Wild-Fire-Simulation in C++

Presented by Claas Kochanke, Jan Lenke, David Alexandre Silva

21.06.2024

Outline

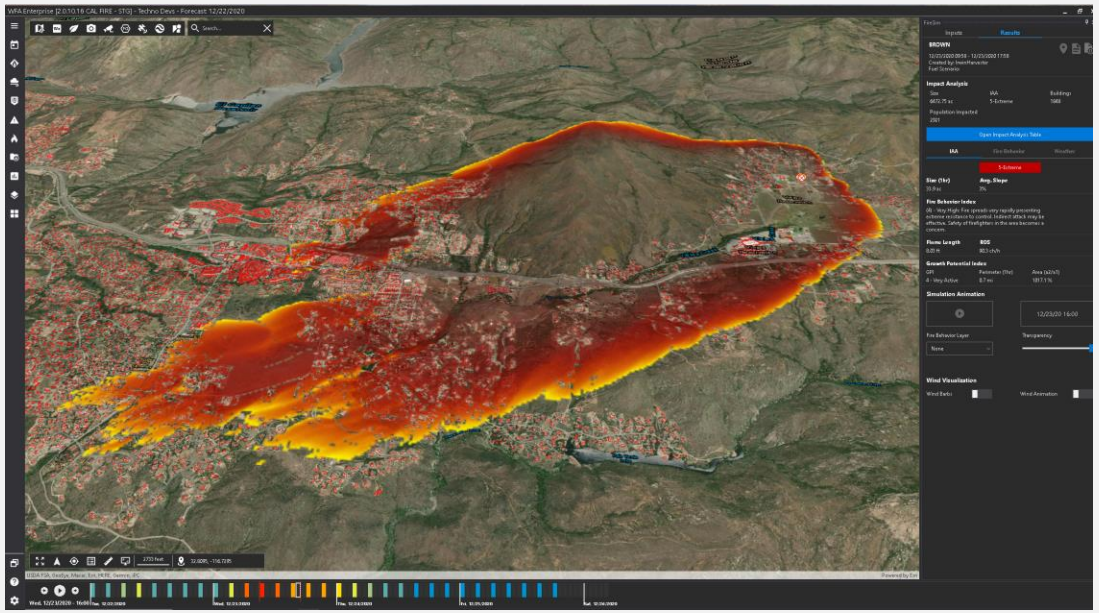
1. Introduction
2. Solution Approach
3. Sequential Solution
4. Parallelized Solution
5. Performance Analysis
6. Conclusion

Goals and Contribution

- Implement a reasonably scientifically accurate wildfire simulation
- Simulate a forest with a 30000x30000 grid in a reasonable time frame
- Show the effectiveness of the parallelization compared to the sequential solution
- Evaluate the performance in detail and show interesting correlations between parameters

Wildfire simulation

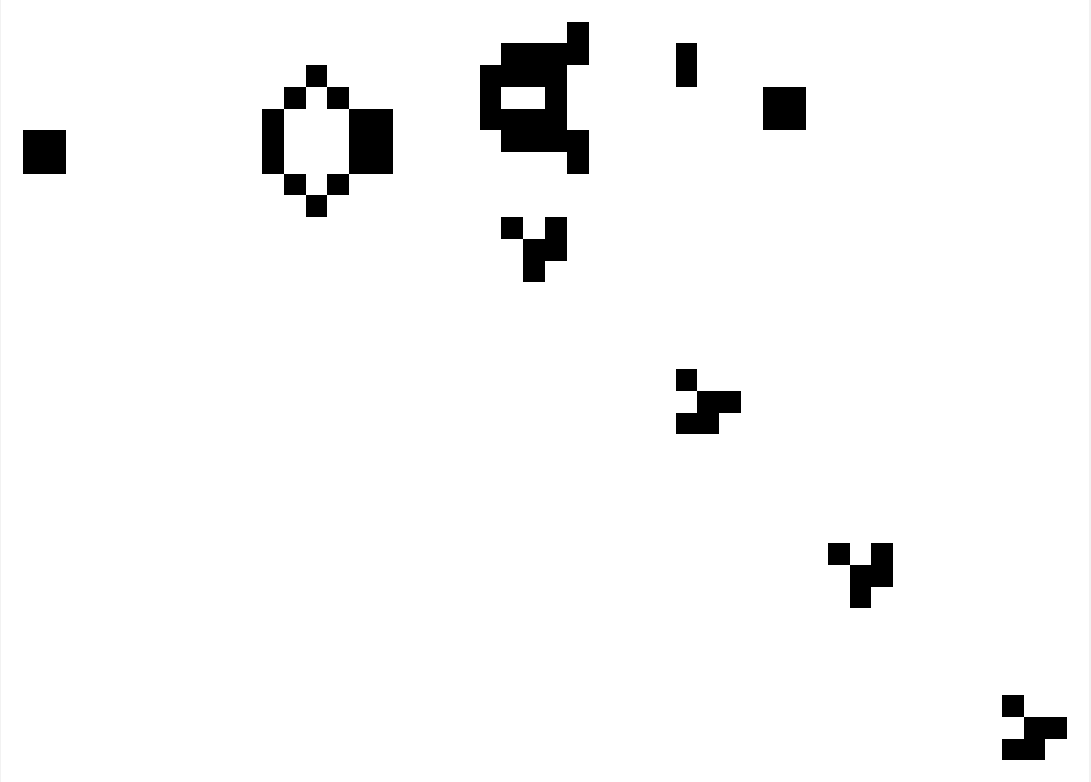
- Probabilistic model using a cellular automata
- Non-trivial parallelization due to edge dependence
- Use-cases:
  - Predict potential spread and intensity of wildfires
  - Aid in planning and designing effective control strategies (e.g. Infrastructure planning)
  - Planning evacuation routes and shelters in an emergency in a timely manner



[4]

Cellular Automata

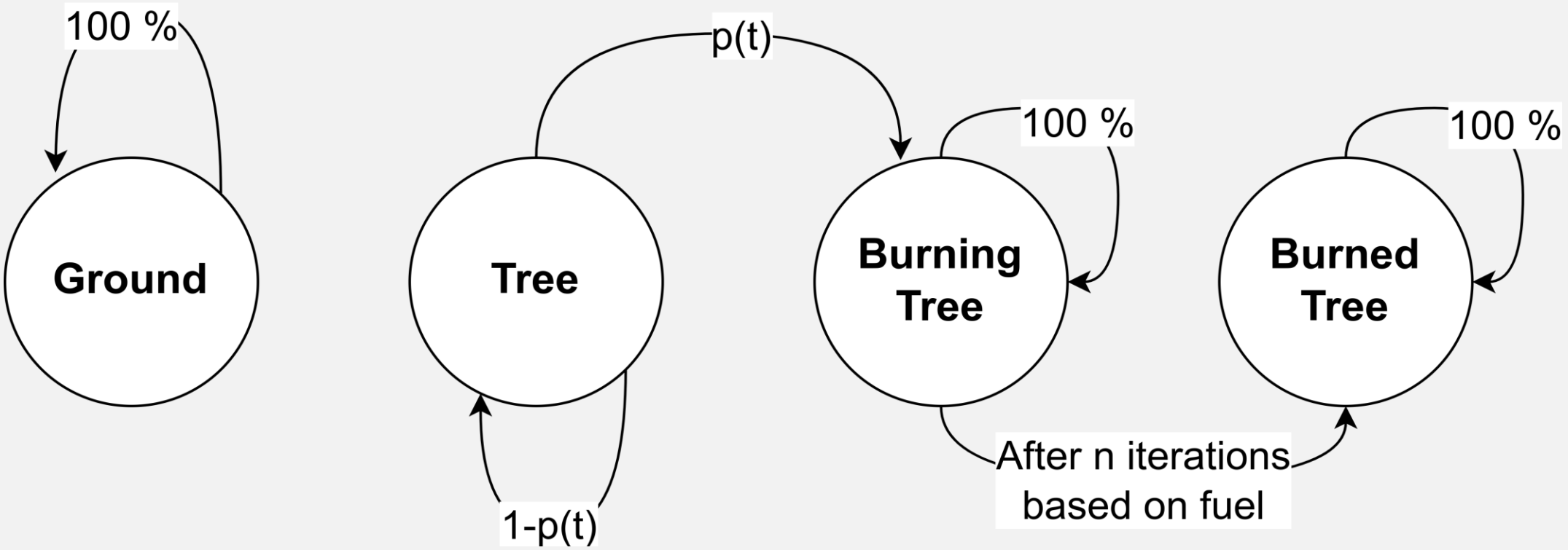
- Components:
  - Grid of cells with finite number of states
  - A set of rules to update the cells
- Used in different fields
- Useful for various simulations like a wildfire simulation



