Project Arimaa – CR4 : Wednesday, October 8th

Presents : Everyone, Christian Raymond, and Nikos Parlavantzas

Order of business :

1. Questions about project (Prateek)
2. GPU VS CPU (Baptiste)
3. C++11 VS OpenMP (Benoît)
4. Parallelization and strategies (Benoit & Mikail)
5. Bibliography recap (Mikail)
6. C++ library with Boost (Gabriel)
7. Report's plan (Dan)

Information :

- Questions about Prateek : explanation of what is parallelization in this project, what is MCTS algorithm (more powerful than the first algorithm Min Max, calcul of one or more iteration and run the one with the higher probability of win. The point is using statistics to choose how to play, get the feedback of the game and conclude)
- GPU VS CPU. GPU has more power but has less cash memory, it may be too long if we need to use more memory than the capacity of the cache. In this case, CPU is better. With the MPI technology, it is possible to run it in all machines. With OpenMP, it's possible to work with GPU on Grid5000 with the Nvidia technology. It's fast, there is no cost of communication but there are again few quick memory to dispatch tasks. (Baptiste) 1 small performing GPU is around 100 better than the CPU one. Baptiste will look for figures. Maybe a good thing will be to decompose the calculus of the node, and give only a small part to the GPU so it could work correctly with a small cache.
- Multithreading C++11 VS OpenMP has not been studied by Benoit. He should have warned us that he was unable to do it. It is for now not a problem because we don't really need this study for the first report but he will have to do it just after we give our final first report. However, we should not forget that we will mention this possibility of multithreading on the first report, but we will not look for a solution, we will just present the possibilities.
  A Json structure could be promising.
- Mikail studied parallelizations. He compared the three simple one (no hybridation). He studied only the first thesis so he's going to read more with the teacher's thesis. His conclusions were that the root one is the best, on most of situations. Leaf parallelization can be efficient as well on numberous conditions. Based on that thesis, we would have chosen root parallelization.
  But Mikail is going to work on other thesis. We are know going to try hybride parallelization for now because it's less risky to begin with a very known way of parallelizing. We need to dig further to compare other thesis on the topic.
  Another problem is redondancy in the calculus
- Mikail worked as well on the bibliography. He has chosen Zotero to create the file .bib for the latex format because he worked on it before on the monography (3rd year project). Dan managed to add this file in a Latex document, and it worked perfectly. It will be important for everyone who will write the report to give the references and the places where they want to put it in the text.
- Library Boost or C++11 for parallelization and advanced options ? (Gabriel) Boost works well, it's similar to C++11 because C++11 has been based on some of Boost fonctions. So there are more possibilities using Boost. It is possible to sequence trees, it is customable with for instance list, vector or almost anything we want. There is a detailed documentation. The affectation of the memory space can be different, it seems more powerful with Boost, there are more functionalities. We won't worry about working with trees.

- Plan of the first report (Dan) : Fetch parts 1,2,3 and 4,5, remove part 7 (not what is asked), put bibliography at the end, the introduction must be about 1 page, but not to deep. The presentation of the project (part 1) will complete it. Part 6 will be superficial : telling what could be used but not answer it already. Don't forget that if you know too less on a subject you have to talk about, ask someone who knows.

Planification

Task	Responsible	Deadline
Parallelization new teacher's thesis	Mikail	10/12
Find figures on CPU/GPU (Bad GPU/CPU of you like)	Baptiste	10/12
Part presentation of our project	Prateek	10/13
End of first draft (give your work to Dan, Mail-Git)	Everyone	10PM 10/13
C++11 VS Multithreading	Benoit	10/29
Finish the game application	Gabriel	12/18

Next meeting : 10/15