Improving the Structure and Content of Early CS Courses with Well Aligned, Engaging Learning Materials

Kalpathi Subramanian[1], Erik Saule[1], Jamie Payton[2], and Matthew Mcquaigue[1]
krs@uncc.edu, esaule@uncc.edu, payton@temple.edu, mmcquaig@uncc.edu

[1]The University of North Carolina at Charlotte
[2]Temple University

SIGCSE 2022

Table of Contents

Who are we?

Who are you?

Goals and Objectives

Course Structure

- How should I structure my course?
- What shoud I cover in my course?
- What are other people doing?

Course Content

- How to foster students interest?
- How to make your course appear relevant?
- How to make your course look fun?

We will talk about how curriculum guidelines can help us with building better courses.

We will talk about how to drive student engagement.

We will present two tools to help with those.

- CS Materials
- BRIDGES

Table of Contents

Curriculum Guidelines

What are they?

Usually they are recommendation of what should/could be taught across a program.
Expressed in term of topics, learning outcome, and competencies. Not in term of courses.
Usually make recommendation on how much one should learn in a particular topic, sometimes specified in number of hours.

How can we use them?

Give us a reference of what we should/could be teaching.
Am I covering all that? Should I? Why not?
Give us a common language to communicate between instructors.

General Guidelines: ACM/IEEE CS 2013

Structured in

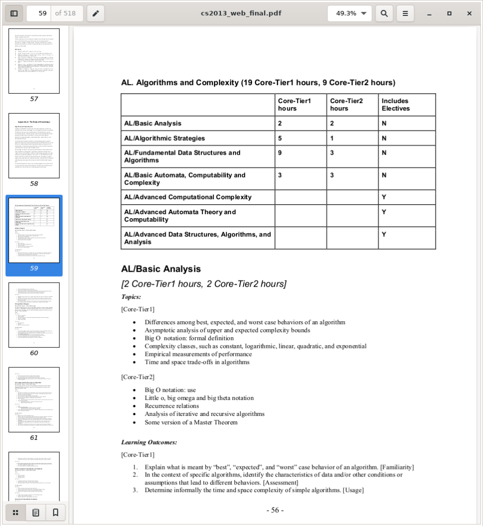
- Knowledge Area
- Knowledge Unit

Topics and Learning Outcomes are classified as

- Tier-1
- Tier-2
- Elective

Other general guidelines:

- Data Science
- Computer Engineering
- Upcoming revised CS



AL. Algorithms and Complexity (19 Core-Tier1 hours, 9 Core-Tier2 hours)

	Core-Tier1 hours	Core-Tier2 hours	Includes Electives
AL/Basic Analysis	2	2	N
AL/Algorithmic Strategies	5	1	N
AL/Fundamental Data Structures and Algorithms	9	3	N
AL/Basic Automata, Computability and Complexity	3	3	N
AL/Advanced Computational Complexity			Y
AL/Advanced Automata Theory and Computability			Y
AL/Advanced Data Structures, Algorithms, and Analysis			Y

AL/Basic Analysis

[2 Core-Tier1 hours, 2 Core-Tier2 hours]

Topics:

[Core-Tier1]

- Differences among best, expected, and worst case behaviors of an algorithm
- Asymptotic analysis of upper and expected complexity bounds
- Big O notation: formal definition
- Complexity classes, such as constant, logarithmic, linear, quadratic, and exponential
- Empirical measurements of performance
- Time and space trade-offs in algorithms

[Core-Tier2]

- Big O notation: use
- Little o, big omega and big theta notation
- Recurrence relations
- Analysis of iterative and recursive algorithms
- Some version of a Master Theorem

Learning Outcomes:

[Core-Tier1]

1. Explain what is meant by "best", "expected", and "worst" case behavior of an algorithm. [Familiarity]
2. In the context of specific algorithms, identify the characteristics of data and/or other conditions or assumptions that lead to different behaviors. [Assessment]
3. Determine informally the time and space complexity of simple algorithms. [Usage]

- 56 -

Specific Guidelines: NSF/IEEE-TCPP PDC 2012

Structured in domains:

- programming
- algorithm
- architecture

More descriptive.
Bloom levels.
Other specific guidelines: graphics, security

Activity

Look at the ACM/IEEE CS 2013 guidelines.
Find some entries relevant to one of your course.
But also browse it to get a sense of the scope of it.
Notice the examplar at the end. Find and read through an examplar for a course similar to what you teach.

