



Obrázek 1: Pole vah, suma suffixů a následné pořadí vrcholů při průchodu preorder. Převzato ze souhrnných materiálů PRL07-MNG ©Petr Hanáček, Model 2019.

1.4 Teoretická složitost

Díky paralelizaci je Eulerovská cesta, stejně jako inicializace pole vah ke každé hraně, vytvořena za konstantní čas

$$t(n) = O(c).$$

Suma suffixů se provede v logaritmickém čase

$$t(n) = O(log\ n).$$

Korekce, tedy přiřazení správného pořadí vrcholům z výsledku sumy suffixů, je provedena opět v konstantním čase

$$t(n) = O(c).$$

Celková teoretická časová složitost celého algoritmu pro přiřazení pořadí preorder vrcholům je

$$t(n) = O(log\ n).$$

A konečně, jak bylo zmíněno v sekci 1 Analýza algoritmu, pro úspěšnou paralelizaci algoritmu pro *n* uzlů binárního stromu je potřebný počet procesorů

$$p(n) = 2n - 2.$$

Celková cena celého algoritmu pro přiřazení pořadí preorder vrcholům je tak

$$c(n) = t(n) * p(n) = O(n\ log\ n).$$

2 Implementace

Algoritmus je implementován v jazyce C++ s využitím knihovny *Open MPI*, která zajišťuje paralelní komunikaci mezi procesory. Je využita pouze asynchronní blokující komunikace, a to pomocí funkcí `MPI_Send()` a `MPI_Recv()`.

Skript `test.sh` překládá a spouští výsledný program s konkrétním počtem procesorů, jak je popsáno v sekci 1 Analýza algoritmu. Pokud je vstupem pouze jeden znak, je program spuštěn s jedním procesorem, který jej vypíše. Také v sobě zahrnuje kontrolu argumentů.

Po inicializaci knihovny *Open MPI* pomocí `MPI_Init()` je vstup převeden do seznamu sousednosti (*adjacency list*), který je implementován jako vektor hran, obsahující všechny informace potřebné k budoucímu výpočtu, které byly popsány v sekci 1.1 Reprezentace binárního stromu. Struktura `Edge`, jež je datovým typem vektoru vah, obsahuje

- jednoznačný identifikátor hrany,
- zda-li je hrana dopředná,
- zda-li má případnou sousední hranu + její identifikátor,
- uzel, ze kterého orientovaná hrana směřuje,
- uzel, do kterého orientovaná hrana směřuje,
- identifikátor reverzní hrany.

Na takto reprezentovaný binární strom je následně použit algoritmus pro vytvoření Eulerovy cesty dle algoritmu v sekci 1.2 Eulerova cesta. Synchronizace dat mezi procesory zajišťují dvě hlavní funkce `sendToProc0()` a `recvFromAll()`, které jsou volány na několika místech v programu vždy současně. První z nich posílá data jedinému procesoru (v programu pojmenovaného `PROC0`), kde jsou data zasílána od zbylých procesorů. Druhou funkci naopak využívá jen procesor s označením `PROC0`, který sesbírá všechna data od ostatních procesorů. Jakmile jsou všechna data obdržena, jsou jako celek opět zaslána všem procesorům, aby mohly všechny pracovat nad stejnými daty.

Dále je vytvořeno pole vah, nad kterým je následně vypočtena suma suffixů, dle postupu uvedeného v sekci 1.3 Suma suffixů. Nakonec je provedena korekce a výsledné pořadí vrcholů včetně přidaného kořenového uzlu je posláno procesoru `PROC0`, který jej také vypíše na standardní výstup ve funkci `printOutput()`.

3 Závěr

Výsledkem je funkční program odpovídající paralelnímu algoritmu *přiřazení pořadí preorder vrcholům binární stromu* pro obecně *n* uzlů. Program byl řádně otestován a nejsou známy žádné implementační nedostatky.