Parallel Implementation of Ray–tracing on Shared Memory Architecture

Vlastimil Havran, havran@cs.felk.cvut.cz
CTU Prague, Dept. of Computer Science

1 Introduction

The ray–tracing has become very popular method in computer graphics. This method of the visualization of the artificial and complex scene enables the sufficient grade of fidelity the result picture. The main disadvantage is the time complexity of the rendering algorithm caused by the calculation of the intersections the ray with the objects in the scene. In order to speed up the computation is used the parallelization of the computation task with all the advantages and problems rising of it.

2 Implementation

The ray-tracer used for parallelization was chosen the product developed in the Department of Computer Science in CTU Prague. It enables to process the scene description in Inventor format, which has become the standard for scene description in wide range of application in computer graphics on Unix platform. For the parallelizaton has been chosen the computer with shared memory architecture in Supercomputing center in Prague, the Power Challenge with six processors R8000 available.

The goal of my work was to implement five strategies to the parallelization and to compare their results of the different implementation. The problem is to design the strategy, which utilizes the multiprocessor system up to the maximal extent. There are two basic approaches to this problem. The first on consists in the division of the image into parts and each processor computes one part. The second solution is based on the subdivision scene space onto the cells assigned to computation units. For my implementation was chosen the image space subdivision approach, which is to be more efficient by the principle and more easily implementable. For the parallelization on the shared memory machine was used the library SHM developed by me especially for purposes of the parallelization of the graphical application on the shared memory computers. It is based on the IPC library from AT&T. This study takes into account not only number of the processors available on the computer, but also the simulation of the more processors by processes.

The problem of parallel implementation is, that the computation of one pixel can consume completely different time because of the different complexity of objects in the space. The solution consists in the assignment the different number of the pixels to the process in order to balance the load among processors. The second requirement on the solution is to preserve the proximity of pixels computed in succession.

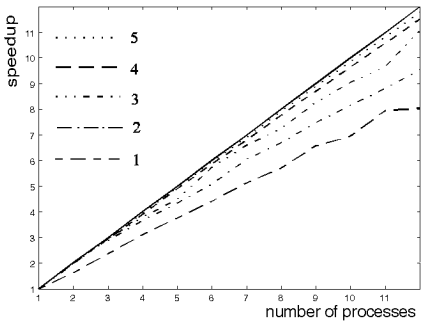
This property causes better usage of the caches, because it is more probable the data remains in the cache and the computation time is smaller, although the computational complexity remains the same.

There are some methods for distribution the computation load among the processors. Let us suppose, that the number of the processes running is n.

1. The image is divided into the n bands. Each process computes one band. The load distribution is not uniform.
2. The image is divided into the $i \times j$ rectangles. Each rectangular area is divided onto the n rectangles, which has to be the same area and should be rectangular as much as possible.
3. The image is divided onto the $i \times j$ rectangles. They are assigned to the processors on demand one by one.
4. Each process computes the color of n-th so to cover the whole image.
5. Each process computes the color of the n-th pixel on the row, then passes to the new row.

3 Results

The computation time for one process (it means also one processor) is 783.2[s] for fractal scene composed from 7381 reflected spheres. The graph lines show speedup for different approaches measured in user time of unix processes. This enables the simulation of more processors on physically six configured processors. The real speedup measured in real time did not exceed 5.84.



4 Conclusion

In my work were verified the schemes of the parallelization of ray–tracing algorithm designed originally for non-shared memory machine by Paddon[2]. The results of measurement show that for such small number of processors available suprisingly the best strategy is to assign each processor n-th row of the image provided the number of rows of result image is much more smaller than the number of processors. For number of processors about 40 or more the situation will probably change, but this situation cannot be enough faithfully simulated. The processor cache is utilized by more processes, although on real more processor machine each process has its own cache. It can significantly influence the results, but I suppose the dynamic scheme will be the best one.

References

[1] Havran, V.: *The simulation of the optical effects.* Master Thesis, CTU Prague, Czech Republic, 1996.

[2] Paddon, D.J.: *Parallel Processing for Computer Graphics.* Research Monographs in Parallel and Distributed Computing, Pitman, 1993.