Table of Contents

A Lingua Franca

CS Guidelines give us a fairly detailed description of what is in CS.
We can use them as ontologies to describe in a common language what a course of a class material is like.

What do you think is in a lecture entitled UNCC-ITCS-2214-Saule-Graphs?

- Depth- and breadth-first traversals
- Representations of graphs (e.g., adjacency list, adjacency matrix)
- Reflexivity, symmetry, transitivity
- Illustrate by example the basic terminology of graph theory, and some of the properties and special cases of each type of graph/tree.
- Undirected graphs
- Directed graphs
- Weighted graphs
- Iterative and recursive traversal of data structures

Study of Coverage

We can easily understand what one course is covering.
We can understand across multiple offfering of the same course what that particular course is about.
We can identify different “flavors” of that course.

Activity

Look at the different data structure course using the coverage map.
For a particular course:

- Note something they are teaching and that you were not expecting.
- Note something you thought they would cover and are not covering

Look at all courses at once:

- What are the key topics/outcome that are covered by most?

Table of Contents

What is Alignment?

Properties of how content flow in

- program
- course
- module

That could apply to

- topics
- outcomes
- competencies

That could be in term of

- what they cover
- what they assume students know

Aligning Modules with Course Objectives

Courses usually have objectives that come from program descriptions and assessments.
How do we ensure that the content of the class actually serve these higher objective? We want to align the objective modules with the objective of the course.
Two main properties to check:

- Are all the course objectives covered appropriately by a module objective?
- Are there module objectives that serve no course objective?

Alignment within Module

Typical module structure

- Exposition to new concept (lecture, textbook)
- Clarification of concept (discussion, hands-on activity)
- Reinforcement of concept (problem, programming assignment)

Properties you want

- The clarification should not introduce new concepts
- The reinforcement should strengthen the exposition and clarification topics
- The materials should cover the topics the module is meant to cover
- The materials should not wander too far from the module objectives

Assessment

Exam should never introduced new concepts

Activity

For a particular course, look at the lectures and assignment

- Are there topics in the assignment that are not part of the lecture?
  - Do you think it is a problem?
- Are there topics in the lecture that are not in the assignment?
  - Do you think it is a problem?

Consider two sections of data structures

- Can you identify differences between the two sections
  - Are any of this difference style or fundamental?

Table of Contents

Have you ever searched for materials?

Let's look at Nifty Assignments

Nifty Assignments

The Nifty Assignments session at the annual SIGCSE meeting is all about gathering and distributing great assignment ideas and their materials. For each assignment, the web pages linked below describe the assignment and provides materials -- handouts, starter code, and so on.

STANFORD COMPUTER SCIENCE

Applying for Nifty is now done as its own track with a similar deadline to special sessions. The format and content of the .zip you submit is unchanged. See the info page for ideas about what makes a nifty assignment and how to apply for the Nifty session.

Please email any suggestions or comments to the nifty-admin email: nifty-admin@cs.stanford.edu
Nick's Home

Nifty Assignments 2021

Sankey Diagrams - Ben Stephenson	CS1 Sankey diagram - neat data visualization algorithm
Rocket Landing Simulator - Adrian A. de Freitas and Troy Weingart	CS1 Rocket Landing Simulator - fun algorithm
Covid Simulator - Steve Bitner	CS1-CS2 Covid 2D infection simulator - timely if scary
Linked List Labyrinth - Keith Schwarz	CS2 Neat memory / debugger skill exercise, custom per student

Nifty Assignments 2020
Thanks to our presenters for getting everything together including videos for this COVID-interrupted year.

