Pair Programming in the Classroom

Mark Sherriff
University of Virginia

June 29, 2016

Some material courtesy of Laurie Williams, NCSU

Overview

- What exactly is Pair Programming?
- The Case for Pair Programming
- The Costs
- Guidelines for a successful pairing experience
- Myths and Legends
- Resources

Pair Programming Definition

- "Pair programming is a style of programming in which *two* programmers work side-by-side at one computer, continuously collaborating on the same design, algorithm, code, or test."
  - Laurie Williams

Slightly Altered Definition

- "Pair programming is a style of programming in which *two* programmers work side-by-side ~~at one computer~~, continuously collaborating on the same **design or algorithm.**"
  (emphasis mine)
- Basic idea: IDE's help us code – people help us design!

Why Pair Programming?

- Pair programming students tend to:
  - Make it through the first class
  - Improves retention
  - Increases programming confidence
  - Perform comparably or better on exams and projects
  - Perform just fine in future solo programming
  - Help create peer groups

Why Pair Programming?

- An instant support system
  - We have found that pairing cuts down on a large number of the "trivial" questions (syntax, assignment clarification, etc.) and a fair number of the more complex questions (debugging, etc.)
  - We have been able to reduce the number of TAs for some courses
  - Instructor office hours are much quieter, and the instructor can spend more time with students that need more help

Why Pair Programming?

- Sometimes it **is** a numbers game
- In a lab of 40 students…
  - having 20 pairs makes it easier for TAs to get to everyone
  - 20 assignments are easier/faster to grade than 40
- Our main CS1 course has on average 500 students a semester…

The Roles

- The Driver
  - The person with "control" of the computer
  - Does the bulk of the typing
- The Navigator
  - Actively follows along with the driver with comments
  - Can take over at any time
- How does this translate to pair design?

Partners vs. Pair Programming

- How is Pair Programming different than just having partner assignments?
  - Mentality of how to approach the assignment
- Partnering:
  - "You go do this part and I'll go do this part and then we'll put it back together."
- Pair Programming:
  - "Let's first do this part together, then we'll tackle the rest."

Partners vs. Pair Programming

- The distinction matters!
- It matters to:
  - Instructors
  - Teaching Assistants / Tutors
  - Students
- Call it framing, perception, spin… whatever
- It's all about attitude!

It’s All About Attitude

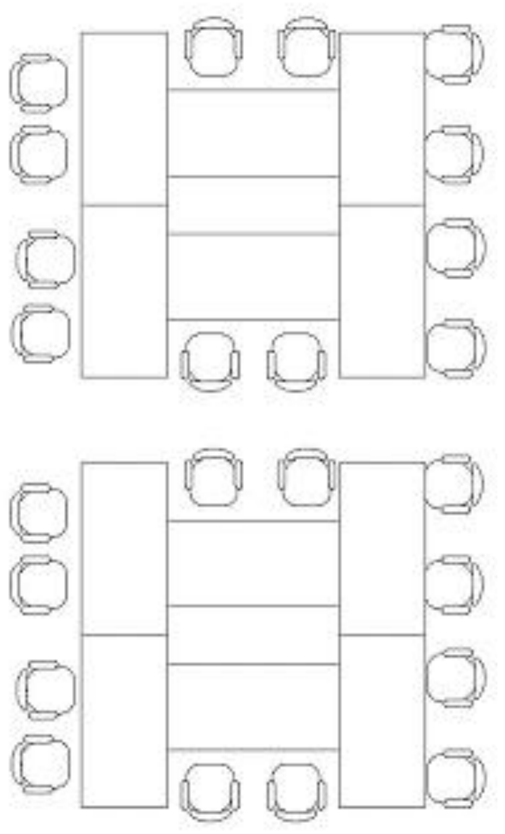
- How do you get the attitude going?
- How do I start using pair programming?
- Things to consider:
  - Teaching the Technique
  - Assignments
  - Pair Creation
  - Pair Evaluation
  - Assessment

Teaching the Technique

- Start with the instructional staff
- Pair programming HAS to be incorporated into the class (or lab) in some structured way
- Students do not naturally work as a "pair" when given a "partner"
- What happens when you tell students they can work with a "partner"?

Teaching the Technique

- The environment matters!



