

Project Presentation

Markus Palacios

markus.palacios.5733@student.uu.se

Jonathan Sharyari

jonathan.sharyari.0152@student.uu.se

Peng Sun

peng.sun.8197@student.uu.se

Department of Information Technology

Uppsala University

March 13th, 2015





Assignment 1

Assignment 2

Assignment 3

Assignment 4

```
40
41 void* tickHelp(void *arg){
42     struct interval myInterval = *((struct interval*)arg);
43     for (int i=myInterval.left; i<myInterval.right; i++){
44         (*myInterval.agents)[i]->whereToGo();
45         (*myInterval.agents)[i]->go();
46     }
47 }
48
49 void Ped::Model::tick(){
50     int numAgents = agents.size();
51     int numProc = np;
52
53     if (getPar() == PTHREAD){
54         int blocksize = agents.size()/numProc;
55         pthread_t threads[numProc];
56         struct interval * intervals[numProc];
57
58         for (int i=0; i<numProc; i++){
59             intervals[i] = new struct interval();
60             intervals[i]->left = i*blocksize;
61             intervals[i]->right = (i+1)*blocksize;
62             intervals[i]->agents = &agents;
63             pthread_create(&threads[i], NULL, &tickHelp, (void*) intervals[i]);
64         }
65         void * result;
66         for (int i=0; i< numProc; i++){
67             pthread_join(threads[i], &result);
68         } else if (getPar() == OMP) {
69             #pragma omp parallel for num_threads(np)
70             for (int i=0; i<numAgents; i++){
71                 agents[i]->whereToGo();
72                 agents[i]->go();
73             }
74         } else {
75             for (int i=0; i<numAgents; i++){
76                 agents[i]->whereToGo();
77                 agents[i]->go();
78             }
79         }
80     }
81 }
```

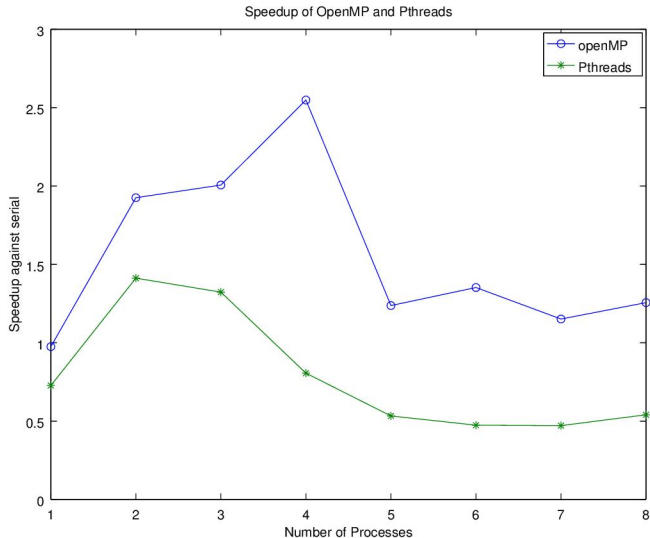


Figure : Graph showing the speedup of openMP and pthreads for 1 to 8 threads.



Assignment 2

Assignment 1

Assignment 2

Assignment 3

Assignment 4

- New data structure: e.g. `x[NumOfAgents]` instead of `agent[i].x`
- New versions of `whereToGo()` and `go()`, for vectorization and CUDA

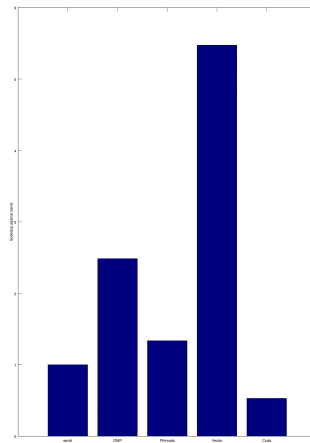


Figure : Speedup 2048 agents

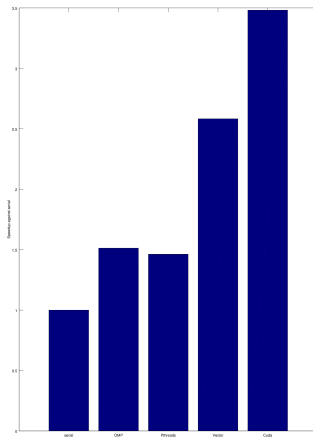


Figure : Speedup 480000 agents



Assignment 3

Assignment 1

Assignment 2

Assignment 3

Assignment 4

- Exchanged the tree structure to a simple 2d grid
- Two solutions: region-based pthreads implementation and simple omp pragma