Typing Test - John DeNero et al	CS1 Fill in algorithm of fun typing-speed test. **(Video)** (intentionally silent)
Color My World - Carl Albing	CS1 or later: Students are given a data file, but no description about what it represents. Can they solve the mystery by generating a reasonable image?
Bar Chart Racer - Kevin Wayne	CS1 - use real data to make a animated bar chart - captivating! **(Video)**
DNA - Brian Yu, David J. Malan	CS1 or CS2 Neat DNA project. **(Video)**
Recursion to the Rescue - Keith Schwarz	Nifty recursion projects using tied to real-world applications. **(Video)**
Decision Makers - Evan Peck	Two hour exercise illuminating algorithms and life

Nifty Assignments 2019

Nifty Post It - Jeffrey L. Popyack	CS0-CS1 Hands On Manipulative
Hawaiin Phonetic Generator - Kendall Bingham	CS1 Fun Text
Motion Parallax - Ben Dicken	CS1 Awesome Graphic Experience
Gerrymandering - Allison Obourn	CS1-CS2 Election Data Analysis and Visualization
Code Crusher - Ben Stephenson	CS1-CS2 Great Popular Game + Code
Blocky - Diane Horton and David Liu	CS2 Recursion Tree Fabulous

Metadata

Summary	Students develop a program to map raw data files into a colorful images.
Topics	visualization, big data, image processing - color maps.
Audience	Use as an early assignment in an HPC class, Scientific Programming class, Data Science/Analysis class, or a Graphics/Image processing class. Appropriate for CS1 or higher students familar with loops, file io, argument parsing, and image processing. The starter code is written in Python.
Difficulty	This assignment is appropriate for various levels, depending on the initial conditions: starter code (or not), existing color maps (or not) and time alloted. A late-semester CS1 class given the starter code and a week.
Strengths	• Solving the mystery of what the image "looks" like • Working with *real-world* data to get visual, graphical feedback. • Allows for some artistic flair resulting in variations among solutions • Depending on the assignment write up there are open ended options including: ◦ creating different colormaps for different images; ◦ scaling the data to fit a given image size; ◦ a "smarter" program to deduce the image size from the data file; ◦ statistical analysis of the data to drive the choice of color map values
Weaknesses	• When creating a colormap from scratch it can be tricky to get color assignments that are both visually pleasing (artistic) and pull out the desired details, though that is part of the point of this assignment. • Use of graphics makes unit testing more challenging.
Dependencies	• if statement • loops • reading files • elementary graphics concepts

Curriculum Guidelines as Features

Features

The problem in classic search is that it is hard to find good matches because people use imprecise textual descriptions.
Curriculum guidelines give us a well established precise features

Search

Give a set of materials that use these topics/outcomes

Recommendation

Give a set of materials that match the same outcomes as these ones.

Activity

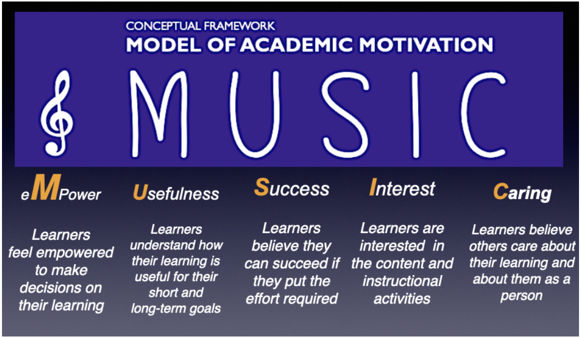
Can you find materials about hash tables?
Can you find materials about shortest path?

