Pedagogical Implications of Parser Combinators in Programming Languages Courses: A Comparative Study

Abbas Attarwala[1], Pablo Raigoza[2]

About Us

Dr. Abbas Attarwala[1]

Associate Professor
Department of Computer Science: California State University, Chico

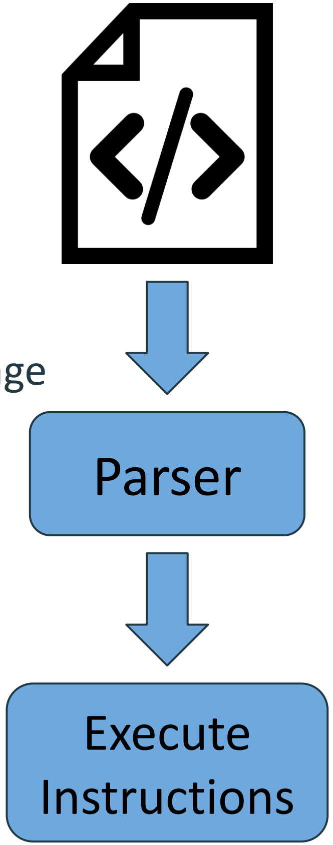
Pablo Raigoza[2]

Undergraduate Student
Computer Science
Cornell University

Motivation

Motivation: Setup

- Asked students to implement stack based programming language
- What challenges does this entail?
  - Students are required to parse large code files
  - Take text code, decipher the text, execute instructions

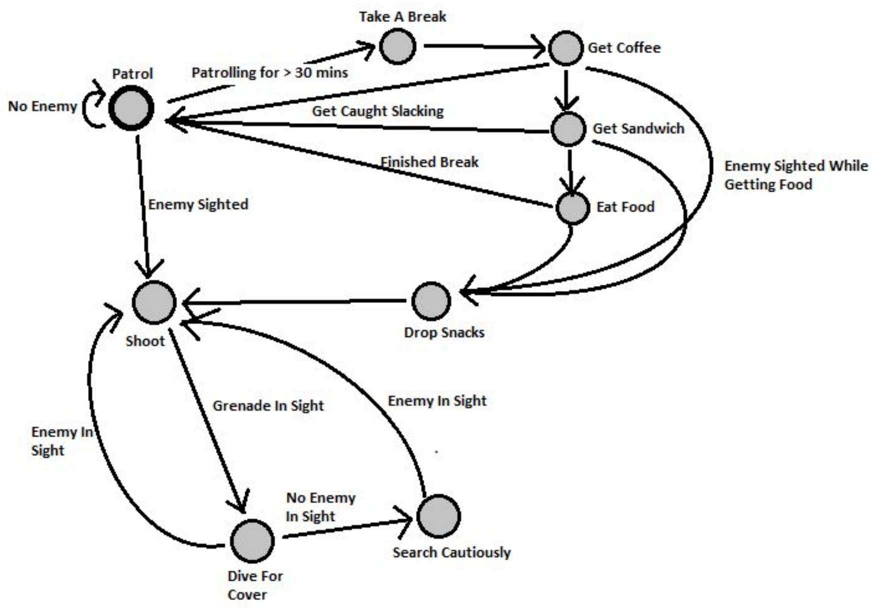


Motivation: Problem

- Student wrote highly specialized *ad hoc* solutions
- **Works** for simple languages
- **Fails** to generalize for complex languages

What if we want to generalize diagrams for deputy behaviors? Sheriff behaviors? Graph completely changes

Complex State Diagrams!



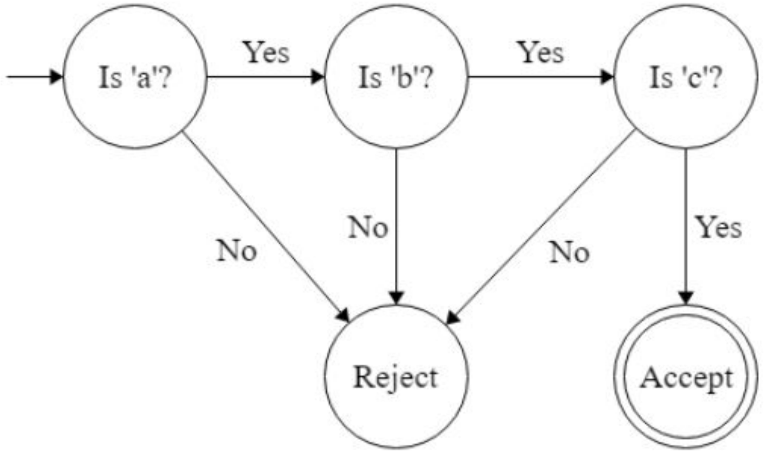
Motivation: Problem (example)

