Problems for "High Performance Computing with R"

Florian Schwendinger, Gregor Kastner, Stefan Theußl

Due: 20.November 2021 (via email to `fschwend@wu.ac.at`)

1. Read the vignette of the parallel package.

2. (a) Implement a sequential (`sum_sequential`) and a parallel (`sum_parallel`) version of the function `sum`. Test your implementation on the following vector `vec <- as.numeric(1:1e7) / 1e7`. Is the parallel version faster than the sequential version? If not, why not?

   (b) A common problem in text mining is to calculate the inverted index of a vector. The file `inverted_index.R` (from `exercises.zip`) contains three implementations of an inverted index function.
   Implement a parallel version and benchmark the execution time for all four versions of the function (three sequential versions given in `inverted_index.R` one parallel version implemented by you) with the following parameters
   `number_of_words <- 1e5L; number_of_unique_words <- 1e4L`,
   `number_of_words <- 1e4L; number_of_unique_words <- 1e3L`,
   `number_of_words <- 1e3L; number_of_unique_words <- 100L` and
   `number_of_words <- 100L; number_of_unique_words <- 10L`.
   How does the performence of the functions change?

3. (a) The file `rng_parallel_faulty.R` (from `exercises.zip`) contains a parallel implementaion of random number generation. As the file name indicates there is something wrong with the implementaion. Find the mistake and fix it. Compare (graphically) the random numbers generated by the faulty script with the random numbers generated by the fixed script.

4. (a) Find the global minimum of the function $f : \mathbb{R} \to \mathbb{R}$ using the `R` function `optim()`. Try several different starting points. What is the behavior of the `"Nelder-Mead"` algorithm (the default) in terms of optima found?

$$f(x,y) = 3(1-x)^2 e^{-x^2-(y+1)^2} - 10(\frac{x}{5} - x^3 - y^5)e^{-x^2-y^2} - \frac{1}{3}e^{-(x+1)^2-y^2}$$

Use **parallel** and **foreach** to solve the above problem.

5. (a) Use Monte Carlo simulation to find the fair price for a lookback call option with payoff

$$\max_{0 \le t \le T} S(t) - S(T),$$

by approximating this maximum for suitable, equally spaced values of $t_j \in [0,T]$. The current price of the underlying $S_0 = 100$, volatility $\sigma = 0.3$, time to maturity $T = 3$ (in years), and the risk-free interest rate $r = 1\%$. Assume that the underlying follows the usual Black-Scholes SDE

$$\frac{dS(t)}{S(t)} = rdt + \sigma dW(t),$$

where W is a standard Brownian motion. Show graphically how estimation accuracy depends on the spacing of the discrete grid (i.e. $t_i - t_{i-1}$) and give (approximate) confidence bounds for the accuracy of your result for different values of simulated underlyings.

(b) Use parallel Monte Carlo simulation to price the option presented above. How does this affect the run time? Provide a table showing the scaling efficiency for several numbers of cores employed. Use both shared memory (e.g. `mclapply`) as well as distributed memory (e.g. `parLapply`) implementations.