VPS 5	Parallel and Distributed Software Systems	WT 21/22, Exercise 1

Deadline: 15.11.2021 15:00

Name ______________________________

Points ________ **Effort in hours** ________

1. Theory – Amuse-Gueule … (2 + 2 + 4 Points)

Assume a given algorithm, which solves a problem of size p in parallel. For a problem size of $p = 10$, the algorithm has a relative sequential part $\sigma = 0.2$ (i.e., 20% of the algorithm cannot be parallelized).

a) Calculate and plot speedup and efficiency, which can be achieved for this algorithm with increasing numbers of processors n (i.e., cores). What is the upper limit of the speedup?

Further assume, that the sequential part of the algorithm has an asymptotic runtime complexity of $O(p)$ and the parallel part of the algorithm as an asymptotic runtime complexity of $O(p^2)$.

b) Calculate and plot the relative sequential part σ with increasing problem sizes.

c) For a problem size of $p = 100$, 1.000 and 10.000, how many processors can be utilized, if the efficiency has to be above 80%?

2. Wator – Eat or be eaten … (4 + 12 Points)

Wator is the name of a small circular planet, far far away from our galaxy, were no one has ever gone before. On Wator there live two different kinds of species: *sharks* and *fish*. Both species live according to a very old set of rules, which has not been changed for the last thousands of years.

For **fish** the rules are:

- at the beginning of all time there were f fish
- each fish has a constant energy E_f
- in each time step a fish moves randomly to one of its four adjacent cells (up, down, left or right), if and only if there is a free cell available
- if all adjacent cells are occupied, the fish doesn't move
- in each time step fish age by one time unit
- if a fish gets older than a specified limit B_f, the fish breeds (i.e., a new fish is born on a free adjacent cell, if such a cell is available)
- after the birth of a new fish the age of the parent fish is reduced by B_f

For **sharks** the rules are:

- at the beginning of all time there were s sharks, each with an initial energy of E_s
- in each time step a sharks consumes one energy unit
- in each time step a shark eats a fish, if a fish is on one of its adjacent cells
- if a shark eats a fish, the energy of the shark increases by the energy value of the eaten fish
- if there is no fish adjacent to the shark, the shark moves like a fish to one of its neighbor cells
- if the energy of a shark gets 0, the shark dies
- if the energy of a shark gets larger than a specified limit B_s, the shark breeds and the energy of the parent shark is equally distributed among the parent and the child shark (i.e., a new shark is born on a free adjacent cell, if such a cell is available)

a) On Moodle, you find a ready to use implementation of Wator. Make a critical review of the application and analyze its design, efficiency, clarity, readability, etc. **Document your review results properly.**

b) Change the application gradually to improve its performance. Think of **three concrete improvements** and implement them. For each improvement, document how the runtime changes (in comparison to the prior and to the initial version) and calculate the speedup. Each single optimization should yield a speedup of at least 1.05 compared to the prior version.

For the experiments in Task b) use the following settings:

Fish Settings:	
FishBreedTime	10
InitialFishEnergy	10
InitialFishPopulation	20.000

General Settings:	
DisplayWorld	**False**
Height	500
Iterations	100
Runs	5
Width	500
Workers	1

Shark Settings:	
InitialSharkEnergy	50
InitialSharkPopulation	5.000
SharkBreedEnergy	100

Notes: Improvements must not alter the simulation's inherent logic (i.e. stick to the listed rules and do not remove simulation logic, such as iteration-wise random execution order).

In this and all upcoming exercises always document your system configuration (i.e., number of cores, memory size, CPU type, etc.) when performing runtime measurements.