Task: Parse out phrase ‘abc’ from string

Ad hoc Solution Code

```
let parse s =
  match ( getFirstCharacter s) with
  | None -> None
  | Some ( firstC , rest ) -> if firstC = 'a' then
    ( match ( getFirstCharacter rest ) with
    | None -> None
    | Some ( secondC , rest ) -> if secondC = 'b' then
      ( match ( getFirstCharacter rest ) with
      | None -> None
      | Some ( thirdC , rest ) -> if thirdC = 'c' then
          Some ( true , rest )
          else None )
    else None )
  else None
```

State machine



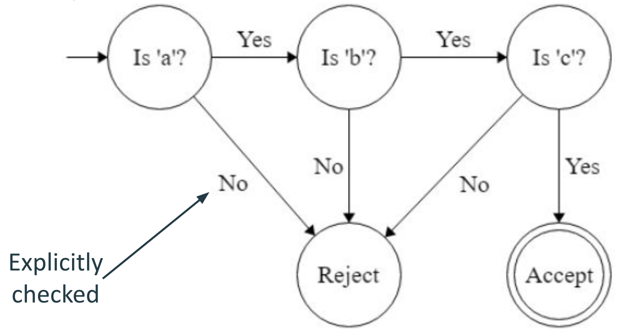
Q: What are some problems with this approach?

Motivation: Problem (example)

Task: Parse out phrase ‘abc’ from string

Problems

- *Explicit* error checks with adjacent characters
- Scaling issues
    - What if we want to accept for ‘aabbcc’?
    - What if we want to accept more complicated expressions?
        - Code can quickly bloat



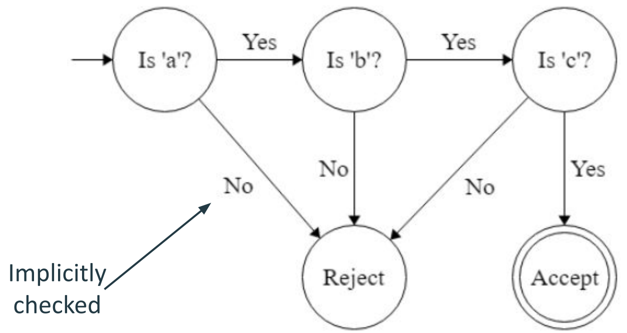
```
let parse s =
  match ( getFirstCharacter s) with
   | None -> None
   | Some ( firstC , rest ) -> if firstC = 'a' then
     ( match ( getFirstCharacter rest ) with
     | None -> None
     | Some ( secondC , rest ) -> if secondC = 'b' then
       ( match ( getFirstCharacter rest ) with
       | None -> None
       | Some ( thirdC , rest ) -> if thirdC = 'c' then
           Some ( true , rest )
           else None )
     else None )
  else None
```

Motivation: Problem (solution)

- Parse with **parser combinators**
- Parser
  - Input: string
  - Output: (a', string) option
- Satisfy
  - Verfies string starts with 'x' character
- p_1 >> p_2
  - Executes p_2 only if p_1 accepts
  - Links pairwise combinators, p_1 and p_2
  - Implicitly handles errors

Implicit error handling!

```
let parse =
    satisfy (fun c -> c = 'a') >>
    satisfy (fun c -> c = 'b') >>
    satisfy (fun c -> c = 'c')
    (* return true *)
```



Contribution: Teaching Methods

Contribution: Teaching Methods

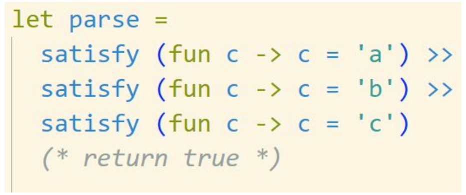
What's been done?

