Throw Down an AI Challenge

Todd W. Neller, Gettysburg College
Ingrid Russell, University of Hartford
Zdravko Markov, Central Connecticut State University
Beginnings

- Recall your introduction to CS.
- Which experiences attracted your interest?
- Which dialog is more likely?
  - “I was really excited to be able to print different patterns of asterisks on the screen. And that Fibonacci sequence assignment… Wow!”
  - “I remember programming a text adventure. My code was a mess in retrospect, but I really enjoyed the accomplishment.”
Lessons from Game AI and Fitness Coaching

- The goal of AI and Game AI differs significantly:
  - AI researchers: **play optimally** so as to win maximally, or **play humanly** so as to build better cognitive models.
  - Game AI developers: **play engagingly** so as to push the human opponents to their best play… and yet lose to them.
- Weight training: Lift so as to stress and build muscle, but do not lift to failure.
Throw Down a Challenge

- Dale Carnegie: “throw down a challenge”
- Students more likely to pursue CS if early challenges are relevant, exciting, and/or interesting.
- Such challenges should be
  - difficult enough to engage
  - not so difficult to lead most students to failure/loss.
- How can AI play a part in engaging students early?
A New Perspective on AI

- AI - the really interesting miscellaneous pile of Computer Science
  - Control beyond classical Control Engineering → Robotics
  - Knowledge beyond Database expressivity → Knowledge Representation & Reasoning
- If it takes intelligence and doesn’t fit elsewhere...
- AI can give intro students a vision of the expansive possibilities of CS!
Machine Learning Experiences in AI (MLExAI)

- NSF grant DUE CCLI-A&I Award Number 0409497

Goals:
- Enhance the student learning experience in the AI course.
- Increase student interest and motivation to learn AI.
- Introduce students to an increasingly important research area, thus motivating them to pursue further study.
- Increase student interest and motivation to build AI applications by allowing them to develop learning systems where they can implement the various concepts covered in the AI course.

Developed six adaptable, hands-on laboratory projects that can be closely integrated into a one-term AI course

Phase 2 underway with 20 faculty members contributing
Project: The Game of Clue

- Popular boardgame Clue (a.k.a. Cluedo) serves as a fun focus problem
  - Learn fundamentals of propositional logic and KR&R terminology
  - Solve basic logic problems with and without the aid of a satisfiability solver
  - Implement expert reasoner for deducing Clue case file contents.
  - Several possible advanced projects as well
Minimalist Introduction to Logic

- Build on what they know: Boolean variables → atomic sentences
- Propositional logic allows simple yet rich discussion of logic, e.g.:
  - atomic sentences, operators, literals, truth assignments, (un)satisfiability, models, validity, tautologies, entailment, and logical equivalence.
- Project materials feature brief introduction
Example Problem

- Amy says, “Bob is a liar.” Bob says, “Cal is a liar.” Cal says, “Amy and Bob are liars.” Who is telling the truth?
- \( \{A \iff \neg B, B \iff \neg C, C \iff (\neg A \land \neg B)\} \)
- Conjunctive Normal Form: \( \{\neg A, \neg B\}, \{B, A\}, \{\neg B, \neg C\}, \{C, B\}, \{\neg C, \neg A\}, \{\neg C, \neg B\}, \{A, B, C\} \)
- Project features set of propositional logic word problems
- Conversion to CNF by instructor?
Satisfiability Reasoning

● Student project feedback:
  ● Reasoning “black box” unsatisfactory
  ● Student desire to implement reasoner
● How many lines of code would it take to implement a simple…
  ● … stochastic local satisfiability search?
  ● … resolution theorem prover?
● Answer: ~100 lines of Java code each
Input Format

- CNF: \{\{\neg A, \neg B\}, \{B, A\}, \{\neg B, \neg C\}, \{C, B\}, \{\neg C, \neg A\}, \{\neg C, \neg B\}, \{A, B, C\}\}
- Corresponding input file format:
  
  
  
  -1  -2
  1  2
  -2  -3
  2  3
  -3  -1
  -3  -2
  3  1  2
Simple WalkSAT Variant

- Generate a random model
- Find all unsatisfied clauses for the model
- iterations = 0
- While there are unsatisfied clauses and iterations < max:
  - Pick a random variable of a random unsatisfied clause and negate it in the model
  - Find all unsatisfied clauses for the new model
  - If more clauses are unsatisfied then revert to the previous model with high probability p
  - iterations = iterations + 1
- Return whether or not clauses are still unsatisfied
Simple Set of Support

- While no contradiction derived
  - Pick the next clause $c_2$ in the support set
  - For each clause $c_1$ before $c_2$:
    - Attempt to resolve $c_1$ and $c_2$, adding result to support set if successful, checking for contradiction

- Resolution:
  - Seek complementary literal in each clause pair
  - For resolved clause, omit
    - Tautologies – (contains complementary literals)
    - Subsumed clauses
Project: Solving the Dice Game Pig

- Using the computation of optimal play for the jeopardy dice game Pig as a central challenge problem, we introduce:
  - dynamic programming, with worked examples, and relevant exercises
  - value iteration, with worked example and exercises
  - several possible advanced projects
Best Pedagogical Games

- The best games for teaching
  - Have interesting play, non-trivial decisions
  - Simple rules with few exceptions: easy to learn, requiring little code

- Examples:
  - Solitaire deterministic: Peg Solitaire, Lunar Lockout, SameGame, Minesweeper
  - Two player deterministic: Nim, Checkers, Hex
  - Two player bluff: Liar’s Dice, Simplified Poker
  - Jeopardy: Blackjack, Pig
Pig Rules

- The first player reaching 100 points wins.
- On each turn, a player rolls a die as many times as desired until either the player holds and scores the sum of the rolls, or rolls a 1 and scores nothing.
“Hold at 20” Turn Comments

// Initialize variables

while () { // Turn not over
    // Roll die
    if () { // 1 rolled
        // Reset turn total and end turn
    }
    else {
        // Add roll to turn total
    }
}
"Hold at 20" Turn Code

```java
Random random = new Random();
int turnTotal = 0;
boolean pigRolled = false;
while (!pigRolled && turnTotal < 20) {
    int roll = random.nextInt(6) + 1;
    System.out.println("Roll: " + roll);
    pigRolled = (roll == 1);
    if (pigRolled)
        turnTotal = 0;
    else
        turnTotal += roll;
}
System.out.println("Score gain: " + turnTotal);
```
Bottom-Up Design

- Simulate a single hold-at-20 turn.
- Given the current score, additionally hold at goal score
- Simulate solitaire game with hold-at-20-or-goal
- Extend to two-player game simulation
- Replace one hold-at-20-or-goal player with a human player.
Object Oriented Design

- Objects:
  - Game
  - Die
  - Player
  - Human player
  - Computer player
Competition

- Such assignments lend themselves to interesting follow-on possibilities:
  - Policy design: Beat hold-at-20-or-goal
  - Design evaluation: Monte Carlo simulation
  - Fun competition: Interface implementation
- Hook: An optimal policy has been computed. Want to know how it’s done? → Pig project
More MLExAI Projects

- More MLExAI projects exist:
  - Web User Profiling and Web Document Classification
  - Character Recognition Using Neural Networks
  - Solving the N-Puzzle Problem
  - And more in phase 2!
- More introductory challenges in paper
Conclusion

- Key engaging learning experiences are:
  - Challenging (but not overly challenging)
  - Relevant, exciting, interesting
- AI opens doors to possibilities beyond databases, networking, etc.
- AI’s challenge: intro-level assignments
  - Engage with difficult play but do not defeat
  - Stress but do not tear mental muscle