Lesson 5: Parallel and Distributed Algorithms

Overview

In this lesson students explore the benefits and limitations of parallel and distributed computing. First they discuss the way human problem solving changes when additional people lend a hand. Then they run a series of demonstrations that show how simple tasks like sorting cards get faster when more people help, but there is a limitation to the efficiency gains. Later in the lesson students watch a video about distributed computing that highlights the ways distributed computing can help tackle new kinds of problems. To conclude the lesson students record new vocabulary in their journals and discuss any open questions.

Purpose

This lesson is a quick tour of the challenges and benefits of parallel and distributed computing. It caps off the unit to showcase ways these techniques are being used to push the boundaries of how efficiently computer can solve problems.

Standards

Full Course Alignment

CSP Conceptual Framework

▶ **CSN-2** - Parallel and distributed computing leverages multiple computers to more quickly solve complex problems or process large data sets.

CSTA K-12 Computer Science Standards (2017)

▶ **AP** - Algorithms & Programming

Agenda

Warm Up (5 minutes)

Activity (30 minutes)

- **Card Sorting Challenge**
- **Debriefing the Challenge**
- **Distributed Computing in Real World Settings**

Wrap Up (10 minutes)

- **Assessment: Check For Understanding**

Objectives

Students will be able to:

- Calculate the speedup of a parallel solution to a problem
- Describe the benefits and challenges of parallel and distributed computing.
- Explain the difference between sequential, parallel, and distributed computing.

Preparation

- Collect the manipulatives you will use for the main activity. While decks of cards are suggested, other manipulatives are possible. See the teaching tip in the main activity for suggestions.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the teachers

- **CSP Unit 6 - Algorithms** - Slides

Teaching Guide

Warm Up (5 minutes)

Discuss: *Brainstorm a task that you can complete faster if you get other people to help. What's the most number of people you'd want to help you and why?*

Discussion Goal: This should be a quick discussion to foreshadow the main ideas of the lesson. Students should brainstorm many potential tasks. When they start mentioning the maximum number of people they'd want to help them direct attention towards why that's the case. You might hear that:

- Adding extra people makes it more complicated
- Sometimes extra people doesn't really speed things up
- If you're working with lots of people then if one person is slower the whole group is slowed down

Have students think about their answers on their own, then share with a partner, and then finally discuss responses with the entire room.

Remarks

> As we've explored in this unit, computer scientists are always looking for more efficient ways to run programs. One way to do this is to develop faster algorithms that run on a single computer. Another idea we'll explore today, is figuring out ways to run programs on many computers at the same time. We just talked about some benefits and challenges when many people help with a task. As we'll see, the same is true with running programs on multiple computers. It can lead to some improvements, but also some new challenges.

Activity (30 minutes)

Card Sorting Challenge

Teaching Tip

Choosing Manipulatives: This activity can easily be done with many different types of manipulatives, not just cards. For example, students could sort pennies by even / odd year, sort coins into piles of different denominations, sort blocks by color / size, or sort any other readily available item. Pick whatever makes the most sense for your context.

Group: Place students in groups of three or four.

Distribute: Give one member of each group a deck of cards.

Challenge 1 - One Person Sort: At the front of the room display the directions for the first sort as well as the clock. Run the clock, and have students put the cards in order. Have students record their time. Then let the another partner repeat the process.

Challenge 2 - Two Person Sort: At the front of the room display the directions for the second. Run the clock, and have students put the cards in order. Have students record their time. If students are in groups of four offer to let the other two students try the challenge.

Challenge 3 - Full Group Sort: At the front of the room display the directions for the challenge. Run the clock, and have students put the cards in order. Have students record their time.

Debriefing the Challenge

Display: Show the slides explaining the difference between parallel and sequential computing models. Talk through the different models.

