Code Hunt for University Courses

Tao Xie
University of Illinois at Urbana-Champaign
In collaboration with the Code Hunt team at Microsoft Research and Illinois ASE students

Related Papers/Resources: https://sites.google.com/site/asergrp/projects/ese
Educational Software Engineering
aka. Software Engineering for Education
develops software engineering (+gaming) technologies for general educational tasks, going beyond educational tasks for software engineering (e.g., software engineering education).

Yearly Code Hunt Workshop with Upcoming SE/Edu Conferences?

*International Code Hunt Workshop on Educational Software Engineering* as **Open** Forum for Exchanging Ideas and Early Results of the Community

Your Contributions and Suggestions to Build and Grow the Community as Follow-up to this 1st Wonderful Code Hunt Workshop!

Inspired from International SPIN Workshop on Model Checking of Software (SPIN)
Observations & Motivations of Coding Duels at Code Hunt

• Gaming/fun aspect (e.g., “reverse-engineering”)
  – Guessing/discovering/exploring, ...

Example User Feedback on Coding Duels

“I used to love the first person shooters and the satisfaction of blowing away a whole team of Noobies playing Rainbow Six, but this is far more fun.”

“I’m afraid I’ll have to constrain myself to spend just an hour or so a day on this really exciting stuff, as I’m really stuffed with work.”

“It really got me *excited*. The part that got me most is about spreading interest in teaching CS: I do think that it’s REALLY great for teaching | learning!”
Observations & Motivations of Coding Duels at Code Hunt

• Gaming/fun aspect (e.g., “reverse-engineering”)
  – Guessing/discovering/exploring, ...

• Engaging/sustaining aspect
  – Not loss of courage/confidence, sense of progression
    • Earlier presented hint generation, progress indication with model counting
Coding Duel Competition
@ICSE 2011

Observations & Motivations of Coding Duels at Code Hunt

• Gaming/fun aspect (e.g., “reverse-engineering”)
  – Guessing/discovering/exploring, ...

• Engaging/sustaining aspect
  – No loss of courage/confidence, sense of progression
    • Earlier presented hint generation, progress indication with model counting

• Teaching/learning aspect
  – Designing coding duels for pedagogical purposes
Observations & Motivations of Coding Duels at Code Hunt

• Gaming/fun aspect (e.g., “reverse-engineering”)
  – Guessing/discovering/exploring, ...

• Engaging/sustaining aspect
  – No loss of courage/confidence, sense of progression
    • Earlier presented hint generation, progress indication with model counting

• Teaching/learning aspect
  – Designing coding duels for pedagogical purposes

• Classroom integration aspect
  – Office Mix + Code Hunt
Outline

• Progress Indication (simple techniques and some initial observations)

• Coding duels for teaching and learning

• Office Mix + Code Hunt for teaching and learning
Motivation: Why Syntactic Edit Distance Alone Not Enough

Student Solution Examples from Pex4Fun

// Fancy years are years that have all the same numbers
// Fancy years are never less than 1000 and never bigger than 999
public static string isFancyYear(int i) {
    if (i < 1000 || i > 9999)
        return "not a fancy year";
    int digit = i % 10;
    while (i != 0) {
        if (i % 10 != digit)
            return "not a fancy year";
        i /= 10;
    }
    return "fancy year";
}

public static string isFancyYear(int i) {
    if (i == 1111) return "fancy year";
    if (i == 2222) return "fancy year";
    if (i == 3333) return "fancy year";
    if (i == 4444) return "fancy year";
    if (i == 5555) return "fancy year";
    if (i == 6666) return "fancy year";
    if (i == 7777) return "fancy year";
    if (i == 8888) return "fancy year";
    if (i == 9999) return "fancy year";
    return "not a fancy year";
}
Example of Behavioral Subsumption

```java
public int Reference(int x) {
    if (x % 7 == 0) return 8;  // Conditional 1
    return x % 7;  // Conditional 2
}
```

Secret Program

```java
public int Program1(int x) {
    if (x % 7 == 0) return 8;
    if (x == 1009) return 1;
    return x;
}
```