[6]

Approach

State diagram

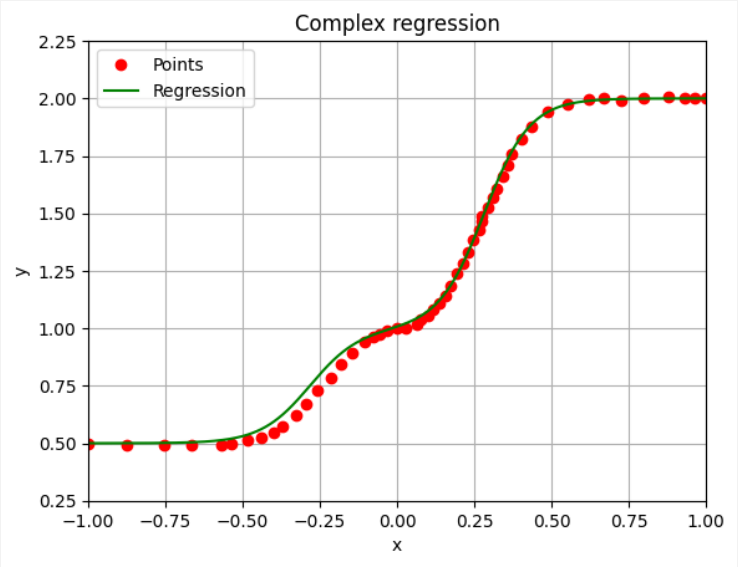


Review of scientific models

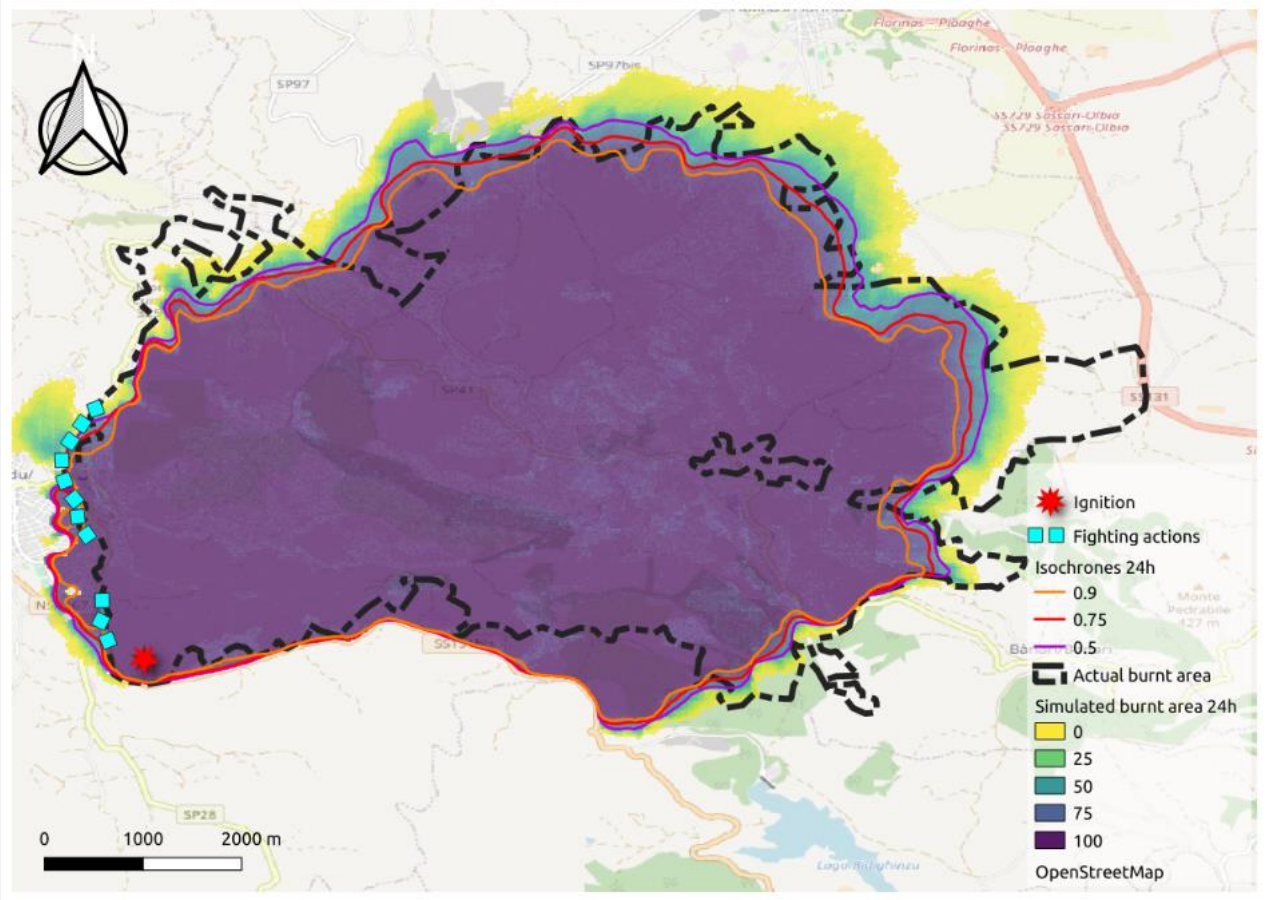
- PROPAGATOR: An Operational Cellular-Automata Based Wildfire Simulator [1]:

$$p_{ij} = (1 - (1 - p_n)^{\alpha_{wh}}) \cdot e_m$$

- Reverse Engineering through Regressions:

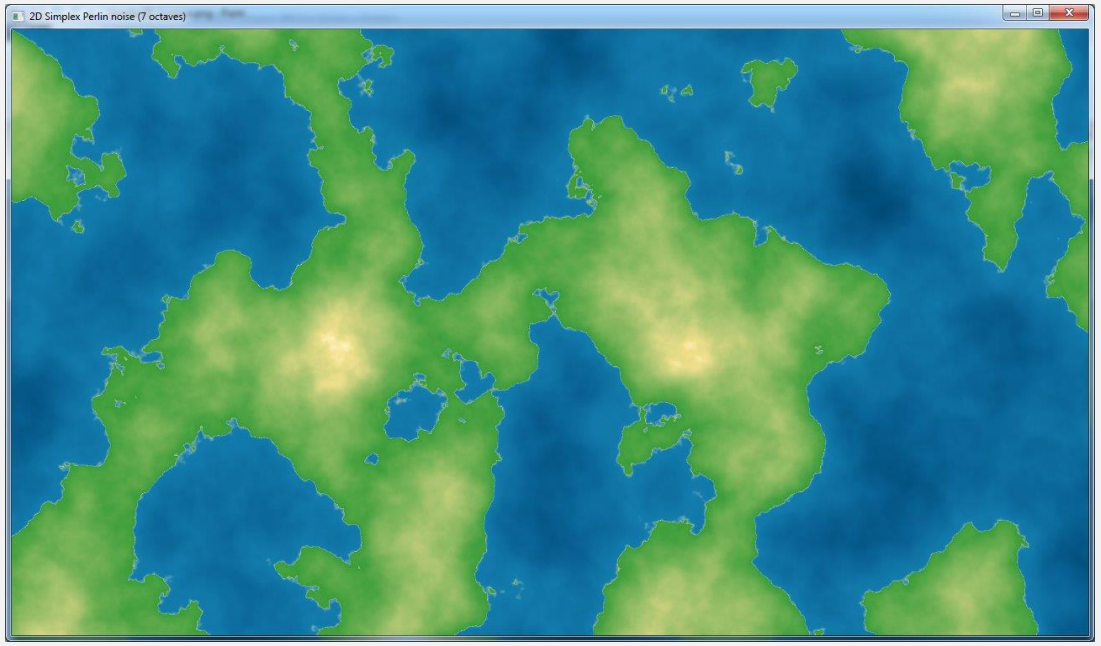


PROPAGATOR: An Operational Cellular-Automata Based Wildfire Simulator



[1]