Discuss: *What portions of your algorithms for Challenges 2 and 3 were parallel? What makes things complicated or slows you down during parallel portions of your algorithm?*

Discussion Goal: This discussion has two goals. The first is to reinforce the difference between parallel and sequential portions of an algorithm. Any time multiple processes are happening at once (for example multiple people are sorting cards), an algorithm is parallel. The second goal is to bring up some common challenges that come up when running parallel algorithms. The remarks cover some of the most important ones but the main point is that while parallel algorithms are faster, they still present challenges.

Discuss: Have groups discuss responses at their tables before sharing with the room.

Remarks

> A lot of the challenges you just encountered show up when you try to run a program on multiple computers as well.
>
> - Sometimes you need to wait because one computer finished before another
> - It can be complicated to split up work and recombine it when moving in and out of parallel portions
> - They're faster, but not always as much faster as you think.

Discuss: *What was your group's speedup in Challenge 2? What about in Challenge 3? Are you surprised?*

Discussion Goal: Use this discussion to reinforce how speedup is calculated, but also to prime students to realize that adding additional parallel processes doesn't always lead to the same amount of speedup. During the parallel portion things are in fact moving faster, but sequential portions still take a long time (e.g. collecting individual piles once each group member has sorted their cards). Therefore speedup is rarely your original time divided by the number of computers running the process. Eventually it will level off.

Have groups calculate their speedup and share with the room.

Display: Cover the primary points of speedup in the real world.

- Students probably noticed their speedup is lower than the number of people helping sort. Sorting with two people doesn't give a speedup of 2. Sorting with 3 people doesn't give a speedup of 3.
- Because some portions are always still sequential, the benefits of adding more processors will go down and eventually the speedup reaches a limit.

Distributed Computing in Real World Settings

Remarks

> We've just explored some of the core and theoretical ideas of parallel computing. It can speed things up, but not infinitely, and it adds complications and many more resources. That said, parallel computing can help tackle some big problems.

Discuss: Before showing the video share these two questions.

- Why is the type of computing presented “distributed”?
- Why is distributed computing used to solve the problem?

Display: Show the video on Folding at home

Discuss: Have students share their responses to the two questions:

- Why is the type of computing presented "distributed"?
- Why is distributed computing used to solve the problem?

Remarks

> Distributed computing is very similar to parallel computing. The main idea is that programs can be run on lots and lots of computers. Distributed and parallel computing are helpful for solving really big problems that you couldn't normally solve on a single computer.

Wrap Up (10 minutes)

Remarks

> Let's sum up what we learned: Parallel computing consists of both a parallel portion that is shared and a sequential portion.
>
> A sequential solution's efficiency is measured as the sum of all of its steps, but a parallel solution takes as along as its sequential tasks plus the longest of its paralell tasks. Often times a parallel solution will be the fastest option, but there is a limit.
>
> Solutions that use parallel computing can scale more effectively than solutions that use sequential computing. Why is this so? If we continue to add tasks, a sequential solution would continue to get larger and larger. However, with a parallel system, those tasks can be balanced.

Journal: Students add the following vocabulary words and definitions to their journals: sequential computing, parallel computing, distributed computing, speedup

Discuss: *Based on what we saw here today, create a list of pros and cons for distributed and parallel computing. Share it with a partner.*

Discussion Goal: This lesson covers a lot of ground so this is a good chance to review some of the main points of the lesson. Here's some big ones to cover.

- Parallel programs typically are faster
- Parallel programs don't get faster forever. At some point adding more processors doesn't really help
- Parallel programs can be more complicated.
- Parallel programs can be slowed down if only one of many devices is slow.
- Distributed programs can be run on thousands or even millions of computers which allows you to take on enormous problems

Have students write their list, then share with a partner, and then finally discuss responses with the entire room.

Assessment: Check For Understanding

Check For Understanding Question(s) and solutions can be found in each lesson on Code Studio. These questions can be used for an exit ticket.

Question: What is the speedup of this parallel solution?

Question: In your own words, explain why the speedup of a parallel algorithm will eventually reach some limit.

1-2

Check For Understanding

1 2