**Blue Waters Petascale Semester Curriculum v1.0**

**Unit 2: Parallel Computing Concepts**

**Lesson 4: Parallel Algorithms 1**

**Instructor Guide**

*Developed by Beau Christ for the Shodor Education Foundation, Inc.*



*Except where otherwise noted, this work by The Shodor Education Foundation, Inc. is licensed under CC BY-SA 4.0. To view a copy of this license, visit*[*https://creativecommons.org/licenses/by-sa/4.0*](https://creativecommons.org/licenses/by-sa/4.0)

*Browse and search the full curriculum at*[*http://shodor.org/petascale/materials/semester-curriculum*](http://shodor.org/petascale/materials/semester-curriculum)

*We welcome your improvements! You can submit your proposed changes to this material and the rest of the curriculum in our GitHub repository at*[*https://github.com/shodor-education/petascale-semester-curriculum*](https://github.com/shodor-education/petascale-semester-curriculum)

*We want to hear from you! Please let us know your experiences using this material by sending email to* [*petascale@shodor.org*](mailto:petascale@shodor.org)

This lesson introduces the idea of an **embarrassingly parallel** **algorithm** toward the purpose of getting students to start thinking about parallel design.

There is no formal code in this lesson, and it is not recommended to dive into a programming language at this stage. The goal is to introduce simple problems and their sequential solutions, and then give students time to think about a parallel solution before revealing the answer.

If your students are familiar with pseudocode and how to write it, you could introduce the three problems themselves, then have them first come up with the sequential pseudocode before approaching the parallel versions.

**Common Pitfalls for Students and Instructors**

A common pitfall is moving too quickly through this lesson. After presenting a sequential solution to a problem, it is recommended to pause to give students time to really think about how to parallelize before simply revealing the solution.

Otherwise, no other pitfalls should be expected.