Procedural generation



[2]

- SimplexNoise algorithm to procedually generate the forest
- Mapping functions influence the output:
  - Trees ($x >= 0$ with $p \propto x$) and Ground ($x < 0$)
  - Terrain height: Normalize values to [0; 100]
  - Tree types:
    - Generate log(k)+1 masks using SimplexNoise
    - Add them together using bit operations
- Each random decision is dependent on a seed and the coordinates (x,y)

Sequential Solution

Wind speed and direction

- Wind speed and direction is a global value and affects the fire spreading probability

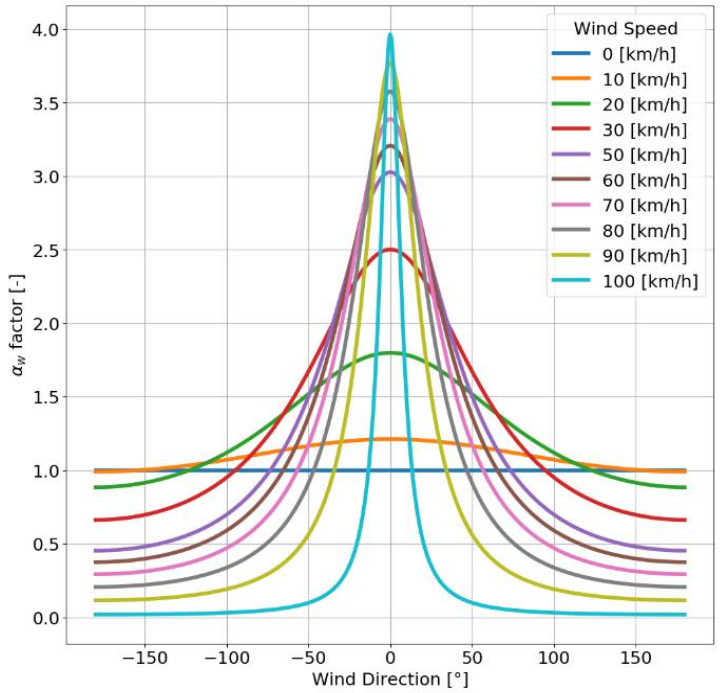
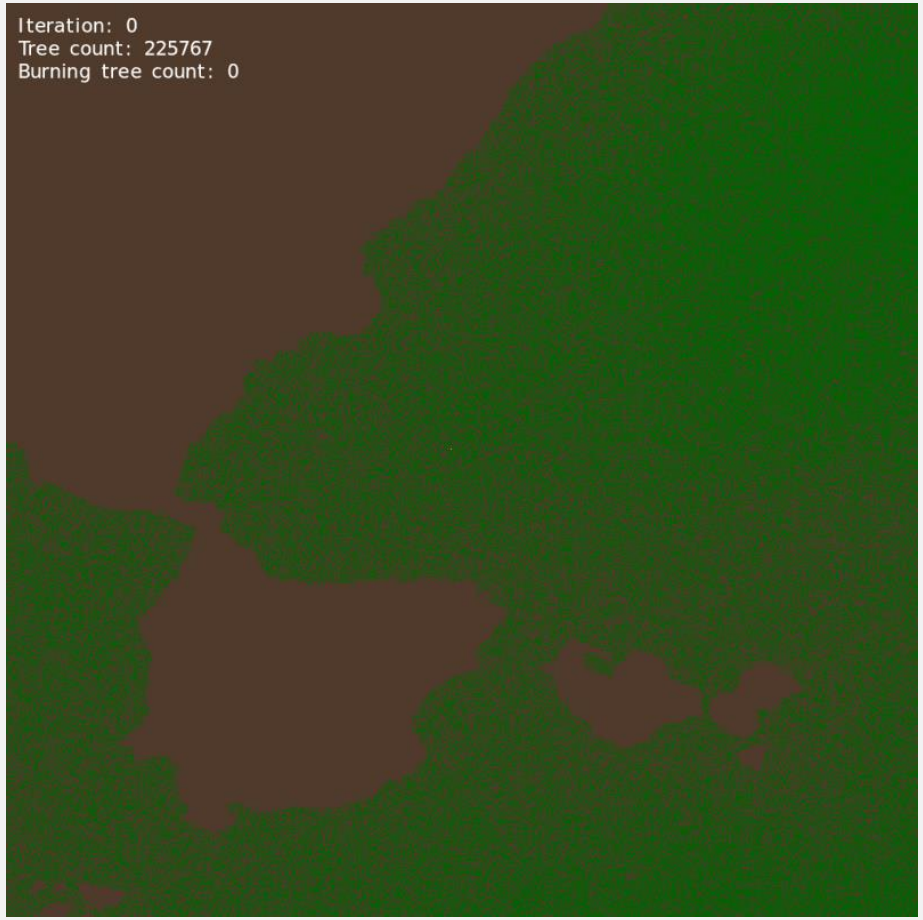


Figure 7. Wind influencing factor α_w.

[1]