- Extensive research in exploring technical advantage of parser combinator
- Pedagogical aspects have been largely overlooked

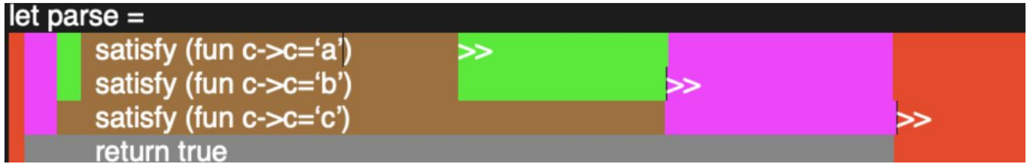
Why focus on teaching?

- Strictly adding more content to a course is not realistic
- Making the assignments *easier* by adding more content is counterintuitive
  - Is this even possible?

Contribution: Teaching Methods



```
let parse =
  satisfy (fun c -> c = 'a') >>
  satisfy (fun c -> c = 'b') >>
  satisfy (fun c -> c = 'c')
  (* return true *)
```



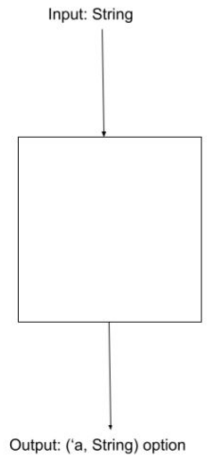
```
let parse =
    satisfy (fun c->c='a')        >>
    satisfy (fun c->c='b')              >>
    satisfy (fun c->c='c')                    >>
    return true
```

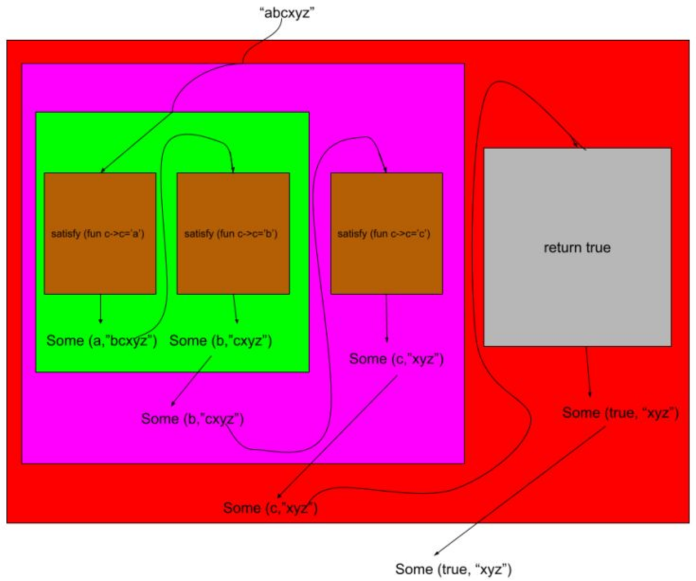
Color Coding / Boxes

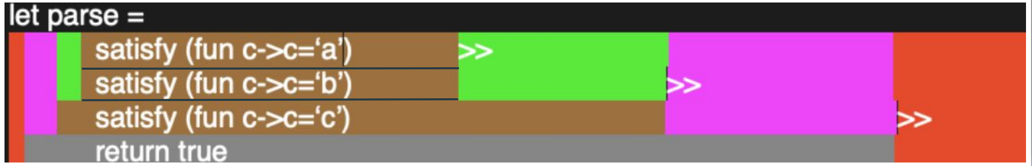
- Applied color boxes to the code to partition different sections of the code
- **Why this method?**
  - OCaml is a functional programming language
  - Everything is modular functions of input / output
  - Color coding / boxing is a natural visual representation

Contribution: Teaching Methods

- Visually see recursive behavior of how the ‘>>’ combinator iteratively builds out the larger combinator
- Clear input / output relations that showcases parser combinator modularity







```
let parse =
    satisfy (fun c->c=‘a’) >>
    satisfy (fun c->c=‘b’) >>
    satisfy (fun c->c=‘c’) >>
    return true
```

Data: Course Evaluation

Data: Course Evaluation

- **Question:** How can we verify whether using code color boxing was an effective teaching method? Did it confuse the students?
- To answer this question we used course evaluation data

