Факултет техничких наука

Универзитет у Новом Саду

Рачунарски системи високих перформанси

Паралелизација проблема проналажења суме у бинарном стаблу

Аутор:
Бојан Попржен

Индекс:
E2 4/2022

11. јануар 2023.

Сажетак

У овом раду анализиран је проблем проналажења суме у бинарном стаблу. Проблем проналажења суме у бинарном стаблу можемо представити питањем: "Да ли чворови бинарног стабла којима је придружен природан број у било којој путањи од корена до листа дају задати збир?". Поступак за решавање овог проблема могуће је разложити на независне делове које је могуће паралелно решити. Решење проблема приказано у овом раду пружа приближно линеарно убрзање доласка до решења са порастом броја процесуирајућих јединица.

Садржај

Списак изворних кодова

Списак слика

Списак табела

1 Увод

У овом раду, анализиран је проблем проналажења суме у бинарном стаблу. Проблем можемо представити и питањем: "Да ли чворови бинарног стабла којима је придружен природан број у било којој путањи од корена до листа дају задати збир?".

Најчешће решење овог има асимптотску временску сложеност извршавања $O(n)$ gde n представља величину улаза tj. број чворова у графу.

У овом раду, предложено је паралелно решење проблема које узима у обзир независност обраде деце одређеног чвора.

Постигнута асимптотска сложеност паралелног решења је приближно: $O(n/p)$, где је n величина улаза тј. број чворова у графу, а p број јединица паралелног извршавања.

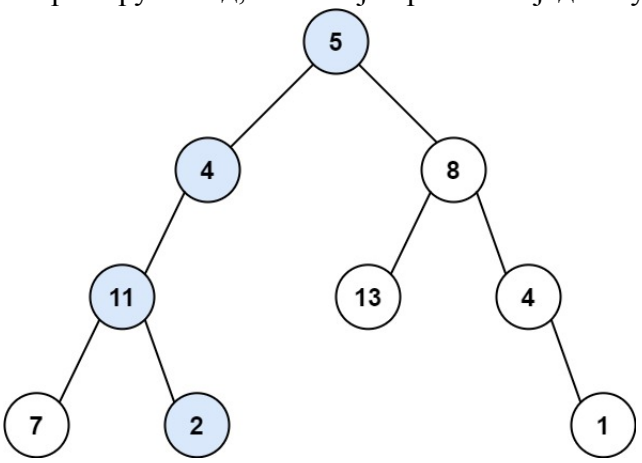
Рад је конципиран на следећи начин: прво ће сам проблем бити описан као и његово секвенцијално решење, потом ће бити приказано паралелно решење и , на крају, биће представљени подаци о ефикасности секвенцијалног и паралелног решења.

2 Проблем проналаска суме у бинарном стаблу

2.1 Опис проблема

Стабло је граф који је повезан и који нема циклуса. Бинарно стабло је усмерено стабло у коме сваки чвор, почевши од коренског чвора, има 0, 1 или 2 деце-чворова. Дететом чвора називамо чвор на кога неки други чвор показује. Родитељем чвора називамо чвор који на неки други чвор показује. Листом називамо чвор који има 0 деце.

У примеру испод, плавом је приказана једна путања која даје тражену суму 22.



За проблем су дата два улаза: стабло и тражена сума, док је тражен један излаз: булова променљива чија је тачност еквивалентна чињеници да је решење пронађено, односно да за дати граф постоји путања од корена до листа која даје тражену суму.

2.2 Секвенцијално решење проблема

Најчешће секвенцијално решење проблема своди се на обилазак графа по дубини (енг. *depth first search*, DFS). Овакво решење има асимптотску временску сложеност извршавања $O(n)$ gde n представља величину улаза tj. број елемената у графу.

Програмски код за секвенцијално решење дато је у листингу 1. Напомена: изостављен је део кода за учитавање и креирање графа.

```c
#include <math.h>
#include <stdint.h>
#include <stdio.h>
#include<unistd.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <sys/types.h>

// Definition for a binary tree TreeNode.
struct TreeNode {
    int val;
    struct TreeNode *left;
    struct TreeNode *right;
};

bool traverse(struct TreeNode* root, u_int64_t* sum) {
    if(root==NULL) return false;

    (*sum) -= root->val;
    if(root->right==NULL&&root->left==NULL&&*sum==0)
    {
        return true;
    }
    u_int64_t lsum, rsum;
    lsum = rsum = *sum;
    return traverse(root->left, &lsum) || traverse(root->right, &rsum);
}
```

Изворни код 1: Имплементација секвенцијалног решења у језику C

3 Паралелно решење проналаска суме у бинарном стаблу

3.1 Опис паралелног решења

Дефинишимо *дете-стабло* као стабло чији је корен дете датог чвора.

Секвенцијално решење не користи чињеницу да се "лево" и "десно" дете-стабло могу независно обрађивати. Међутим, у имплементационом смислу, оптимално решење није да се свако дете-стабло обрађује независно тј. у посебном ОпенМП задатку, јер тиме се оптерећује извршно окружење ОпенМП-а тј. доста процесорског времена се потроши на прекључивање контекста задатака уместо на ефективном раду и извршавању задатака.