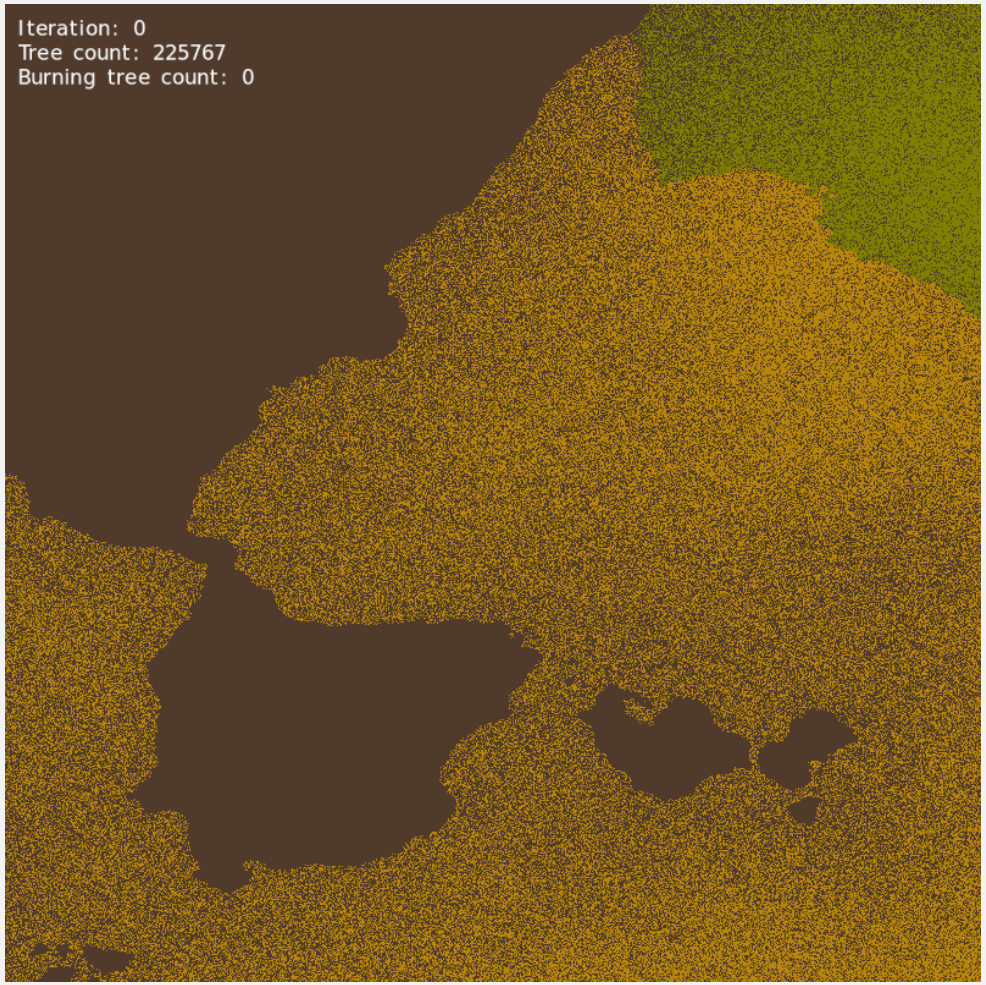


Tree types

- Tree types effect base fire spreading probability
- e.g. Not fire-prone forest doesn't spread fire easily (7.5 %)

		Burning Cell					
		Broadleaves	Shrubs	Grassland	Fire-Prone Conifers	Agro-Forestry Areas	Not Fire-Prone Forest
neighbor cells	Broadleaves	0.3	0.375	0.25	0.275	0.25	0.25
	Shrubs	0.375	0.375	0.35	0.4	0.3	0.375
	Grassland	0.45	0.475	0.475	0.475	0.375	0.475
	Fire-prone conifers	0.225	0.325	0.25	0.35	0.2	0.35
	Agro-forestry areas	0.25	0.25	0.3	0.475	0.35	0.25
	Not fire-prone forest	0.075	0.1	0.075	0.275	0.075	0.075

[1]



Terrain height

- The fire spreads faster uphill and slower downhill

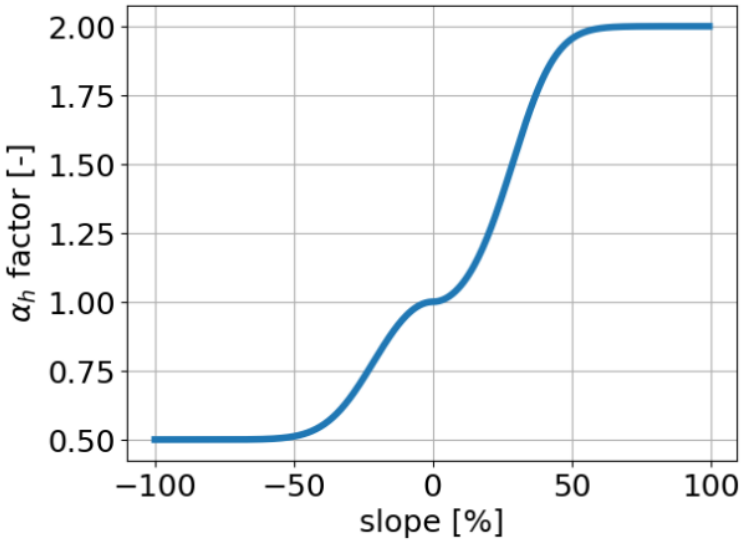
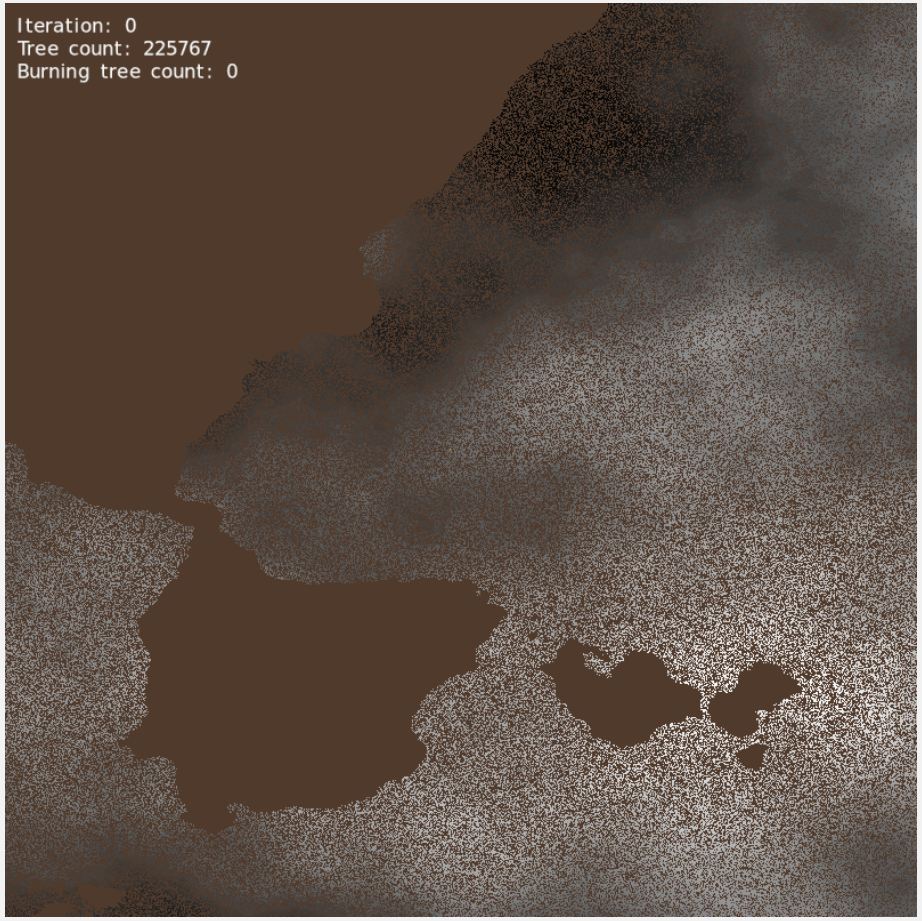


Figure 6. Topographic influencing factor α_h.