Student Program 1

```java
1 public int Program2(int x) {
2     if (x % 7 == 0) return 8;
3     if (x == 1009) return 1;
4     if (x == 383) return 5;
5     return x;
6 }
```

Student Program 2

Student Solution Examples from Pex4Fun

*Subsumed by*
Metrics for Behavioral Similarity

Key Idea: \(#\text{Passing Paths} / (\#\text{Passing Paths} + \#\text{Failing Paths})\)
Subsumption, comparison of input domain contents

• Random Sampling (RS)

• Single-program Symbolic Execution (SSE)

• Paired-program Symbolic Execution (PSE)

```
1 public void PairedProgram (object[] args) {
2     Debug.Assert(Program1(args) == Program2(args));
3 }
```

Advantages: simple techniques that piggy-bag Pex’s existing effort/cost for supporting Code Hunt
Example: PSE works but RS and SSE do not work

```
int Reference(int x) {
    return 42 - x;
}
```

(a) Reference Program

```
int Submission1(int x) {
    return 42;
}
```

(b) Submission 1

```
int Submission2(int x) {
    if (x == 0) return 42;
    if (x == 42) return 0;
    return 0;
}
```

(c) Submission 2

Student Solution Examples from Pex4Fun
Example: SSE and PSE produce desirable metric values for behavioral similarity

```java
public static int Puzzle(byte number) {
    int ct = 0;
    while (number != 0) {
        if (number % 2 > 0) ct++;
        number >>= 1;
    }
    return ct;
}
```

(a) Reference Program

```java
public static int Puzzle(byte number) {
    var sum = 0;
    for (int i = 0; i < 8; i++)
    {
        sum += (number >> i & 1);
    }
    return sum;
}
```

(b) Submission 1

(c) Submission 2

Student Solution Examples from Pex4Fun
Outline

• Progress Indication (simple techniques and some initial observations)

• Coding duels for teaching and learning

• Office Mix + Code Hunt for teaching and learning
Teaching and Learning

The Social Classroom

Course Description: Learn how PexForFun turns teaching computer science into a social gaming experience.

Teacher: the Pex Team

Associated Pages:
- The Social Classroom Whitepaper
- The Social Classroom Sample Page

Registered Students:
- mbarnett
- Murray
- Kai
- (no nickname)
- Chris C Sharp
- (no nickname)
- TaoXie
- (no nickname)
- meisl
- (no nickname)
- Nima
- Rocky
- Micgi
- Ishtiaque

The following link allows any signed in user to register for and access this course.
http://pexforfun.com/thesocialclassroom

Pex for fun - from Microsoft Research - Page - Windows Internet Explorer

Status of The Social Classroom

Your Progress: Nikolai Tillmann, you already won 1 Coding Duel; 1 more to go!

Factorial 1 attempt
ArraySort won after 15 attempts

All Students' Progress:

<table>
<thead>
<tr>
<th>Coding Duels</th>
<th>#0</th>
<th>#1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chris C Sharp</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>meisl</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Nima</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Micgi</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Microsoft Research RISE
Skills Being Trained

Induction
Problem solving/debugging
Program understanding/programming
Testing
Specification writing
Observed Benefits
• Automatic Grading
• Real-time Feedback (for Both Students and Teachers)
• Fun Learning Experiences

http://pexforfun.com/gradsofteng
public class Program {
  // param:obj_type:non-empty Object type to add a new property to.
  // param:prop_name:Name of the new property to add. This name needs to be a valid identifier, which is no longer than 32 characters, starting with a letter (a-z) and consisting of only small letters (a-z), numbers (0-9) and/or underscores.
  // param:prop_type:Type of the new property: 1 for integer, 2 for string (max. 255 characters), 3 for text blob (max. 64kb)
  // Can you write preconditions in Code Contracts for the above natural-language requirements?

