Comparison of Structure of Arrays and Array of Structures

December 2019

Introduction

There are 2 ways of storing structures in memory. One is Structure of Arrays and the other is Array of Structures.

Example of SOA

```
struct pointlist3D {
    float x[N];
    float y[N];
    float z[N];
};

struct pointlist3D pointsSOA;
```

Example of AOS

```
struct point3D {
    float x;
    float y;
    float z;
};
struct point3D pointsAOS[N];
```

Both the formats of storing data have their pros and cons. The following C++ program was implemented to demonstrate their efficiency.

C+ code

```
#include <iostream>
#include <chrono>

using namespace std;
using namespace std::chrono;
```

```
#define N 1000000

struct pointlist3D {
    float x[N];
    float y[N];
    float z[N];
};

struct pointlist3D pointsSOA;


struct point3D {
    float x;
    float y;
    float z;
};
struct point3D pointsAOS[N];
/*
float get_point_xAOS(int i)
{
        return pointsAOS[i].x;
}
*/
int main()
{
        int x,y,z;
        float percent;
        for(int i=0; i<N ; i++)
        {
                pointsSOA.x[i] = i;
                pointsAOS[i].x = i;
        }
        printf("Accesing only coordinate x ");

        auto start = high_resolution_clock::now();

        for(int i=0; i<N ; i++)
                x = pointsAOS[i].x;
```

```
	auto stop = high_resolution_clock::now();

	auto duration = duration_cast<microseconds>(stop - start);
	cout << "\nTime taken for AOS : "<<duration.count();

	auto start2 = high_resolution_clock::now();

	for(int i=0; i<N ; i++)
		x = pointsSOA.x[i];

	auto stop2 = high_resolution_clock::now();

	auto duration2 = duration_cast<microseconds>(stop2 - start2);
	cout << "\nTime taken for SOA : "<<duration2.count();

	cout<<"\n";
	if(duration2 < duration)
	{
		cout<<"\nSOA is better when accesing only point x\n";
	}
	else
		cout<<"\nAOS is better when accesing only point x\n";


	printf("\nAccesing all coordinates  ");

	auto start3 = high_resolution_clock::now();

	for(int i=0; i<N ; i++)
	{
		x = pointsAOS[i].x;
		y = pointsAOS[i].y;
		z = pointsAOS[i].z;
	}

	auto stop3 = high_resolution_clock::now();

	auto duration3 = duration_cast<microseconds>(stop3 - start3);
	cout << "\nTime taken for AOS : "<<duration3.count();
```

```
        auto start4 = high_resolution_clock::now();

        for(int i=0; i<N ; i++)
        {
                x = pointsSOA.x[i];
                y = pointsSOA.y[i];
                z = pointsSOA.z[i];
        }

        auto stop4 = high_resolution_clock::now();

        auto duration4 = duration_cast<microseconds>(stop4 - start4);
        cout << "\nTime taken for SOA : "<<duration4.count();

        cout<<"\n";

        cout<<"\n";
        if(duration4 < duration3)
        {
                cout<<"\nSOA is better when accesing all coordinates\n";
        }
        else
                cout<<"\nAOS is better when accesing all coordinates\n";

}
```

Description

The structures used in the program above store the coordinates (x,y,z) of a point in 3-Dimensional space.

In the first part of the program, both the SOA and AOS are iterated over and only the x coordinate is accesed. The time taken for this operation in both the cases is noted.

The chrono library is used for obtaining the timestamps.

In the second part of the program, both the SOA and AOS are iterated over and all the 3 cooridinates are accesed. The time taken for this operation in both the cases is noted.

Result

The following are some of the results obtained when the above code is run.
Note : The number of points used is 1 million.

Eg output1

Accesing only coordinate x
Time taken for AOS : 2570
Time taken for SOA : 2292

SOA is better when accesing only point x

Accesing all coordinates
Time taken for AOS : 5825
Time taken for SOA : 3774

SOA is better when accesing all coordinates

Eg output2

Accesing only coordinate x
Time taken for AOS : 3312
Time taken for SOA : 2576

SOA is better when accesing only point x

Accesing all coordinates
Time taken for AOS : 5109
Time taken for SOA : 3191

SOA is better when accesing all coordinates

Eg output3

Accesing only coordinate x
Time taken for AOS : 2574
Time taken for SOA : 3951

AOS is better when accesing only point x

Accesing all coordinates
Time taken for AOS : 7863
Time taken for SOA : 4606

SOA is better when accesing all coordinates

Eg output4

Accesing only coordinate x
Time taken for AOS : 9171
Time taken for SOA : 7399

SOA is better when accesing only point x

Accesing all coordinates
Time taken for AOS : 10274
Time taken for SOA : 4902

SOA is better when accesing all coordinates

Observation

It was observered that SOA is outperforms AOS in most cases. Since the number of elements per structure is quite small (just 3 integers) the differences between the 1st case and the 2nd do not seem to affect the output charecterisitics.

Report by:

Arjun Goel