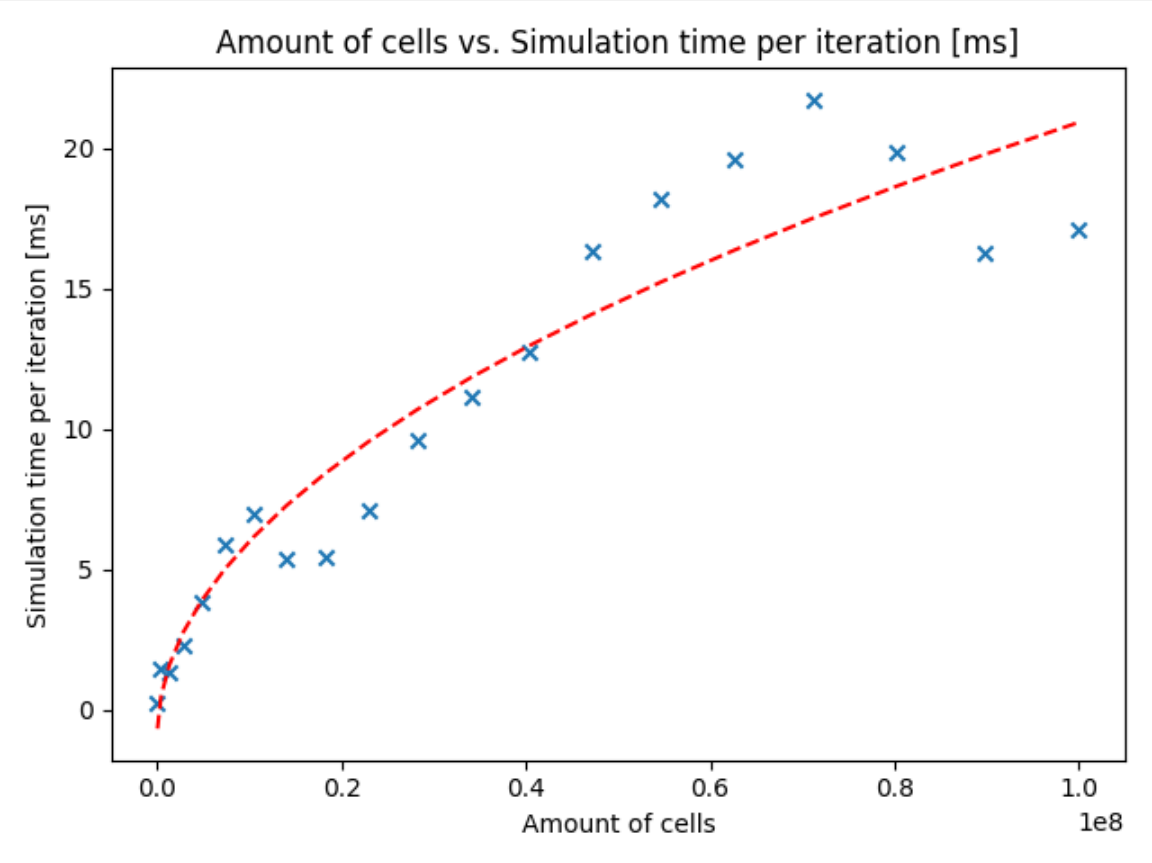
[1]

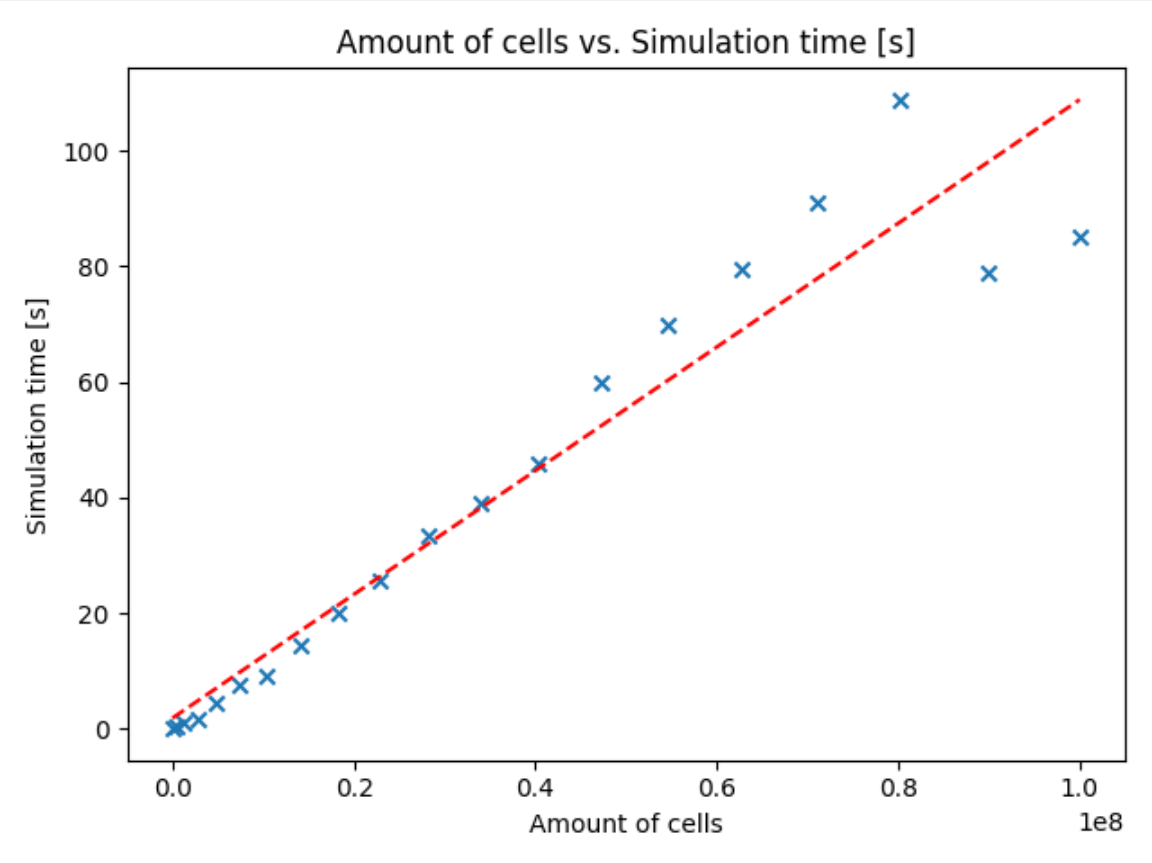


Optimization of the sequential solution and its limitations

- 100 % CPU utilization on a single-core
- Time per iteration grows based on problem size significantly
  - Time per iteration depends on amount of burning trees in each iteration
  - Number of iterations depends on size of forest
- Probabilistic simulations are run multiple times to average the results (e.g. N = 100)
- RAM can also a limitation (7 Byte per cell)
  - 100.000x100.000 = 65,19 GB
  - 200.000x200.000 = 260,77 GB

Sequential solution: Performance analysis





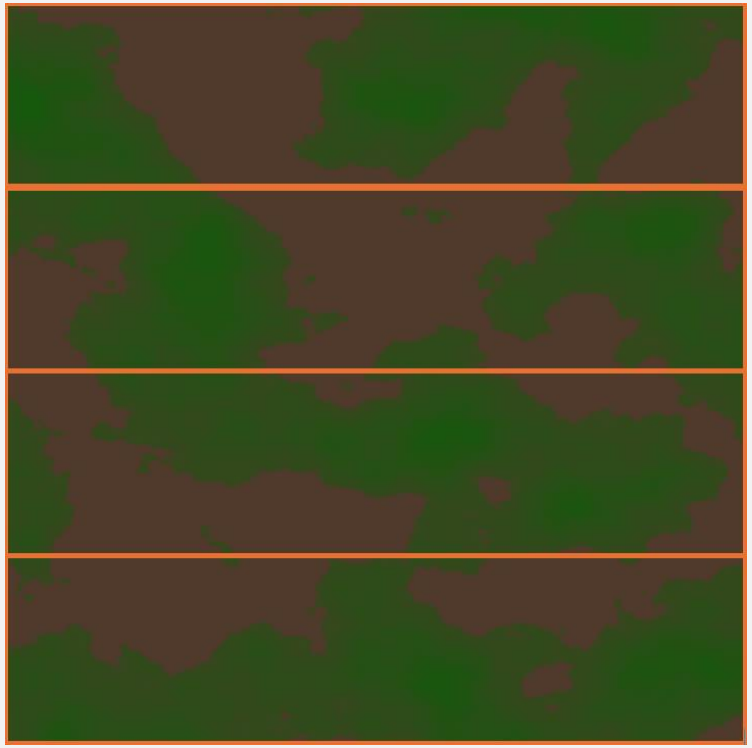
Parallelized Solution

What elements can be parallelized?