  public static int Puzzle(string obj_type, string prop_name, int prop_type)
  {
    return prop_type;
  }
}
Ex Coding Duel for Training Requirements

```csharp
public class Program {
    // param:obj_type:non-empty Object type to add a new property to.
    // param:prop_name:Name of the new property to add. This name needs to be a valid identifier, which is no longer than 32 characters, starting with a letter (a-z) and consisting of only small letters (a-z), numbers (0-9) and/or underscores.
    // param:prop_type:Type of the new property: 1 for integer, 2 for string (max. 255 characters), 3 for text blob (max. 64kb)
    public static int Puzzle(string obj_type, string prop_name, int prop_type)
    {
        Contract.Requires(obj_type != null);
        Contract.Requires(obj_type.Length > 0);
        Contract.Requires(prop_type > 0);
        Contract.Requires(prop_type <= 3);
        Contract.Requires(prop_name != null);
        Contract.Requires(prop_name.Length > 0);
        Contract.Requires(prop_name.Length <= 32);
        Contract.Requires(_regex.IsMatch(prop_name.Substring(0,1),"[a-z]")
        Contract.Requires(_regex.IsMatch(prop_name,@"^[a-z0-9]*$"));
        return prop_type;
    }
```
public static string Puzzle(int[] elems, int capacity, int elem) {
    if ((maxsize <= 0) || (elems == null) || (elems.Length > (capacity + 1)))
        return "Assumption Violation!";

    Stack s = new Stack(capacity);

    // Set up a stack with some elements
    for (int i = 0; i < elems.Length; i++)
        s.Push(elems[i]);

    // Cache values used in assertions
    int origSize = s.GetNumOfElements();
    s.Push(elems[i]);

    // Please fill in below test scenario on the s stack

    // The lines below include assertions to assert the program behavior
    PexAssert.IsTrue(s.GetNumOfElements() == origSize + 1);
    PexAssert.IsTrue(s.Top() == elem); PexAssert.IsTrue(!s.IsEmpty());
    PexAssert.IsTrue(s.IsMember(elem));

    return s.GetNumOfElements().ToString() + "; " + s.Top().ToString() + "; " + s.IsMember(elem).ToString() + "; " + s.IsEmpty();
}
public static string Puzzle(int[] elems, int capacity, int elem) {
    if ((maxsize <= 0) || (elems == null) || (elems.Length > (capacity + 1)))
        return "Assumption Violation!";
    Stack s = new Stack(capacity);
    for (int i = 0; i < elems.Length; i++)
        s.Push(elems[i]);
    int origSize = s.GetNumOfElements();
    //Please fill in below test scenario on the s stack
    if (s.IsMember(elem)) return "Assumption Violation!";
    if (s.GetNumberOfElements() >= s.MaxSize()) return "Assumption Violation!";
    s.Push(elem);
    //The lines below include assertions to assert the program behavior
    PexAssert.IsTrue(s.GetNumOfElements() == origSize + 1);
    PexAssert.IsTrue(s.Top() == elem); PexAssert.IsTrue(!s.IsEmpty());
    PexAssert.IsTrue(s.IsMember(elem));
    return s.GetNumOfElements().ToString() + "; " + s.Top().ToString() + "; " + s.IsMember(elem).ToString() + "; " + s.IsEmpty(); }
Can you complete method bodies of some specified methods of the given classes to match the secret implementation?

```csharp
public class DesignPatternEx {
    //Please fill in the method bodies of CreatePages() declared on Document1 and Document2
    //to match the behavior of the secret implementation.
    //Hint: the classes below realize a design pattern

    public static string Puzzle(int[] wordNumsPerPage) {
        Document[] documents = new Document[2];
        documents[0] = new Document1();
        documents[1] = new Document2();
        if (wordNumsPerPage == null) return "Assumption Violation!";
        if (wordNumsPerPage.Length != (documents[0].Pages.Count +
                                         documents[1].Pages.Count)) return "Assumption Violation!";
        int totalFee = 0; int pageIndex = 0;
        foreach (Document document in documents) {
            foreach (Page page in document.Pages) {
                totalFee = totalFee + page.CalPageFee(wordNumsPerPage[pageIndex++]);
            }
        }
        return totalFee.ToString();
    }
}
```
Coding Duel 1 for Training Design Patterns

```csharp
class Document1 : Document {
    public override void CreatePages() {
        //fill in code here
    }
}

class Document2 : Document {
    public override void CreatePages() {
        //fill in code here
    }
}

//abstract classes
abstract class Page {
    abstract public int CalPageFee(int wordCount);
}

abstract class Document {
    private List<Page> _pages = new List<Page>();
    public Document() { this.CreatePages(); }
    public List<Page> Pages { get { return _pages; } }
    public abstract void CreatePages(); }

class PageType1 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 1; }
}

class PageType2 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 2; }
}

class PageType3 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 3; }
}

class PageType4 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 4; }
}

class PageType5 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 5; }
}
```
class Document1 : Document {
public override void CreatePages() {
    Pages.Add(new PageType1());
    Pages.Add(new PageType3());
    Pages.Add(new PageType4());
}
}