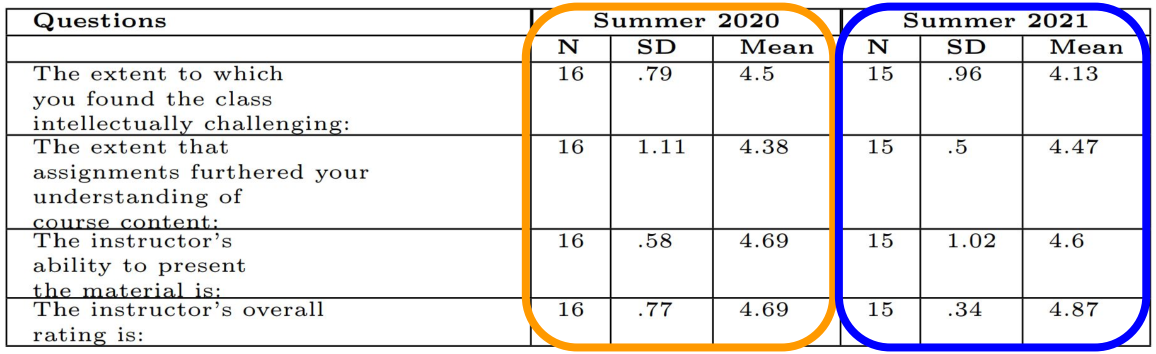
Taught over **three** summer sessions

- Boston University
  - Summer 2020, without parser combinators
  - Summer 2021, with parser combinators
- California State University, Chico
  - Summer 2023, with parser combinators

Data: Course Evaluation

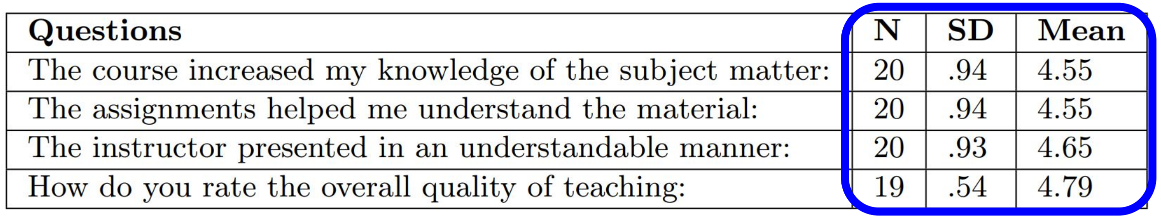
- 🟧 Without parser combinators
- 🟦 With parser combinators

BOSTON UNIVERSITY



Questions	Summer 2020			Summer 2021		
	N	SD	Mean	N	SD	Mean
The extent to which you found the class intellectually challenging:	16	.79	4.5	15	.96	4.13
The extent that assignments furthered your understanding of course content:	16	1.11	4.38	15	.5	4.47
The instructor's ability to present the material is:	16	.58	4.69	15	1.02	4.6
The instructor's overall rating is:	16	.77	4.69	15	.34	4.87

California State University Chico



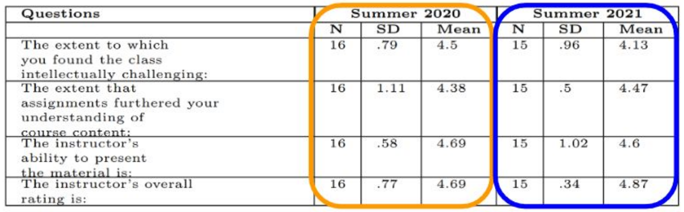
Questions	N	SD	Mean
The course increased my knowledge of the subject matter:	20	.94	4.55
The assignments helped me understand the material:	20	.94	4.55
The instructor presented in an understandable manner:	20	.93	4.65
How do you rate the overall quality of teaching:	19	.54	4.79

Data: Course Evaluation

- The data did differ across two different universities
  - Increases variance, but does capture wider range of audience
- Could have used more data across full semesters
  - Decided to keep length of classes as a control (6-week summer classes)

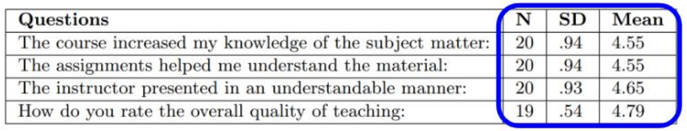
🟧 Without parser combinators
🟦 With parser combinators