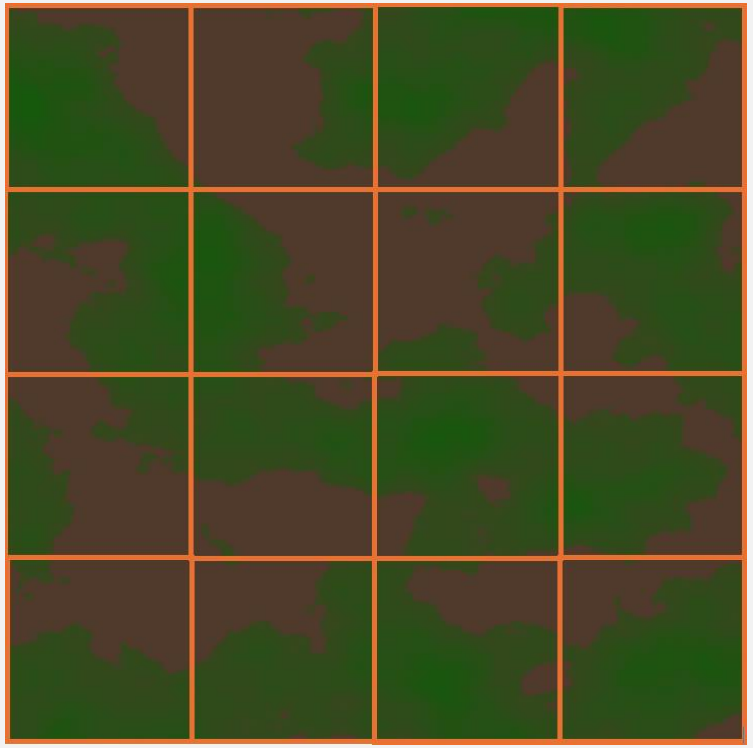
- The procedural generation of the forest
  - Each worker generates its own chunk using the SimplexNoise algorithm
  - Minimum & maximum are communicated between workers (using MPI_Allreduce)
  - No further communication needed
- Each iteration in the wildfire simulation
  - No temporal parallelization
    - Each iteration depends on the previous iteration
  - Spatial parallelization

Parallelized solution

Line parallelization



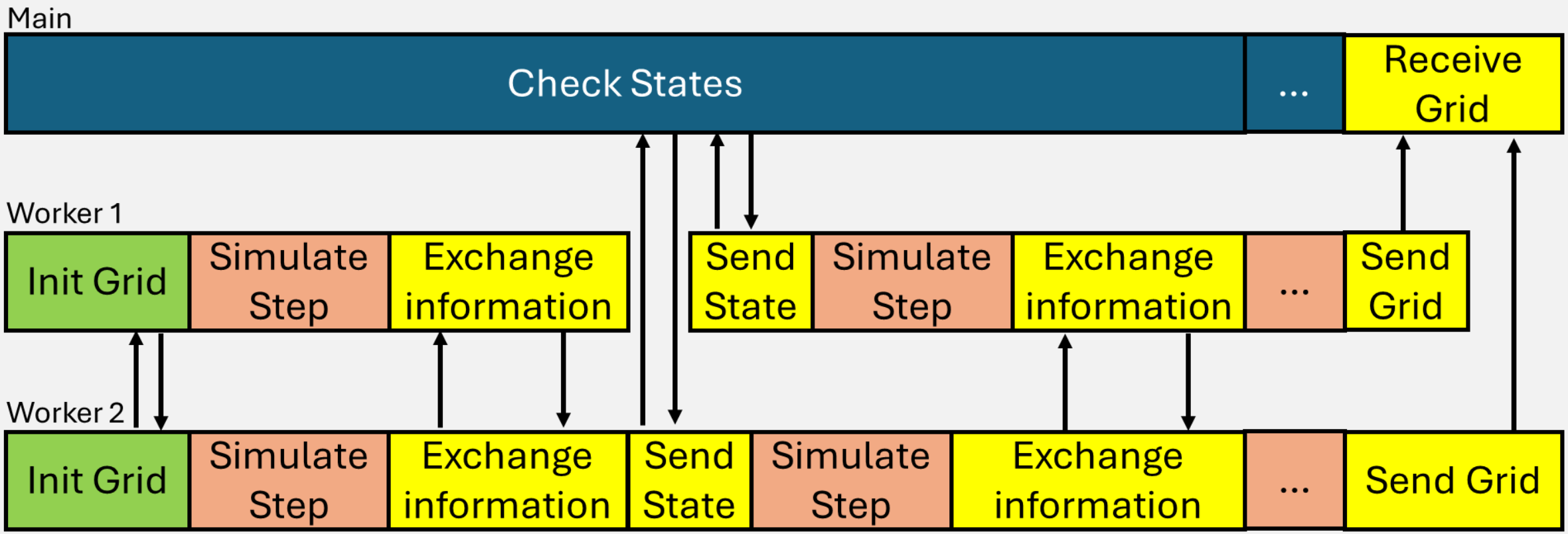
Square parallelization



Parallelized solution

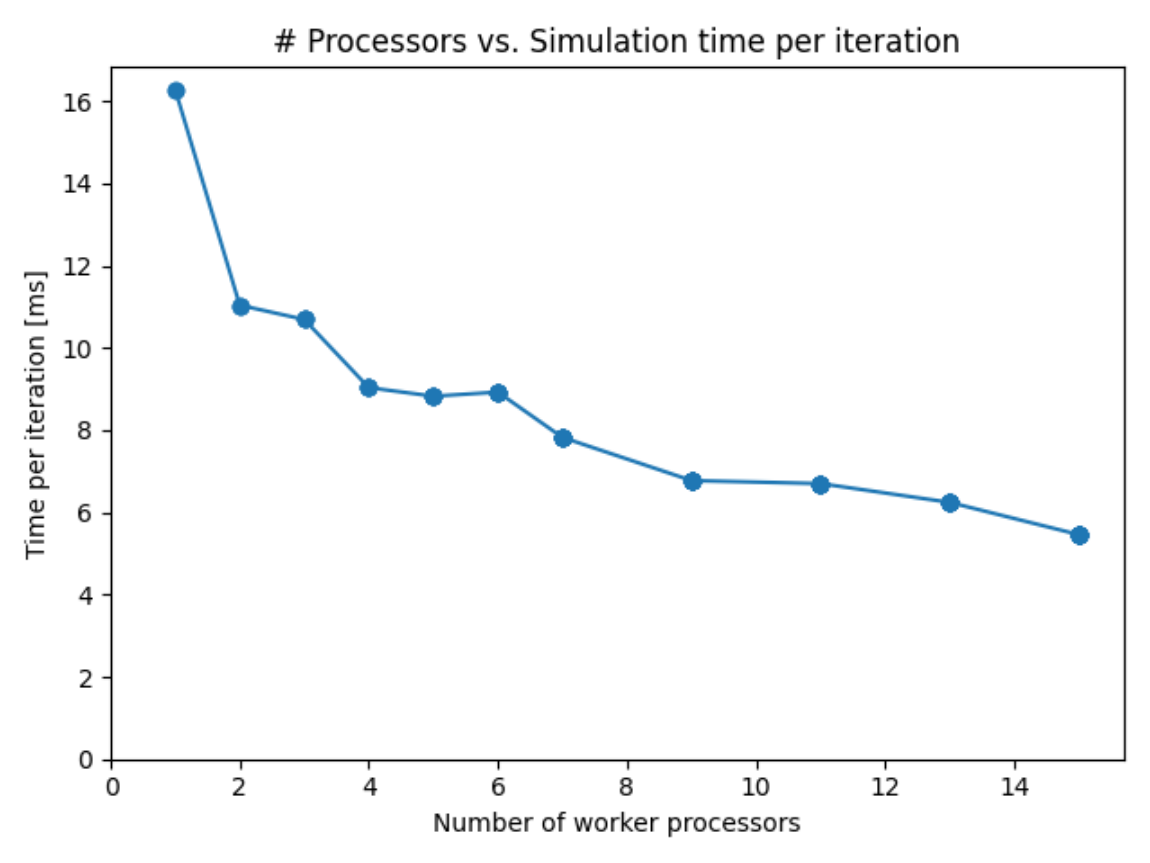
- Split the grid into smaller stripes with buffers
- Use MPI to send buffer information to the upper and lower neighbors
  - Only send edges to neighbors if needed: send a “ping” first with data amount (no update or update followed by the new data)
  - Only communicate coordinates of newly burned trees, not the entire grid
- Main process coordinates N worker processes
  - Stop when all workers stopped the simulation
  - Workers send their chunks to main process for visualization

