Parallel Communication for Animating Fluids

J. K. Loots
Academy for Information Technology
University of Johannesburg
james.loots@gmail.com

A. Hardy
Academy for Information Technology
University of Johannesburg
alexandre.hardy@gmail.com

Problem

A general rule for a parallel application to benefit from parallelisation is to ensure that the time spent communicating is significantly outweighed by the time spent performing calculations. The time spent doing communication for a parallel fluid solver however, decreases speedup and in some cases even results in a slow down. This is due to the fact that the data areas needed are divided between the processes and need to be continually updated.

The main communication cost is encountered during the stages of the algorithm that require iterative solving. This is due to the fact that one needs to communicate the border areas during every iteration of every iterative solver used in the calculation of the frame. There are four iterative solvers used, during the advection phase of every advected vector field (for *x*,*y* and *density*) as well as the projection step.

The cost for communication due to these solvers can be huge when using a parallel computer which communicates using a network. Even on parallel computers that communicate through shared memory the communication has an impact. One way to reduce this problem is to ensure that the time taken in actual computation dominates the time taken for communication. This would restrict us to large data sets. However we want to be able to take advantage of parallelisation for smaller data sets too, due to the fact that even these small data sets take a relatively long time to compute.

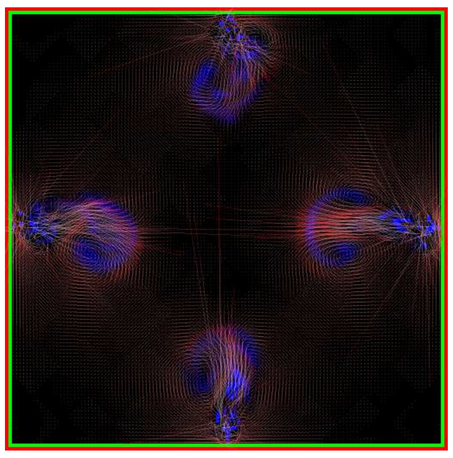

Figure 1: A logical view of the data area during a simulation step

Figure 2: The data area for each of the 4 processes in a 4 process simulation

Solution

Our solution is to remove the requirement for the parallel application to strictly adhere to the full algorithm used by the single process application. This can be done as we are exploring fluid simulations in an animation environment where strict physical accuracy is not required. This is accomplished by removing the requirement to communicate after every iteration of the iterative solvers and only communicate every 5 iterations. This means the the data area each processor is responsible for is updated every iteration, but the boundaries between the data areas of processes are only updated after a couple of iterations.

Results

The following two tables contain the runtime and speedup for two algorithms. The parallel computer used was a workstation cluster containing 16 3.2 GHz Pentium 4 processors connected using a 100Mbit Ethernet switch. The Ethernet connection makes communication very important as the communication time is slower than a commercial parallel interconnect. Table 1(a) contains information from a application that communicated after every iteration of the solvers and table 1(b) contains information from a application that only communicated every fifth iteration of the solvers. The column headings represent the grid size used, while the rows represent the time taken and speedup obtained for every frame generated. The times for simulations that obtained speedups are printed in bold face.

		64x64	128x128	256x256	512x512
1 Processor	Time per frame	0.196s	0.765s	3.363s	13.494s
	Speedup	–	–	–	–
2 Processors	Time per Frame	0.786s	1.480s	3.383s	**10.215s**
	Speedup	0.249	0.517	0.994	**1.321**
4 Processors	Time per Frame	1.151s	1.942s	3.903s	**9.793s**
	Speedup	0.170	0.394	0.862	**1.378**
8 Processors	Time per Frame	1.356s	2.211s	4.583s	**11.460s**
	Speedup	0.145	0.346	0.734	**1.177**
16 Processors	Time per Frame	1.509s	2.563s	5.540s	17.441s
	Speedup	0.130	0.298	0.607	0.774

(a) Full Communications

		64x64	128x128	256x256	512x512
1 Processor	Time per frame	0.194s	0.760s	3.144s	13.176s
	Speedup	–	–	–	–
2 Processors	Time per Frame	0.348s	0.761s	**2.258s**	**8.3207s**
	Speedup	0.557	0.998	**1.392**	**1.583**
4 Processors	Time per Frame	0.471s	0.876s	**2.206s**	**7.247s**
	Speedup	0.411	0.867	**1.425**	**1.818**
8 Processors	Time per Frame	0.752s	1.189s	**2.988s**	**8.980s**
	Speedup	0.257	0.639	**1.052**	**1.467**
16 Processors	Time per Frame	0.944s	1.531s	3.863s	14.944s
	Speedup	0.205	0.496	0.813	0.881

(b) 1 in 5 Solver Communications

Table 1: Time per Frame and Speedup for the Algorithms

From the table we can see that the algorithm that only communicated every fifth iteration of the solvers outperformed the original algorithm for every parallel iteration. The algorithm is not perfect though, very small grid sizes and large amounts of processes can still produce slowdowns rather than speedups. However the tables do show that increasing the grid size improves speedup, this can be easily attributed to the fact that increasing the grid size increases the amount of time spent in performing calculations. The amount of time for communication between the same amount of processes, across all grid sizes, has less of an impact as the time to set up a communication highly outweighs the time used to send data.

The results from the algorithm that only communicated every fifth iteration of the solver were slightly different, but visually similar. Thus we conclude that only sending information after every couple of iterations of the solver rather than after every iteration not only improves the performance, but also makes parallelisation a viable method for solving fluid animations.

UNIVERSITY OF JOHANNESBURG