BOSTON UNIVERSITY



Questions	Summer 2020			Summer 2021		
	N	SD	Mean	N	SD	Mean
The extent to which you found the class intellectually challenging:	16	.79	4.5	15	.96	4.13
The extent that assignments furthered your understanding of course content:	16	1.11	4.38	15	.5	4.47
The instructor's ability to present the material is:	16	.58	4.69	15	1.02	4.6
The instructor's overall rating is:	16	.77	4.69	15	.34	4.87

California State University Chico

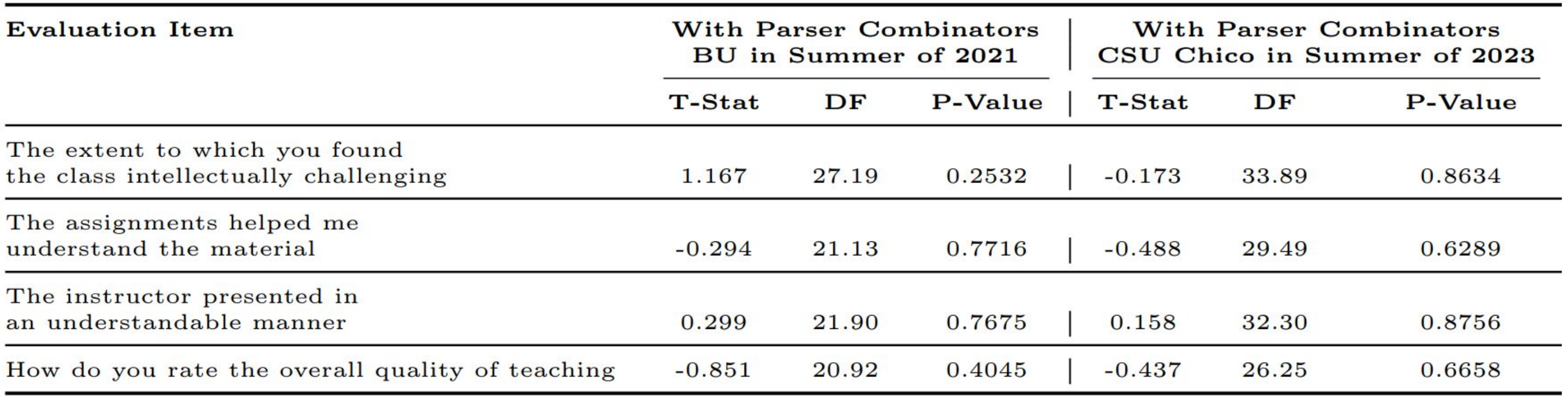


Questions	N	SD	Mean
The course increased my knowledge of the subject matter:	20	.94	4.55
The assignments helped me understand the material:	20	.94	4.55
The instructor presented in an understandable manner:	20	.93	4.65
How do you rate the overall quality of teaching:	19	.54	4.79

Results

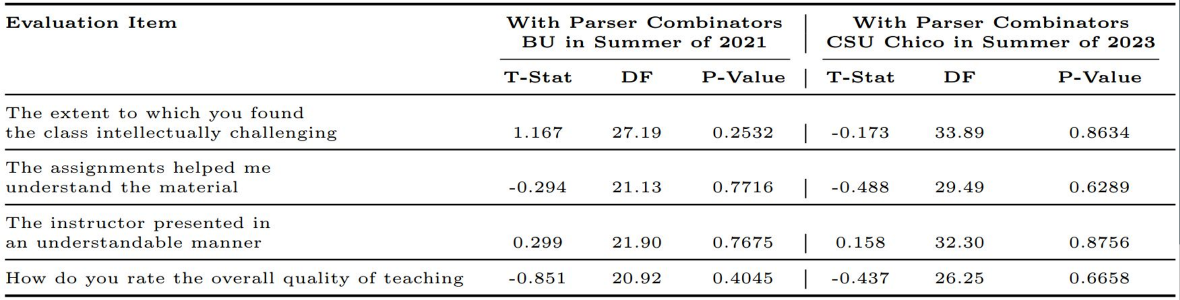
Results

- Applied Welch’s two-tailed t-test and not the Student’s t-test
- Student’s t-test assumes same variances across groups
  - This assumption does not hold
- Unfortunately no statistically significant results



