# Code Like The Real World

## - An Interdisciplinary Approach to Introductory Computer Science -

**Lisa Wang, Stephen Cooper**

*Stanford University, Department of Computer Science*

## Abstract

Each year, over 1,500 students representing a large spectrum of majors and interests take an introductory computer science course at Stanford.
Since computer science has become an increasingly interdisciplinary field, both majors and non-majors could benefit from an early exposure to a variety of computational challenges.
We hypothesize that teaching interdisciplinary problems from the real world will motivate students to learn the material, help them remember abstract concepts and increase their awareness of computer science as a broad discipline with applications to other fields.

## Introduction

In order to cope with the large enrollments in introductory CS courses, the Stanford Computer Science Department has introduced the Section Leading Program. Instead of graduate students, advanced undergraduate students teach lab sessions in a small seminar setting to about ten students and grade assignments[1]. Every week, the section leader works through a few section problems related to the material taught in class, but most of these have not been interdisciplinary problems.

Since sections are very flexible and allow for in-class discussion, they present a great setting to introduce students to interdisciplinary fields. In addition, both section leaders and students come from a variety of backgrounds, allowing majors and non-majors to inspire each other.

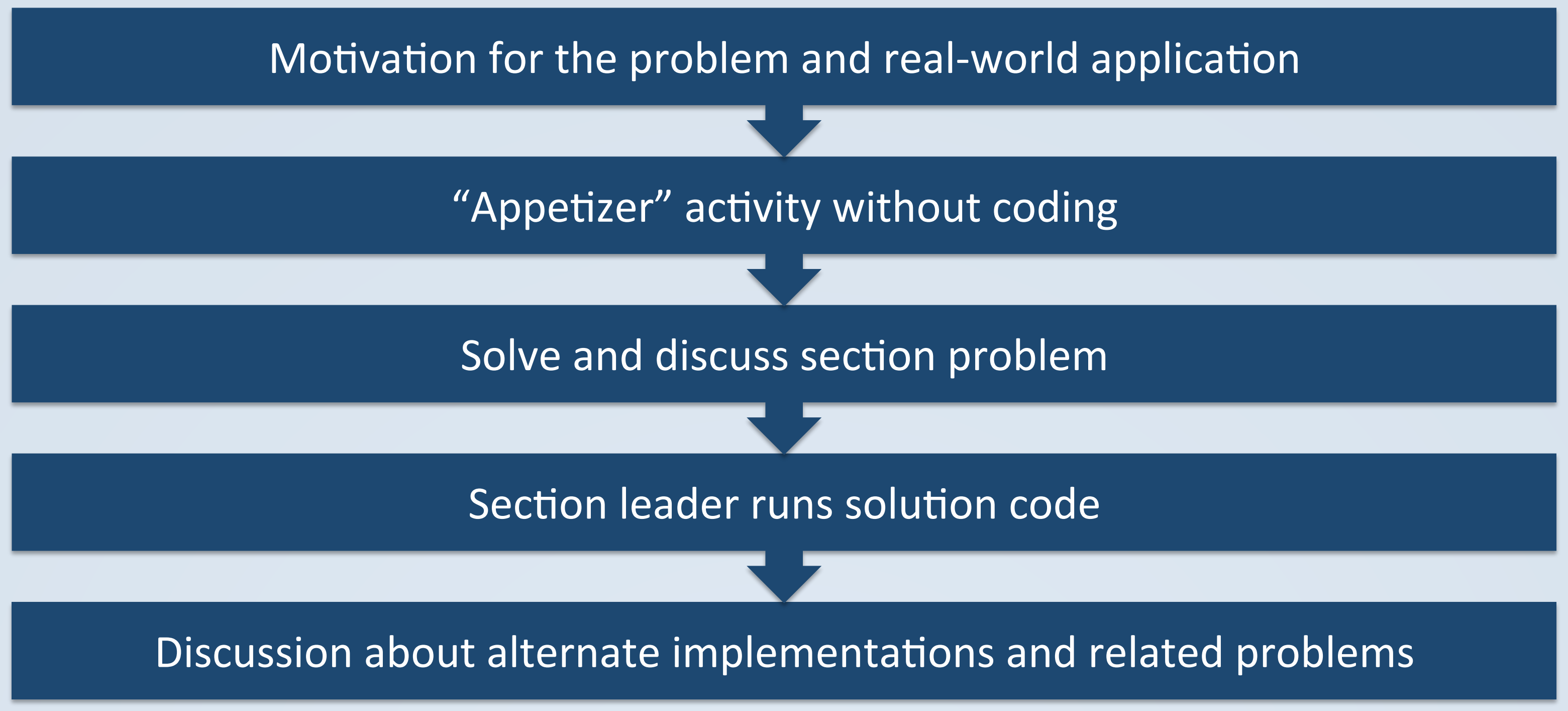
## Goals

- Show how programming is relevant to other disciplines
- Introduce students to computer science related research topics
- Increase section attendance
- Motivate students to learn the material
- Pitch more advanced computer science courses
- Promote more discussion between students and their section leader
- Inspire interest in the course and computer science

## Approaches

We studied the section problems and materials for CS 106B from the past two years, summarized the most important concepts covered in sections and created engaging problems that approach critical CS 106B concepts (e.g. recursion) from an interdisciplinary perspective.

## Revised Section Outline



Motivation for the problem and real-world application

"Appetizer" activity without coding

Solve and discuss section problem

Section leader runs solution code

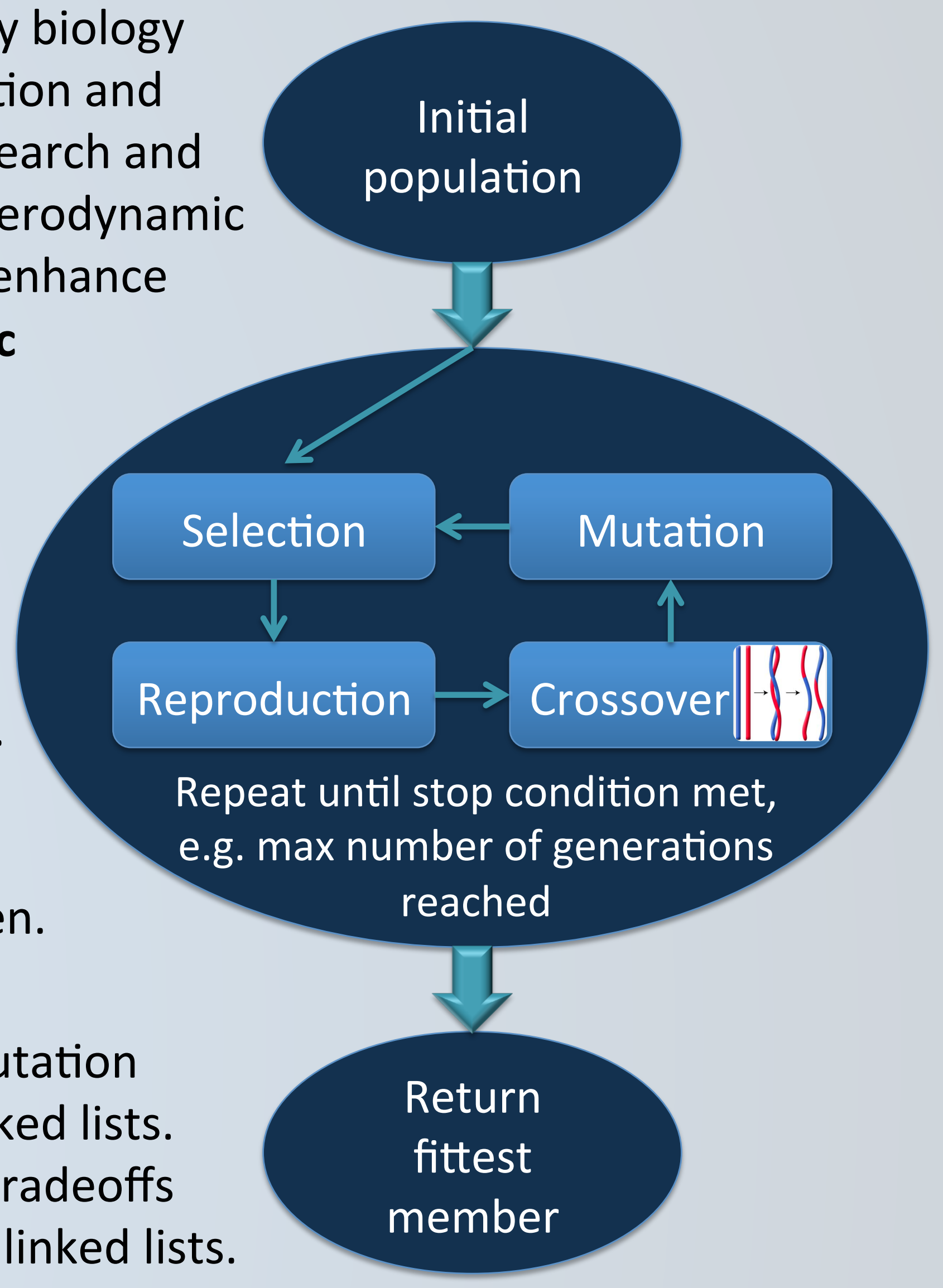
Discussion about alternate implementations and related problems

