ZombieSim Project

Practical Course on High-Performance Computing

Abdullah Amawi, Maaike Bierenbroodspot

September 20, 2022

Supervised by Prof. Dr. Julian Kunkel. University of Göttingen

Table of Contents

- Introduction & Recap
- Solution & Performance Analysis (HOTSPOT)
- Conclusion

Introduction & Recap

Introduction - ZombieSim

- Zombie simulator to investigate how infection spreads based on:
  - Behavior, such as humans going from their homes to work.
  - Environment, such as the population density.
  - Biological factors, such as transmission rate.
  - We intend making factors easy to add, to allow the freedom of research for additional factors.
  - Goal is to learn more about parallel computing and how to analyze parallel efficiency.

Tech Stack

- C++ as the programming language in use.
- Boost library[3]for parallel execution.
- olcPixelGameEngine[1] for the visualisation.
- HOTSPOT for performance analysis.

General Flow - ZombieSim

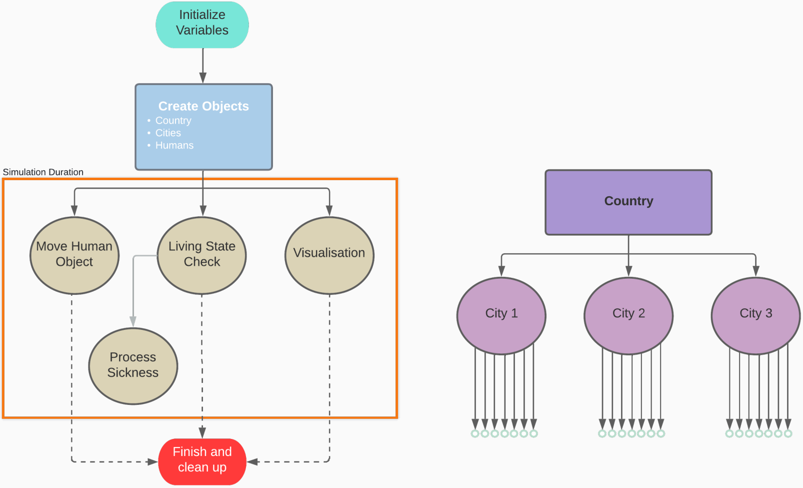


Figure 1: ZombieSim general flow & structure

Land Overview - ZombieSim

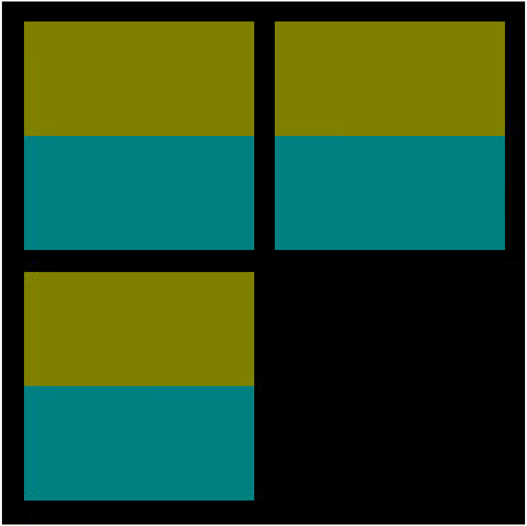


Figure 2: ZombieSim visual overview

- Residential area in green.
- Business area in blue.
- Different city sizes and densities

Challenges

- Vampir[2] & usage of Scientific Linux library.
- Compilation for Scientific Linux distribution.
- HOTSPOT[**?**] ease of use was hindered by the limited hardware support.

Solution & Performance Analysis

ZombieSim Demo

An animation showing our ZombieSim in action.

Sequential Vs Parallel

- Started the design for the parallel execution

- Multiple Threads
  - Visualization thread
  - Moving threads
  - Managing threads 10 base