Teaching the Technique

- What are you actually teaching them to do?
- 1. Take turns being the one coding (“driver”)
- 2. Whoever is not coding, comment actively
- 3. Whoever is coding, talk through what you are doing
- 4. Switch at regular intervals
- 5. Nothing is done independently from the other partner

Teaching the Technique

- Switching roles can be problematics
- Some ideas:
  - Go around and tap people on the shoulder
  - Have a audio cue
  - Have a visual cue
- Try to enforce even roles as much as possible
- Try to enforce no "splitting up work" as much as possible

Assignments

- Do I have to totally change my course material to do pair programming?
- Answer: Probably not, but some changes might make things go better

Assignments

- Biggest problem: assignment scope
- If you use your current assignments with no modification at all, it's possible that no switching will occur and/or the point of pairing won't be obvious
- Example: Convert Fahrenheit to Celsius
- Counter Argument: Two novices learning together from the very beginning could help with self-confidence

Assignments

- If the assignment scope is too large or if there is an obvious “split point”, divide and conquer becomes more tempting
- Example: Write a Student and Course class that work together to keep up with course enrollment

Assignments

- An assignment I like for pair programming:
- Email Hunt
  - Given a website that has a bunch of email addresses on it, write a program that can read the website and extract the email addresses
  - http://cs1110.cs.virginia.edu/emails.html
- Things I like:
  - No one way to do it (in fact, it takes more than one idea to get all the emails out)
  - Allows for some creativity

Pair Creation

- How do you create partners?
- Big philosophic question:
  - Do you assign partners or do you let students pick their own partners?
  - Advantages and disadvantages to both

Pair Creation – Assigned Pairs

- How can you assign pairs?
  - Randomly
  - Based on programming experience / confidence
  - Personality / friendships
  - Other interests / survey results

Pair Creation – Assigned Pairs

- Randomly
  - Easiest to setup
  - Good if you have no other information to work from
  - Has potential to lead to problems (but not as many as you might think)
  - Consider "random with replacement" for subsequent assignments (no one can work with same person twice)

Pair Creation – Assigned Pairs

- Based on programming experience / confidence
  - Research indicates this has the highest likelihood of producing good partnerships
  - Hard to setup until you have data
  - Even then, it can be difficult because research shows that *perception* of partner's ability (not *actual* ability) is a higher indicator of a good match

Pair Creation – Assigned Pairs

- Personality / Friendships
  - Most likely to have the fewest personality conflicts
  - Enforcing cliques
- Other survey results
  - I haven't used anything else, but could imagine using things like:
    - Schedule
    - Outside interests
    - Common friends

Pair Creation – Self-Selected Pairs

- Self-selected pairs often have elements of the assigned pairings with similar experience and friendships
- So it has similar benefits and drawbacks
- However, you HAVE to monitor closely for the “last student picked” problem
- Probably should enforce replacement for later assignments