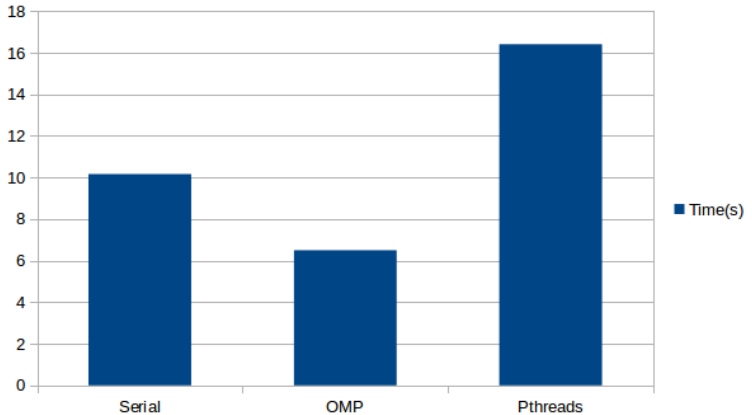


Figure : Runtime 2048 agents (before duplicates)

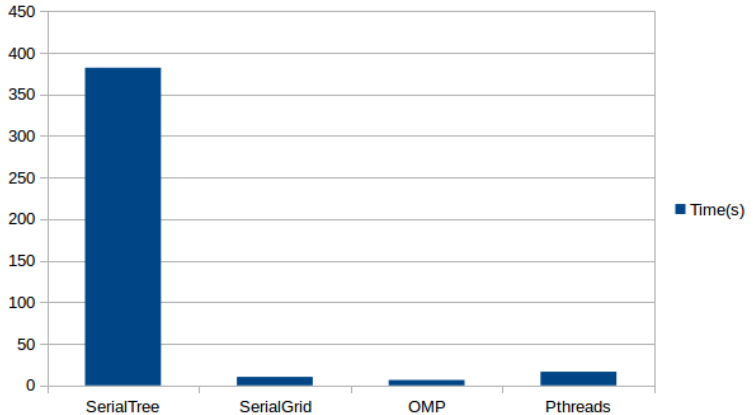


Figure : Runtime 2048 agents (before duplicates)



Assignment 4

Assignment 1

Assignment 2

Assignment 3

Assignment 4

Four CUDA kernels:

- Fading the heatmap
- Updating the heatmap with agent information
- Scaling the heatmap
- Adding a gaussian blur filter to the heatmap



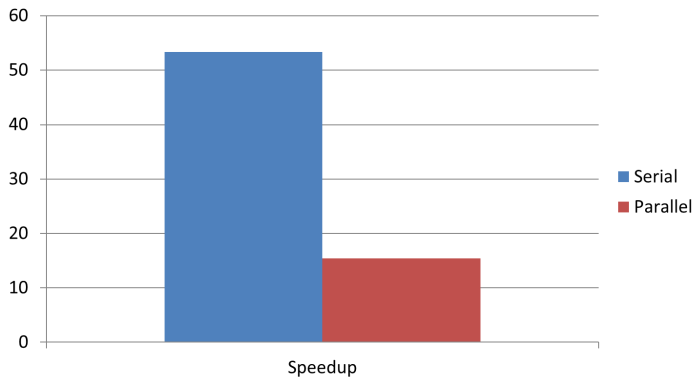
Assignment 1

Assignment 2

Assignment 3

Assignment 4

CPU	GPU
WhereToGo	Fade
Synchronize with GPU	
Collision detection	Update
	Scale
	Blur
Synchronize with GPU	





Observations

Assignment 1

Assignment 2

Assignment 3

Assignment 4

- Cuda could lead to a performance degradation when dealing with small amounts of data, memory allocation overhead, synchronization, etc.
- Easier to implement and debug with SIMD
- Parallel execution in CPU and GPU with CUDA
- Right design can reduce synchronization between CPU and GPU