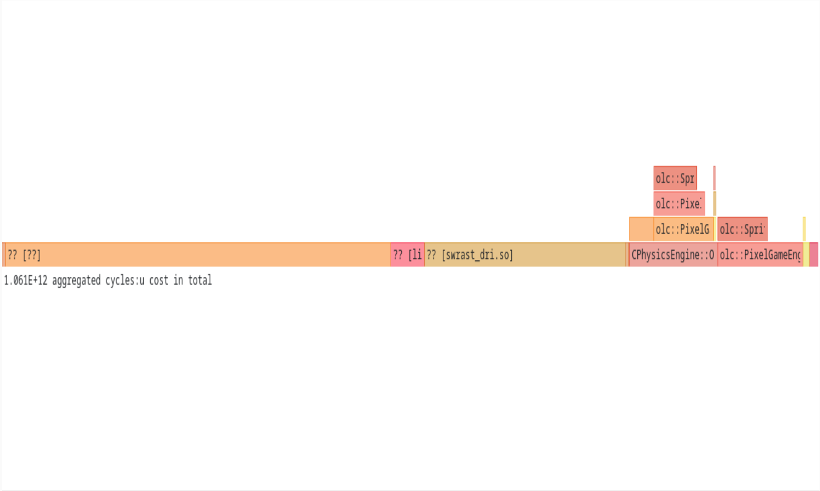
Performance Analysis

- Human object scaling, more humans, slower, or more threads..
- More threads are needed to scale.
- Infection rate, more people infected, more overlap has to be done, so its slower.
- Value overlap threads, more threads, workload is divided, so it scales better

Scaling

- Compared 1 thread performance to 10.
- Ran for 120 seconds as configured in the conf file, makes it modular.
- Hotspot shows 241 vs 1039 threads.
- Over 60% degradation in the performance.

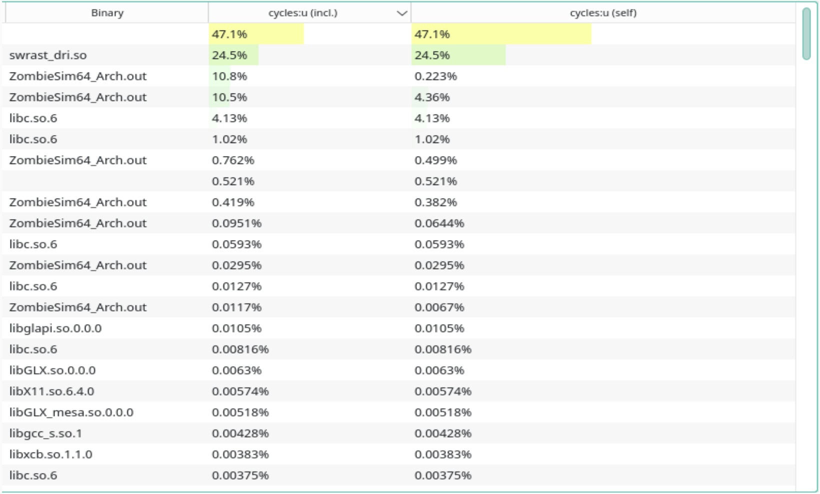
Graph-1



1.061E+12 aggregated cycles:u cost in total

Figure 3: Hotspot graph result

Graph-2



Binary	cycles:u (incl.)	cycles:u (self)
	47.1%	47.1%
swrast_dri.so	24.5%	24.5%
ZombieSim64_Arch.out	10.8%	0.223%
ZombieSim64_Arch.out	10.5%	4.36%
libc.so.6	4.13%	4.13%
libc.so.6	1.02%	1.02%
ZombieSim64_Arch.out	0.762%	0.499%
	0.521%	0.521%
ZombieSim64_Arch.out	0.419%	0.382%
ZombieSim64_Arch.out	0.0951%	0.0644%
libc.so.6	0.0593%	0.0593%
ZombieSim64_Arch.out	0.0295%	0.0295%
libc.so.6	0.0127%	0.0127%
ZombieSim64_Arch.out	0.0117%	0.0067%
libglapi.so.0.0.0	0.0105%	0.0105%
libc.so.6	0.00816%	0.00816%
libGLX.so.0.0.0	0.0063%	0.0063%
libX11.so.6.4.0	0.00574%	0.00574%
libGLX_mesa.so.0.0.0	0.00518%	0.00518%
libgcc_s.so.1	0.00428%	0.00428%
libxcb.so.1.1.0	0.00383%	0.00383%
libc.so.6	0.00375%	0.00375%

Figure 4: Hotspot results

Conclusion

Conclusion

- A lot was learned about parallel execution importance.
- Challenges were an eye opener in many areas.
- Goals were partially obtained, more to do in the work for the report.
- The practical course & the project were a great learning experience.

Thank you!

Questions?

References i

For additional information, please refer to our Technical Design document for ZombieSim, or the references for the libraries and tools used.

https://docs.google.com/document/d/1URngIbPCFHhuE6nuxOl2-06GZLJ0niEb/edit

- https://github.com/onelonecoder/olcpixelgameengine.
- https://vampir.eu/.
- https://www.boost.org/.