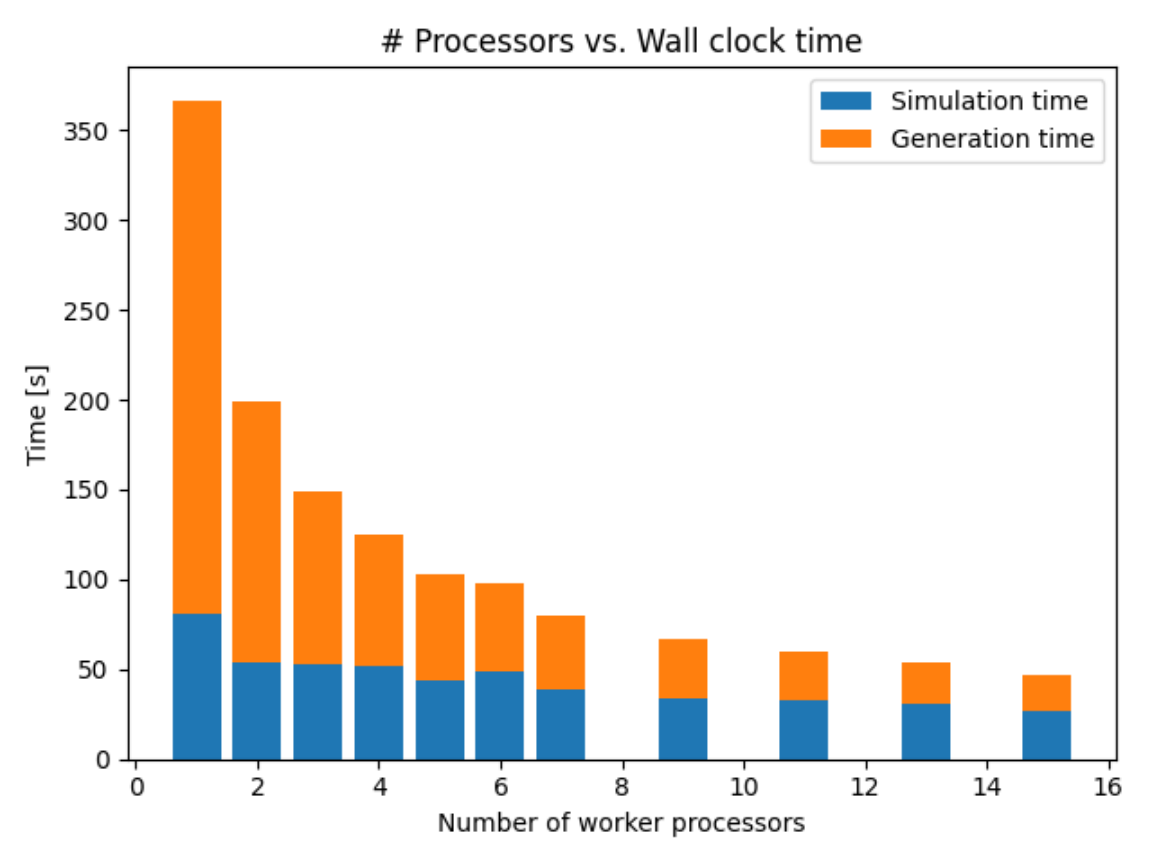
Parallelized solution



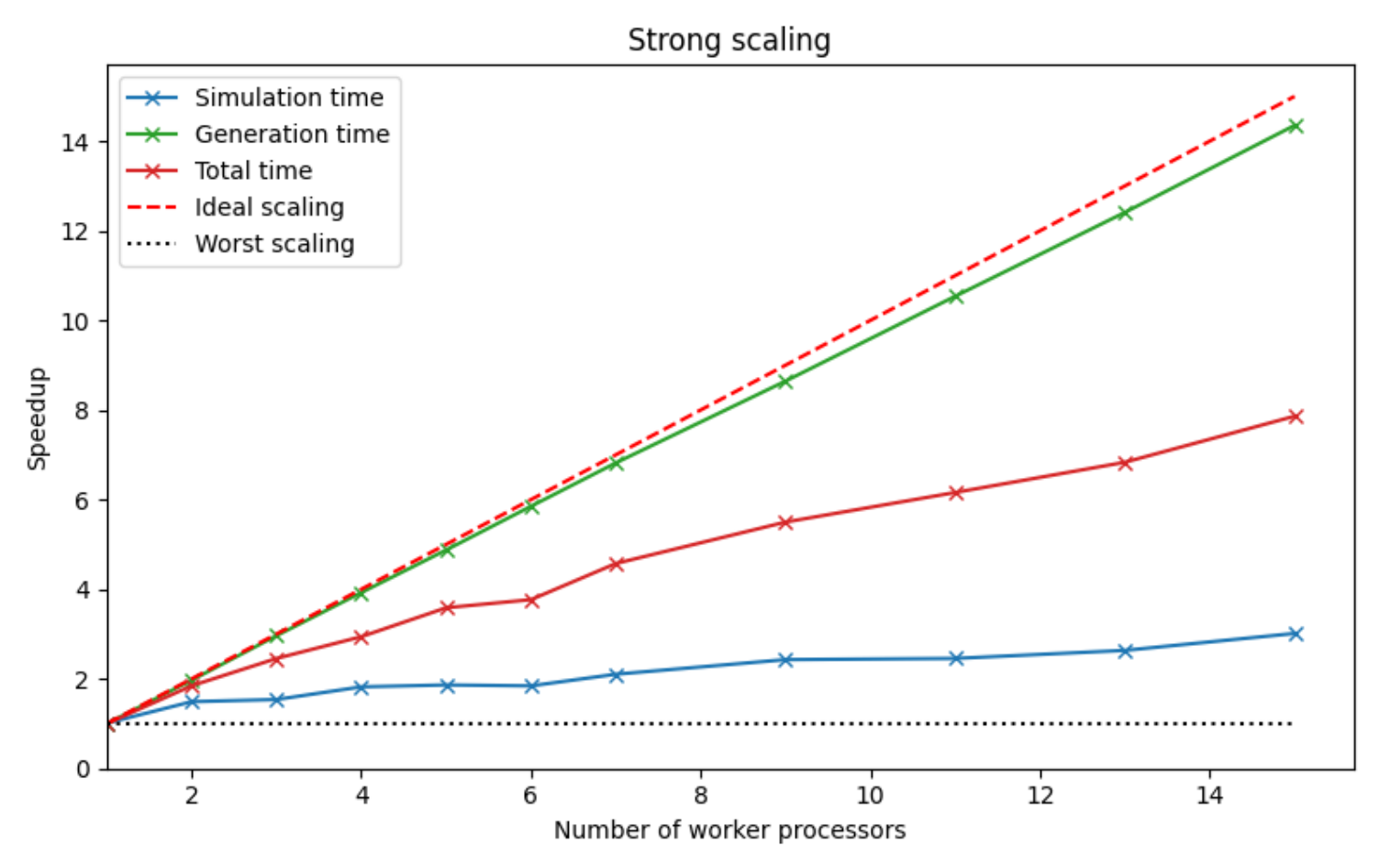
Performance Analysis

Performance Analysis



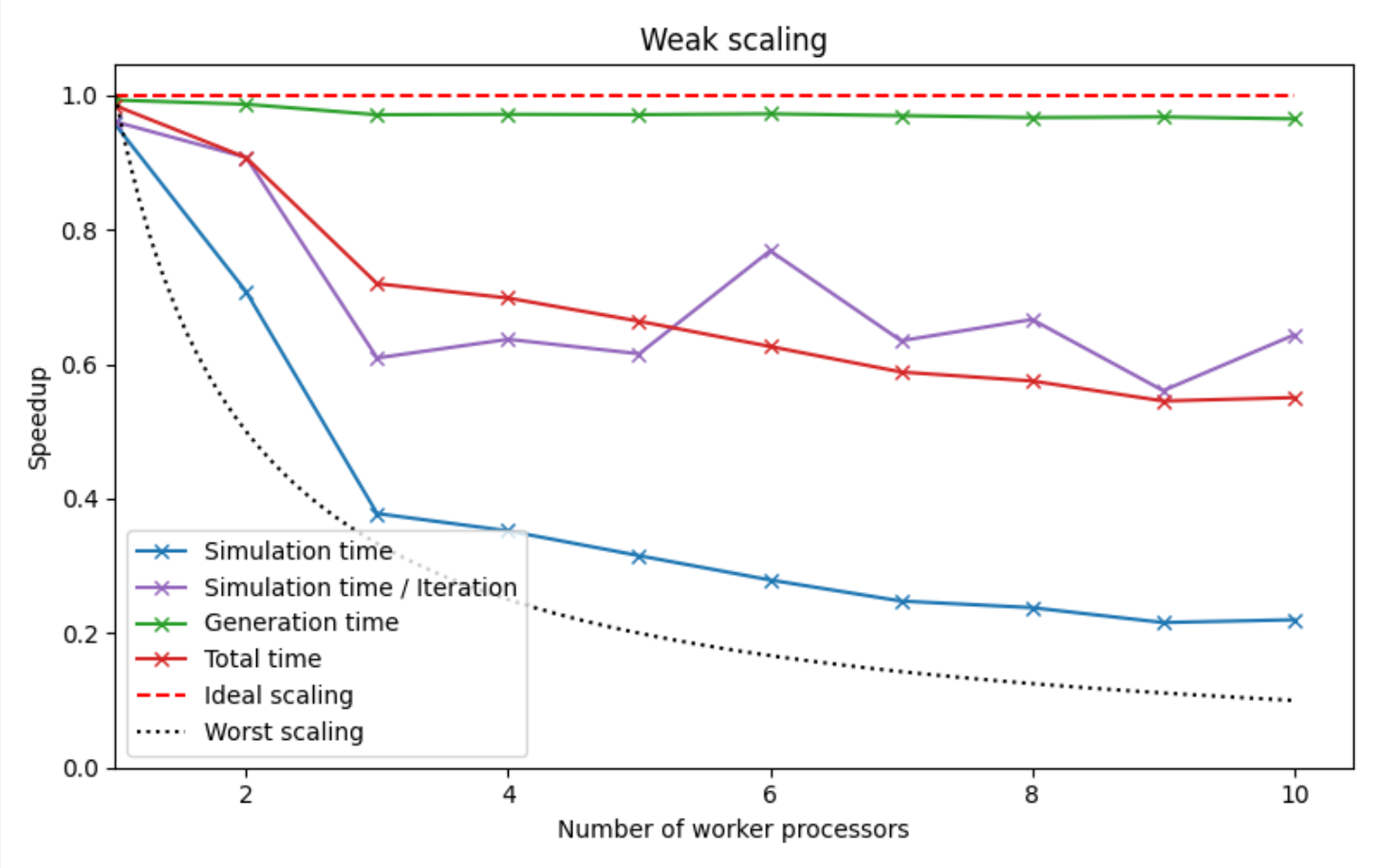


Strong scaling



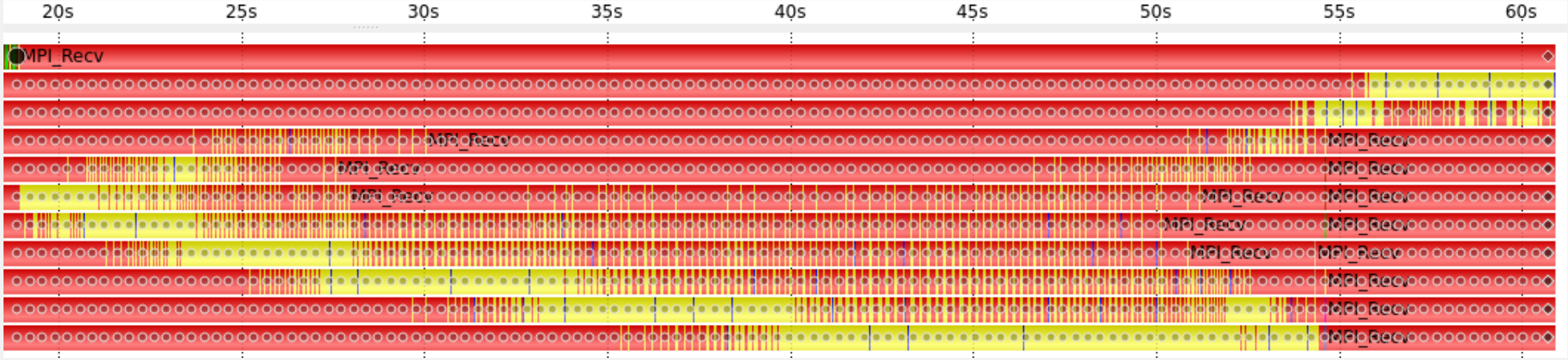
- 10 000 x 10 000 grid
- N tasks on 1 node

Weak scaling



- Number of cells per worker stays constant
- N tasks on 1 node

Vampir



Conclusion

Future work and further optimizations

- Distribute the workload better through slicing
  - (1) … (N) (1) … (N) (1) … (N)
- Dynamic load balancing
  - Shared memory is beneficial here
- More detailed performance analysis
- Show correlations between parameters using phase diagrams
  - e.g. how fast does the forest burn down
  - e.g. how wind, terrain height, tree types, … influence fire spreading

Conclusion

- A sequential solution can be optimized effectively
  - especially in lower-level languages like C++
- MPI improved the simulation, especially the generation
- Dynamic load balancing is needed for more efficiency
- SimplexNoise is a very adaptible algorithm
  - Can be used for various types of procedural generation

- Our goals:
  - [✓] Implement a reasonably scientifically accurate wildfire simulation