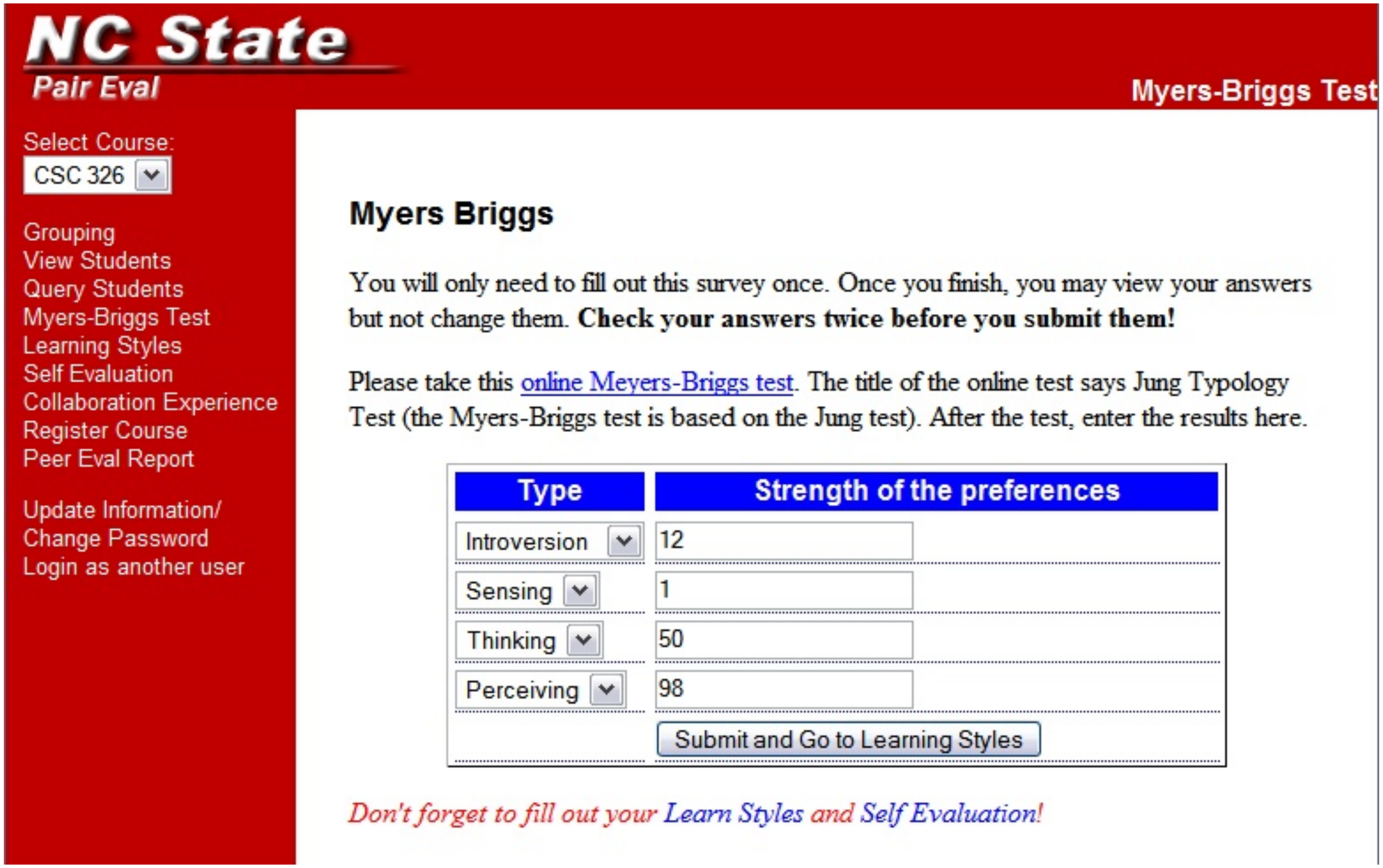
Pair Replacement

- Reassign several times per semester
- Good for students
  - Get to meet new people, learn about working with new people
  - If they don't like their partner, they know they will get a new one soon
- Good for instructor
  - Multiple forms of feedback
  - Natural handling of dysfunctional pairs

Pair Management and Evaluation

- Auto-Assign Pair Creation
  - CATME – http://www.catme.org
  - Data needed to auto-create pairs varies
- Self-Reported Pairs
  - Google Forms

