Verification Games
Making Verification Fun

Werner Dietl

Stephanie Dietzel, Michael D. Ernst,
Nathaniel Mote, Brian Walker,
Seth Cooper, Timothy Pavlik, Zoran Popović

http://cs.washington.edu/verigames

University of Washington
Computer Science & Engineering
Angry Birds
Software verification
Which is more fun?

- Play games
- Prove your programs correct

Angry Birds:
- 02 Nov 2011: 200000 years play-time
- 11 May 2012: downloaded one billion times
Crowd-sourced verification

1. Make software verification easy and fun
2. Make the game accessible to everyone
3. Harness the power of the crowd

Goal: Verify software while you wait for the bus
Highly-skilled, expensive labor

Verified software (with proof/annotations)

Automatic translation

Encodes a constraint system

Volunteers

Completed game

YOU WIN!

Automatic translation

Code

Game
Code

Automatic translation

Game

Bug detected, notify programmer

Completed game with buzzsaws

Volunteers

Automatic translation

YOU WIN!
Example: null pointer errors

Goal: no dereference of null

Pipe ↔ a variable
Pipe width ↔ narrow: non-null
    wide: maybe null

Ball ↔ a value
Ball size ↔ small: non-null
    large: maybe null

Pinch point ↔ dereference

Unmodifiable pipe/ball ↔ literal null, object creation
Program $\leftrightarrow$ game correspondence

Pipe $\leftrightarrow$ a variable
Pipe width $\leftrightarrow$ type of the variable
Ball $\leftrightarrow$ a value
Ball size $\leftrightarrow$ a property of the value
Pinch point $\leftrightarrow$ requirement
Unmodifiable pipe/ball $\leftrightarrow$ requirement

Intuition: dataflow
Other examples

SQL injection
unintended side effects
format string and regexp validation
incorrect equality checks
race conditions and deadlocks
units of measurement
aliasing

...
Challenges

Will the game be fun?
Better than waiting for the bus

Do people outperform verification algorithms?
Inference is undecidable

Hypothesis:
no for correct, verifiable programs
yes for incorrect or unverifiable programs

Game players only have to reduce overall verification cost, not fully verify the program
Scoring & Collaboration

1. Game score influenced by
   - Collisions (verifiability)
   - Use of buzzsaws (trusted assumptions)
   - Pipe widths, distinguishing input and output pipes (re-usability of modules)

2. Collaboration & competition between players
   - High-score boards
   - Collaborative teams solve challenges
   - Social aspects (chats, forums, ...)
Scalability & Optimization

1. Brute force not feasible for large programs

2. Scale-up verification by
   • Crowdsourcing games
     1. Distribute games to humans
     2. Reconfigure games to adjust difficulty
     3. Redundancy
   • Automatic inference and optimizations
     1. How many easy challenges should be left for humans to feel good about progress?
FoldIt

1. Proteomics game at UW
2. Effectively created the genre of games that solve hard problems
3. Three Nature papers in under 2 years
4. Over 240,000 players, 200+ new per day
FoldIt
Contributions

Gamification of program verification

Game...

- encodes correctness conditions
- utilizes human intuition & insight
- is playable by anyone

Goal: cheaper verification $\Rightarrow$ more verification

http://cs.washington.edu/verigames
Verification Games
Making Verification Fun

Werner Dietl

Stephanie Dietzel, Michael D. Ernst,
Nathaniel Mote, Brian Walker,
Seth Cooper, Timothy Pavlik, Zoran Popović

http://cs.washington.edu/verigames

University of Washington
Computer Science & Engineering
Checker Framework Tutorial

Do you want to learn how to build your own pluggable type systems?

Come see my PLDI tutorial!

Saturday, 16 June from 9:00 to 12:00

Conference 9