Предлаже се паралелно решење које независно обрађује само стабла чији су корени на нивоу l, таквом да:

$$l = max_i 2^i \leq \text{omp\_get\_num\_threads}()$$

. Интуиција иза овога јесте да **паралелно** можемо да обрађујемо само онолико стабала колико имамо процесорских јединица на располагању. Стога, треба да паралелно обрађујемо стабла са кореном на оном нивоу стабла на ком је број чворова такав да већ следећи ниво стабла има више чворова него што програм има процесорске моћи.

У листингу 2 представљено је паралелно решење.

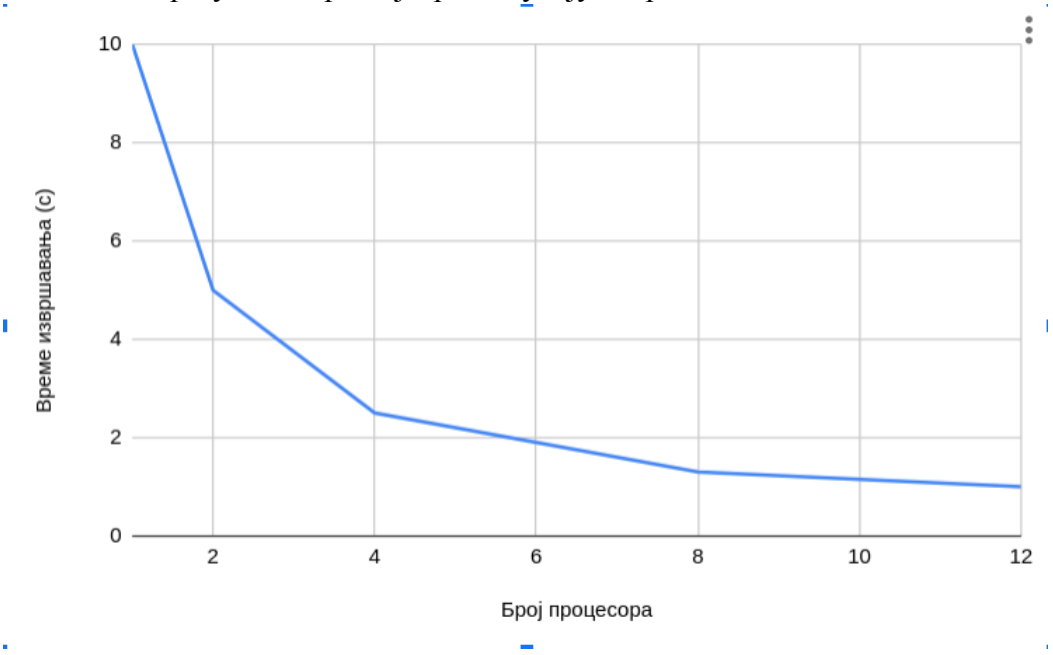
3.2 Сложеност паралелног решења

Асимптотска сложеност овог решења је приближно: $O(n/p)$, где је n величина улаза тј. број чворова у графу, а p број јединица паралелног извршавања.

```c
#include <math.h>
#include <stdint.h>
#include <stdio.h>
#include<unistd.h>
#include <omp.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <sys/types.h>

#define N 5
#define MAX_THREADS 1

// Definition for a binary tree TreeNode.
struct TreeNode {
    int val;
    struct TreeNode *left;
    struct TreeNode *right;
};

struct TreeNode* newTreeNode(int val)
{
    struct TreeNode* node = malloc(sizeof(struct TreeNode));
    node->val = val;
    node->left = node->right = NULL;
    return (node);
}

bool traverse(struct TreeNode* root, u_int64_t* sum)
{
    usleep(1);
    if(root==NULL) return false;

    (*sum) -= root->val;
    // printf("%d [%lu]\n", root->val, *sum);
    if(root->right==NULL&&root->left==NULL&&*sum==0)
    {
        printf("found!\n");
        return true;
    }
    u_int64_t lsum, rsum;
    lsum = rsum = *sum;
    return traverse(root->left, &lsum) || traverse(root->right, &rsum);
}

bool traverse_parallel(struct TreeNode* root, u_int64_t* sum, int level
{
    if(root==NULL) return false;
```

4 Експеримент

На слици испод могу да се виде резултати покретања решења са растућим бројем процесора. Покретање је вршено над графом величине 100 хиљада чворова, тражена сума је бирана тако да алгоритам никако не може да нађе решење. То је рађено како би се дошло до резултата при најгорем случају извршавања.



5 Закључак

У овом раду, анализиран је проблем проналажења суме у бинарном стаблу. Дат је приказ проблема и његовог секвенцијалног решења. Потом је приказано и његово паралелно решење. На крају, приказани су резултати који указују на линеарно убрзање доласка до решења са порастом броја процесуирајућих јединица.

Библиографија