PairEval



Myers Briggs

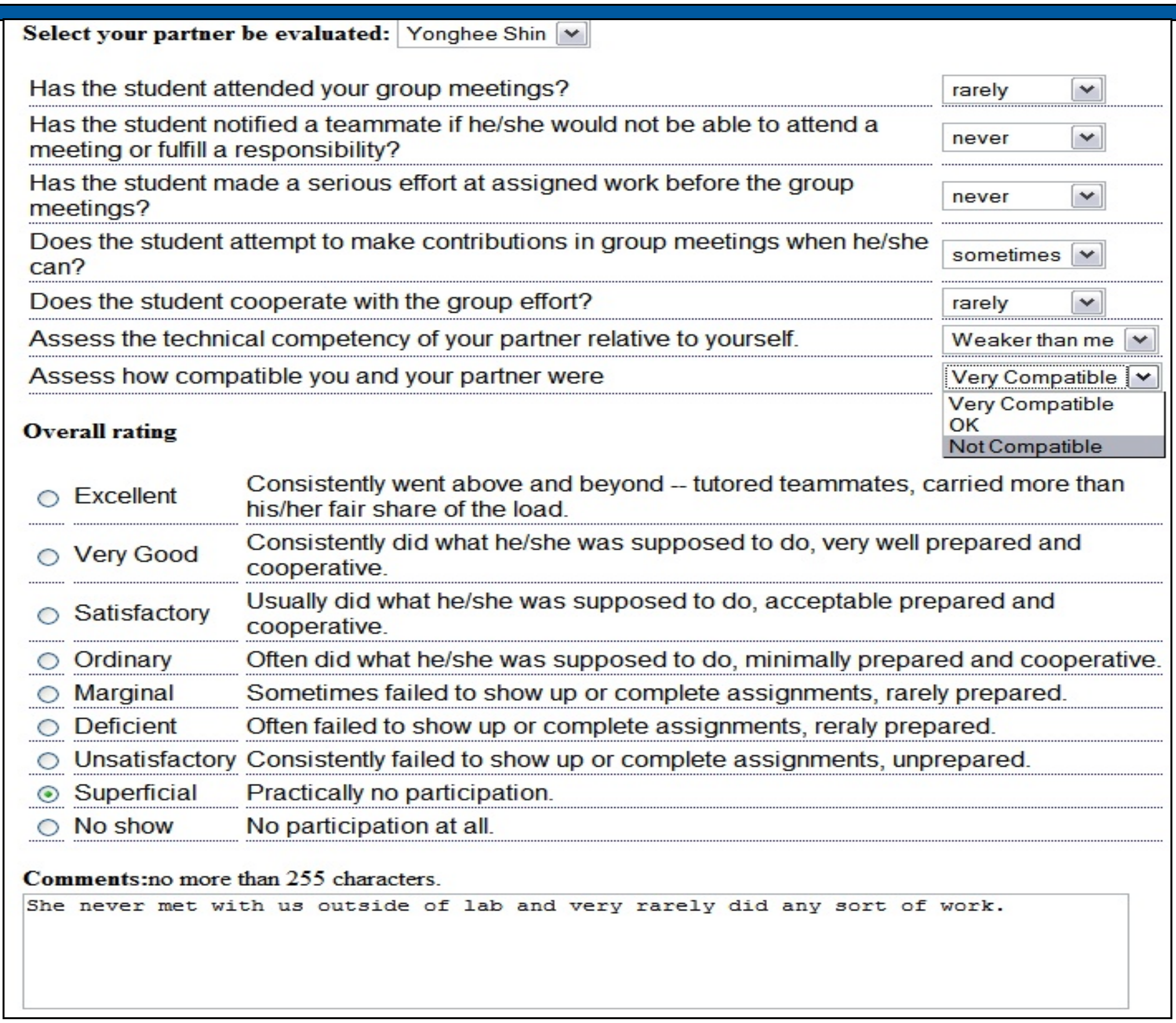
You will only need to fill out this survey once. Once you finish, you may view your answers but not change them. **Check your answers twice before you submit them!**

Please take this online Meyers-Briggs test. The title of the online test says Jung Typology Test (the Myers-Briggs test is based on the Jung test). After the test, enter the results here.

Type	Strength of the preferences
Introversion	12
Sensing	1
Thinking	50
Perceiving	98

Don't forget to fill out your Learn Styles and Self Evaluation!

PairEval



Select your partner be evaluated: Yonghee Shin

Has the student attended your group meetings?	rarely
Has the student notified a teammate if he/she would not be able to attend a meeting or fulfill a responsibility?	never
Has the student made a serious effort at assigned work before the group meetings?	never
Does the student attempt to make contributions in group meetings when he/she can?	sometimes
Does the student cooperate with the group effort?	rarely
Assess the technical competency of your partner relative to yourself.	Weaker than me
Assess how compatible you and your partner were	Very Compatible

Overall rating

○	Excellent	Consistently went above and beyond -- tutored teammates, carried more than his/her fair share of the load.
○	Very Good	Consistently did what he/she was supposed to do, very well prepared and cooperative.
○	Satisfactory	Usually did what he/she was supposed to do, acceptable prepared and cooperative.
○	Ordinary	Often did what he/she was supposed to do, minimally prepared and cooperative.
○	Marginal	Sometimes failed to show up or complete assignments, rarely prepared.
○	Deficient	Often failed to show up or complete assignments, reraly prepared.
○	Unsatisfactory	Consistently failed to show up or complete assignments, unprepared.
●	Superficial	Practically no participation.
○	No show	No participation at all.