Table of Contents

Table of Contents

Engagement and Motivation

- Well understood that student engagement and motivation can drive student success.
- Engagement and motivation are closely tied to each other
- How do we motivate and engage students? Many models have been proposed, such as the MUSIC model of motivation (Jones, 2009)



Engaging Students: Experiences from an OOP Course

Two semesters of a project based OOP course, using student reflections after each course module

- e**M**powerment: Project choice, freedom to be creative, experimentation and tinkering
- **U**sefulness: Working with real-world data/tools, team environment
- **S**uccess: Assignments with clear instructions, predictability, reflect on personal successes/failures, feedback
- **I**nterest: Fun factor, games, real world images used as part of course
- **C**aring: Sensitive to student needs, prompt feedback, deadline flexibility

Activity

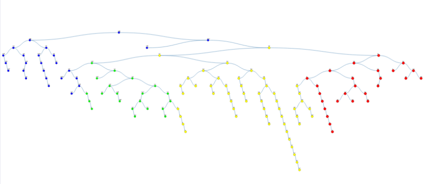
Thoughts on engaging students as part of course activities?

Table of Contents

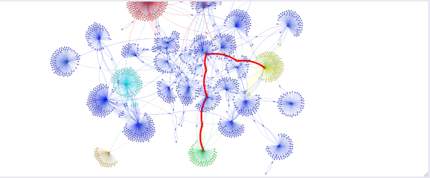
Making it Interactive/Visual

- Visualizations of classic CS concepts can be helpful in making them real and more meaningful.
- Complex data structures and algorithm concepts can be explained better with visualizations.
- Interactive applications is a more attractive approach to experimentation - changing parameters to see its effect on a phenomenon, solution, performance.

Indexing USGS Earthquake



Bacon Number [IMDB Data]



Activity

- Review BRIDGES tutorials

Table of Contents

Make it Real!

- Using real-world data in course work is an important engagement tool
- Students respond to data from different real-world scenarios and appreciate the use of images, maps, games
- Data is everywhere, the harder part is
  - Accessing data in a ready-to-use form for course work
  - Mapping the right data to course work to meet objectives.
- Example: A BRIDGES example for retrieving Earthquake records

```
// create Bridges object
// command line args provide credentials and server to test on
Bridges bridges (atoi(argv[1]), argv[2], argv[3]);
if (argc > 4)
    bridges.setServer(argv[4]);
// set title
bridges.setTitle("Accessing USGIS Earthquake Data (USGIS Data)");

// read the earth quake  data
DataSource ds (&bridges);
vector<EarthquakeUSGS> eq_list = ds.getEarthquakeUSGSData(max_quakes);

// print the first quake record
```

Activity: BRIDGES Data Access and Assignments that use real-world data and Visualizations

- Accessing Earthquake Data
- Bacon Number Computation (Graph BFS)
- OpenStreet Map (Graphs - Shortest Path)
- Image Representation/Compression (Spatial Search Trees - Kd-Tree)
- Algorithm Benchmarking - Comparing Sorting Algorithms

Table of Contents

The Power of Choice

Providing choices in learning materials (lectures, assignments, etc.) provides flexibility and choice for students as they might have different preferences/interests

- Challenge: Designing multiple assignments that meet the same learning objectives does involve a higher load on instructors
- Examples:
  - Assignments that can use different real-world datasets
  - Different assignments that rely on the same underlying algorithm
- Choice in learning materials has shown in prior work being appreciated by students.

Activity

- Group 1 (Different datasets):
  - Linked list using IMDB data
  - Linked list using USGS Earthquake data
- Group 2 (Different assignments, same algorithm)
  - Bacon Number (Graph - BFS)
  - Maze Solution (2D array - BFS)

Table of Contents

What we can do to help you

Classify a course in CS Materials

It can help you:

- identify gaps in coverage
- better align your course
- find new materials to use
- share your cool materials.

Integrate BRIDGES in a course

- motivate your students
- access real datasets
- easily generate visualization
- adopt engaging assignments

We have stipends for adopters and will run 3-day workshops over summer on both tools.

Activity

Making Plans

What do you think are the points in your class that could use improvement?
What would you need to do to perform this?
What is a realisitic time frame to make these improvements?

Plan, discuss, and and report.

Table of Contents

Surveys

SIGCSE survey
Paula's survey