  - [✓] Simulate a forest with a 30000x30000 grid in a reasonable time frame
  - [✓] Show the effectiveness of the parallelization compared to the sequential solution
  - [...] Evaluate the performance in detail and show interesting correlations between parameters

References

- [1] Trucchia, A., D'Andrea, M., Baghino, F., Fiorucci, P., Ferraris, L., Negro, D., … Severino, M. (2020). PROPAGATOR: An Operational Cellular-Automata Based Wildfire Simulator. Fire, 3(3). doi:10.3390/fire3030026
- [2] Perlin, K. (2002). Improving noise. ACM Trans. Graph., 21(3), 681–682. doi:10.1145/566654.566636
- [3] Papadopoulos, G. D., & Pavlidou, F. N. (2011). A comparative review on wildfire simulators. IEEE systems Journal, 5(2), 233-243.
- [4] technologyreview.com. (n.d.). What the complex math of fire modeling tells us about the future of California's forests. Retrieved from https://www.technologyreview.com/2021/01/18/1016215/complex-math-fire-modeling-future-california-forests/
- [5] ETH Zürich - Laboratory of Hydraulics, H., & Glaciology. (n.d.). Solving partial differential equations in parallel on GPUs - Lecture 8. Retrieved from https://pde-on-gpu.vaw.ethz.ch/lecture8/
- [6] commons.wikimedia.org (2005). https://commons.wikimedia.org/wiki/File:Gospers_glider_gun.gif