Comments:no more than 255 characters.

```
She never met with us outside of lab and very rarely did any sort of work.
```

Pair Evaluation

- With or without a tool, it boils down to a few questions:
  - Did the pair get along?
  - Did you get the work done?
  - Do you feel like you “did your fair share?”
- More data is nice/interesting, but this is all you really need
- Reliable feedback system is needed (both for you and the students)

Pair Evaluation

- NCWIT resources have surveys you can use!
- Example in your packet
- http://www.ncwit.org/pairprogramming

Pair Evaluation and Assessment

- If there's no problem… then great!
- If there is…
  - If possible, ask the students one at a time: "If 100% effort is you doing exactly what you should have been doing, what percentage did you actually do?"
  - 95% of the time, this works!
  - For the other 5%, you have to use your best judgement

Assessment

- For other class assessments, I do not adjust anything
- All tests/exams, pop quizzes, etc. all stay the same as if it were a solo programming only course

The Biggest Cost

- Training!
- Instructors, TAs, **and students** need to be taught how to do effective pair programming in a controlled environment!
- The controlled environment could be a closed lab or lecture-lab system

But we don’t have a closed lab?

- CS1:
  - Assigned pairs not advisable if they don’t know the partners Try to introduce in guided labs / in-class activities first
- CS2:
  - Proceed with caution for assigned pairs for first assignment
  - Works better after first month or so
  - At least bond in lab + some outside work
- CS2+:
  - After at least one paired class
  - Bonding still beneficial, outside work fine

Getting Involved

- Instructors and Teaching Assistants have to take an active role in lab
  - Must monitor and approach pairs if they seem to be dysfunctional
  - Should "strongly encourage" drivers and navigators to switch
- Instructors also must understand that some pairings are just not going to work
  - Don’t let it discourage you!

How Many Pairings Fail?

Class	**Very compatible**	**OK**	**Not compatible**
CS1	64%	32%	4%
SE-P1	60%	33%	7%
SE-P2	56%	35%	9%
OO	76%	15%	9%
Total	**60%**	**33%**	**7%**

Problem Pairs

- Will problem pairs happen? Yes.
- Particular cases:
  - The “I don’t care” student
  - The special needs student
  - The absent student
  - The “liberal arts vs. engineering” student
- These problems are not pair programming related, but pair programming can make these come to the surface more often

Guidelines To Follow

- Strict tardiness / absence policy must be followed for pair activities to guard against lazy partners.
  - Loss of partner, points, and bad evaluation
- There **must** be a reporting mechanism for students to provide feedback on partners
  - CATME or a simple Google Form
  - "If you could rate your effort based on 100%.."

Guidelines To Follow

- Assignments should be a bit more challenging
  - "Softball" assignments tend to be finished by a single person without consulting their partner

- The environment for pairing must be conducive to pairing

Guidelines To Follow

- Don't go overboard!
  - Everything in moderation ☺
  - Pairing isn't for every assignment
  - There must be a balance (in work and in grade)

Myths and Legends

- Myth: Half the students will learn
  - *"In the first course, students need some time to absorb the ideas themselves."*
  - *"My inclination is to allow more group work starting in the second course."*
  - *"We want to be sure that each student writes enough code him/herself to learn the introductory concepts."*
  - *"I am against pair-programming in introductory courses, where students need to develop strong programming skills themselves."*

Myths and Legends

- In fact, all the students learn pretty well…
  - Studies at NCSU and SDSU showed that exam scores were comparable or improved for all students in introductory classes
  - Also, the percentage of students whose grade in CS2 went down by over 1/3 of a grade dropped once pairing was used in CS1

Williams, L., Layman, L.,
Lab Partners: If They're Good Enough for the Sciences, Why Aren't They Good Enough for Us?,
Conference on Software Engineering Education and Training (CSEE&T '07)

Myths and Legends

- By falling for this myth, you're perpetuating another one
  - "All computer scientist work by themselves in cubicles struggling to code."
- We all know that creating software is HIGHLY collaborative!
- Why give the wrong impression in the first class they take!?

Myths and Legends

- Myth: Cheating will increase
  - *"With loose rules about who partners are, people will just pass code around. There has to be structure!"*
  - *"Old partners may feel obliged to help their former teammates."*

Myths and Legends

- Think about it a little differently…
- When we provide partners, students now have a support system they can turn to
  - Anecdotal evidence from students indicated that the stress of feeling alone and isolated made them consider cheating
- Two people now have to agree on cheating!
  - Well… there are exceptions to this one…
  - Moss and etector are valuable tools

Other Guidelines and Myths

- Any others to add?

Resources

- http://www.realsearchgroup.org/pairlearning
- http://www.ncwit.org/pairprogramming

- My personal website: http://www.cs.virginia.edu/~sherriff

- My email: sherriff@virginia.edu