## New Section Problems

For each section, we have prepared a handout for students, a handout with notes specifically for section leaders, a solution handout and a project file with commented C++ code.
The following table provides an overview of the sections created this summer.

| Section Title | Problem | CS 106B Concept | Interdisciplinary Aspects and Pitched CS Courses |
|---|---|---|---|
| Self-Driving Cars: Localization | Given a grid world with cells in two colors, determine the location of the robot by a sequence of moves and sensor measurements. | Grid | Mechanical Engineering, Robotics<br>CS 109, CS 223A |
| Self-Reference and Recursion | Write a self-referential statement and a program that outputs itself (quine). | Recursion | Philosophy<br>CS 103 |
| Genetic Sequence Alignment | Find the best alignment of two DNA strands. | Recursive Backtracking | Biology, Genetics<br>CS 161, CS 273A, CS 274 |
| Take The Right Turn | Given a world with streets, find the fastest route from A to B by reducing left turns. | Recursive Backtracking | Environmental Science<br>CS 103, CS 161 |
| Sentiment Graphs | Starting with a small set of classified adjectives (positive or negative), classify more adjectives in the synonym graph. | Graph Algorithms, Graph Modeling | Linguistics, NLP<br>CS103, CS 124, CS 224N, CS 229 |
| Genetic Algorithms (GAs) | Model chromosomes as linked lists, implement crossover and mutation, two essential steps in GAs. | Linked List, Data Structure Tradeoffs | Biology, Genetics<br>CS 274, CS 161, CS 166, CS 334A |

## Example Section: Genetic Algorithms



**Genetic algorithms (GA)** are inspired by biology and mimic the process of natural selection and evolution. They are heuristics used in search and optimization problems, e.g. to create aerodynamic **race car designs**, to generate **puns**, to enhance **encryption** or to find solutions to **traffic routing problems** like the NP-hard Traveling Salesman Problem.

**Evolution in GAs (see flowchart):**
Each generation consists of a set of chromosomes, which are modeled as linear structures consisting of genes. New chromosomes are created by **crossing over** old chromosomes. In addition, random **mutations** can happen.

**Section problem:**
a) Implement the crossover and the mutation steps for chromosomes modeled as linked lists.
b) Discuss with your partner what the tradeoffs would be, if you used arrays instead of linked lists.

**What students learn:**

- An interdisciplinary approach to improve slow algorithms
- Linked list operations
- Tradeoffs between different data structures

**Which topics this section problem motivates:**

- Biocomputation (CS 274 / BMI 214 / GENE 214)
- Algorithms and complexity theory (CS 103, CS 161)
- Data structures (CS 166)
- Optimization problems (CS 334A / EE 364A)

In addition, the handout provides interested students with further readings, e.g. related Science Daily articles and introduces higher-level courses they can take in the future.

## Future Work / Planned Study

We are planning to run a study during the 2014/2015 academic year. We will expose a subset of CS 106B sections to the new interdisciplinary problems and compare student satisfaction, motivation, engagement and performance to the other sections. In addition, we would like to measure how many students take another CS course. We hope that our study will show whether an inter-disciplinary approach to teaching introductory computer science is effective.

## Acknowledgements

This research project has been sponsored by Professor Stephen Cooper and was partially supported through NSF - 1031351.

## References

[1] Eric Roberts , John Lilly , Bryan Rollins, Using undergraduates as teaching assistants in introductory programming courses: an update on the Stanford experience, ACM SIGCSE Bulletin, v.27 n.1, p.48-52, March 1995 [doi>10.1145/199691.199716]