class Document2 : Document {
public override void CreatePages() {
    Pages.Add(new PageType2());
    Pages.Add(new PageType5());
}
}

class PageType1 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 1; }
}

class PageType2 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 2; }
}

class PageType3 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 3; }
}

class PageType4 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 4; }
}

class PageType5 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 5; }
}

//abstract classes
abstract class Page {
    abstract public int CalPageFee(int wordCount);
}

abstract class Document {
private List<Page> _pages = new List<Page>();
public Document() { this.CreatePages(); }
public List<Page> Pages { get { return _pages; } }
public abstract void CreatePages();
}
class Document1 : Document {
    public override void CreatePages() {
        Pages.Add(new PageType1());
        Pages.Add(new PageType3());
        Pages.Add(new PageType4());
    }
}

class Document2 : Document {
    public override void CreatePages() {
        Pages.Add(new PageType2());
        Pages.Add(new PageType5());
    }
}

class PageType1 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 1; }
}

class PageType2 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 2; }
}

class PageType3 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 3; }
}

class PageType4 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 4; }
}

class PageType5 : Page {
    public override int CalPageFee(int wordCount) { return wordCount * 5; }
}

//abstract classes
abstract class Page {
    abstract public int CalPageFee(int wordCount);
}

abstract class Document {
    private List<Page> _pages = new List<Page>();
    public Document() { this.CreatePages(); }
    public List<Page> Pages { get { return _pages; } }
    public abstract void CreatePages();
}
Coding Duel 3 for Training Design Patterns

```csharp
public class DesignPatternEx
{
    // Please fill in the method bodies of GetTotalFee() and CalPageFee(int) declared on CalPageFeeSystem to match the behavior of the secret implementation.
    // Hint: the classes below realize a design pattern
    public static string Puzzle(int[] wordNumsPerPage)
    {
        if (wordNumsPerPage == null)
            return "Assumption Violation!";
        if (wordNumsPerPage.Length != 5)
            return "Assumption Violation!";

        CalPageFeeSystem calPageFeeSystem = new CalPageFeeSystem();
        calPageFeeSystem.CalPageFee(wordNumsPerPage[0]);
        calPageFeeSystem.CalPageFee(wordNumsPerPage[1]);
        calPageFeeSystem.CalPageFee(wordNumsPerPage[2]);
        calPageFeeSystem.CalPageFee(wordNumsPerPage[3]);
        calPageFeeSystem.CalPageFee(wordNumsPerPage[4]);
        return calPageFeeSystem.GetTotalFee().ToString();
    }
}
```
class CalPageFeeSystem
{
    private CalPageFeeByType _calPageFeeByType;
    public CalPageFeeSystem()
    {
        _calPageFeeByType = new CalPageFeeByType1(0, this);
    }

    public CalPageFeeByType CalPageFeeByType
    { get { return _calPageFeeByType; } set { _calPageFeeByType = value; } }

    public int GetTotalFee()
    {
        //fill in code here by replacing the line below
    }

    public void CalPageFee(int wordCount)
    {
        //fill in code here
    }
}
Coding Duel 3 for Training Design Patterns