Evaluation Item	With Parser Combinators BU in Summer of 2021			With Parser Combinators CSU Chico in Summer of 2023		
	T-Stat	DF	P-Value	T-Stat	DF	P-Value
The extent to which you found the class intellectually challenging	1.167	27.19	0.2532	-0.173	33.89	0.8634
The assignments helped me understand the material	-0.294	21.13	0.7716	-0.488	29.49	0.6289
The instructor presented in an understandable manner	0.299	21.90	0.7675	0.158	32.30	0.8756
How do you rate the overall quality of teaching	-0.851	20.92	0.4045	-0.437	26.25	0.6658

Results



Evaluation Item	With Parser Combinators BU in Summer of 2021			With Parser Combinators CSU Chico in Summer of 2023		
	T-Stat	DF	P-Value	T-Stat	DF	P-Value
The extent to which you found the class intellectually challenging	1.167	27.19	0.2532	-0.173	33.89	0.8634
The assignments helped me understand the material	-0.294	21.13	0.7716	-0.488	29.49	0.6289
The instructor presented in an understandable manner	0.299	21.90	0.7675	0.158	32.30	0.8756
How do you rate the overall quality of teaching	-0.851	20.92	0.4045	-0.437	26.25	0.6658

Bust? No!

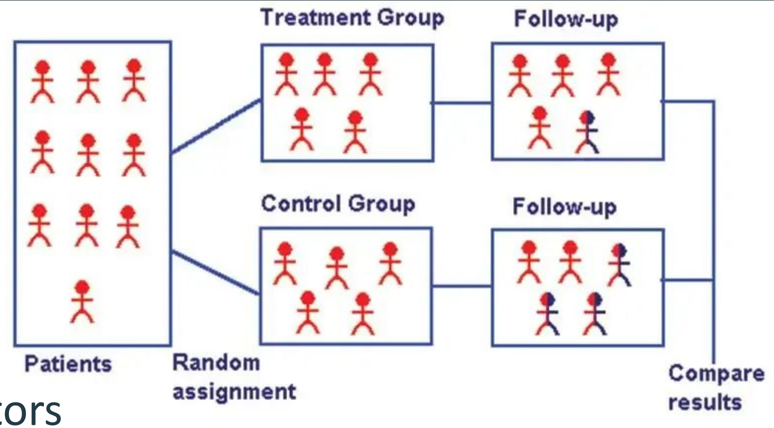
- Many of the survey results **improved** slightly or held constant
  - Adding more content to a course can make it more difficult for students
  - Increases perceived difficulty / frustration
- Can be successful because of the fact evaluations didn’t drastically decrease

Anecdotal Responses | Anonymous student responses

- *“[Professor Attarwala’s] color coding, visualizations, and reinforcements really drilled in the material”*
- *“Good visuals pointers for the current material that was talked about”*

Future Improvements

Future Improvements



Randomized Trial Control Experiments (RTCE)

- Half students don't learn parser combinators
- Half students learn parser combinators
- Compare course evaluations and midterm / final scores

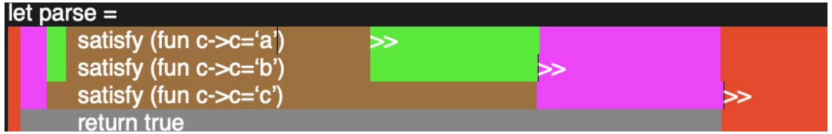
Why is this better?

- Semester to semester is different
  - Length, morning vs afternoon class, average student competency, etc

Requires Institutional Review Board (IRB) approval due to moral concerns

Take-Aways

Take-Aways



```
let parse =
    satisfy (fun c->c='a')      >>
    satisfy (fun c->c='b')           >>
    satisfy (fun c->c='c')                 >>
    return true
```

1. Parser combinators are modular and can easily be generalized
2. Anecdotally, students enjoy visual color coding / boxes examples
3. Even with no statistically significant results, keep parser combinators
   a. Color coding / boxes did not negatively impact perceived course enjoyment
4. Addresses a gap in current literature
   a. Pedagogical aspects not studied as well for parser combinators
5. Ideally use Randomized Trial Control Experiments
   a. Impossible without proactive IRB approval

Thank you!

Questions?