class CalPageFeeByType1 : CalPageFeeByType
{
    public CalPageFeeByType1(int totalFee, CalPageFeeSystem sys)
    {
        this.TotalFee = totalFee;  this.CalSystem = sys;  
    }

    public override void CalPageFee(int wordCount)
    {
        this.TotalFee =  this.TotalFee + wordCount * 1;
        this.CalSystem.CalPageFeeByType = new CalPageFeeByType2(this.TotalFee, this.CalSystem); 
    }
}

class CalPageFeeByType2 : CalPageFeeByType
{
    public CalPageFeeByType2(int totalFee, CalPageFeeSystem sys)
    {
        this.TotalFee = totalFee;  this.CalSystem = sys;  
    }

    public override void CalPageFee(int wordCount)
    {
        this.TotalFee =  this.TotalFee + wordCount * 2;
        this.CalSystem.CalPageFeeByType = new CalPageFeeByType3(this.TotalFee, this.CalSystem); 
    }
}
class CalPageFeeByType3 : CalPageFeeByType
{
    public CalPageFeeByType3(int totalFee, CalPageFeeSystem sys)
    {
        this.TotalFee = totalFee;
        this.CalSystem = sys;
    }

    public override void CalPageFee(int wordCount)
    {
        this.TotalFee = this.TotalFee + wordCount * 3;
        this.CalSystem.CalPageFeeByType = new CalPageFeeByType4(this.TotalFee, this.CalSystem);
    }
}

class CalPageFeeByType4 : CalPageFeeByType
{
    public CalPageFeeByType4(int totalFee, CalPageFeeSystem sys)
    {
        this.TotalFee = totalFee;
        this.CalSystem = sys;
    }

    public override void CalPageFee(int wordCount)
    {
        this.TotalFee = this.TotalFee + wordCount * 4;
        this.CalSystem.CalPageFeeByType = new CalPageFeeByType5(this.TotalFee, this.CalSystem);
    }
}
class CalPageFeeSystem
{
    private CalPageFeeByType _calPageFeeByType;
    public CalPageFeeSystem()
    {
        _calPageFeeByType = new CalPageFeeByType1(0, this);
    }

    public CalPageFeeByType CalPageFeeByType
    { get { return _calPageFeeByType; } set { _calPageFeeByType = value; } }

    public int GetTotalFee()
    {
        //fill in code here by replacing the line below
        return _calPageFeeByType.GetTotalFee();
    }

    public void CalPageFee(int wordCount)
    {
        //_calPageFeeByType.CalPageFee(wordCount);
    }
}
Reminder: Coding Duels ARE Powerful/Flexible

http://parsonspuzzles.appspot.com/

Amenable to Touch-based Devices for Moving Objects/Lines ... Hololens?
Outline

• Progress Indication (simple techniques and some initial observations)

• Coding duels for teaching and learning

• Office Mix + Code Hunt for teaching and learning
Office Mix + Code Hunt

https://mix.office.com/

https://mix.office.com/watch/4ibulo2jy5tp

https://www.codehunt.com

Teaching and Learning Programming and Software Engineering via Interactive Gaming

Tao Xie
University of Illinois at Urbana-Champaign
Summary: Testing Tool ➔ Educational Gaming

DSE/Pex

Support

Code Hunt/Pex for Fun:
Interactive Gaming for Teaching and Learning

https://www.codehunt.com
http://pex4fun.com
Testing Tool ➔ Educational Gaming


Support

using System;
public class Program {
    public static int Puzzle(int x) {
        return x * x;
    }
}

Q & A

Thank you!

Contact: taoxie@illinois.edu
http://www.cs.illinois.edu/homes/taoxie/
Related Papers/Resources: https://sites.google.com/site/asergrp/projects/ese
Summary: Testing Tool ➔ Educational Gaming

DSE/Pex

Support

Code Hunt/Pex for Fun:
Interactive Gaming for Teaching and Learning

https://www.codehunt.com
http://pex4fun.com