The Intelligent Game Designer:
Game Design as a New Domain for Automated Discovery

Adam M. Smith

UC Santa Cruz
amsmith@cs.ucsc.edu

29 July 2009
PREFACE
Preface

- Game design is clearly a **creative activity**.

- I claim a **machine** can do it.
Preface

Bruce Buchanan (in AAAI-2000 Presidential Address) says of existing creative systems...

- they do not accumulate experience, and thus, cannot reason about it;

- they work within fixed frameworks including fixed assumptions, and criteria of success;

- they lack the means to transfer concepts and methods from one domain to another.
Preface

- My “intelligent game designer” is all about turning experience into communicable knowledge (producing games along the way).

- But how?
  - **Operationalize** game design as an automatable scientific process.
  - **Re-conceptualize** creative design of expressive artifacts knowledge-seeking effort.
Why is this realistic?

- Why me?
  - Game development
  - Generative art

- Why now?
  - Fresh tools
    - Abductive/Inductive logic learning
    - Automated debugging for logic programs
  - Fresh formalisms for games
    - Event-calculus
    - Recombinable mechanics
INTRODUCTION

Context
Research Questions
Proposal Outline
Perspectives in Game Design

- Experience sharing (informal knowledge, words)
  - Textbooks, forum posts, and technical talks
- Code Sharing (formal knowledge, code)
  - Procedural content generation, drama management, game engines, and miscellaneous middleware
- Nearly-automated Systems
  - Peer or design buddy?
Perspectives in Learning / Creativity

- Statistical ML / Computational Intelligence
  - Structured data in, predictive model out

- Discovery systems
  - Data must be drawn out by experiment
  - Predictions should be consistent with rich, domain-specific, background knowledge

- Creative art systems
  - Artifacts are like exquisite experiments, results ignored
  - Leverage highly nuanced audience model, often fixed

- Domain-aware, creative discovery systems
  - Learning is the focus, artifact creation as side-effect
Research Questions

- **Function:**
  - How does an **intelligent game designer** function?

- **Implication:**
  - What does such a system imply for the relationship between **discovery** and **expressive domains**?
Function: “games”

- Recognizable as “video games”

- Focusing assumptions:
  - Single-player
  - Real-time
  - Mechanics-heavy
  - Abstracted representation
  - Minimal setting
Example “game”: Dyson

“Remotely command semi-autonomous self-replicating mining machines to take over an entire asteroid belt.”

- Single-player
- Real-time
- Mechanics-heavy
- Abstracted representation
- Minimal setting

http://www.dyson-game.com
Function: “intelligent”

- Learning from experience
  - Knowledge production as a function of past design and discovery actions
  - Documentation as proof
Function: “game design”

- Game design: the informed construction of rules systems and supporting logic required to produce playable games

- OK if playable games are a little rough, some human polish might be needed
Implications: Game Design

What does ______ mean in game design?

- Discovery?
- Conjecture?
- Experiment (environments, observations, instruments)?
- Verification?
- Proof?
Implications: Discovery

- What does _____ mean in discovery?
  - Prototyping and play testing?
  - Publishing a game?
  - Games vs. abstract state progression systems?
  - Expressive goals?
  - Fun?
Research Questions Revisited

- **Function:**
  - How does an *intelligent game designer* function?
    - Need to build a system!

- **Implication:**
  - What does such a system *imply* for the relationship between *discovery* and *expressive domains*?
    - Need some theories to generalize!
Outline

- Related work
  - Game design
  - Discovery and creativity systems

- Prior work
  - Interactive generative art
  - Logical games
  - Elementary discoveries in game design

- Proposed work
  - Theories
  - Systems
  - Experimental validation
  - Time line
Textbook game design
A call for structure
Game studies
Artificial intelligence

Models of discovery
Discovery systems
Computational creativity
Generative art

RELATED WORK
Textbook Game Design

Artifacts
- Design documents
- Prototypes
  - Paper
  - Computer-assisted
  - Computational
- Complete games

Processes
- Concept development
- Design
- Prototyping
- Play testing
  - Self-testing
  - Testing with friends
  - Testing with target audience
- Tuning
- Marketing
“Not enough is done to build on past discoveries, share concepts behind successes, and apply lessons learned from one domain or genre to another.” – Doug Church

- **Formal Abstract Design Tools** (Church 1999)
- **400 Project** (Barwood 2001)
- **The Case for Game Design Patterns** (Kreimeier 2002)
Game Studies

- **Swap Adjacent Games to Make Sets of Three** (Juul 2007)
- **Patterns in Game Design** (Bjork 2005)
- **Game Ontology Project** (Zagal 2005)
AI: Game Generation

- **EGGG** (Orwant 2000)
- **Automated Puzzle Generation** (Colton 2002)
- **Towards Automated Game Design** (Nelson 2007)
- **An Experiment in Game Design** (Togelius 2008)
- **Rhythm-Based Level Generation for 2D Platformers** (Smith 2009)
AI: General Game Playing

- GGP: getting machines to play arbitrary games well given only the rules and a little bit of time to practice (evolved from AI chess)

- Game Description Language (Love 2006) describes games as state transition systems in datalog.
AI: Game Design Assistance

- Parallel research by Mark J. Nelson at EIS
- Goal: create a game-design assistant that helps designers prototype their rule systems

- Gist:
  - Let the machine comment on formal issues
    - Reachability, exploits, indirect constraints
  - Let human players comment on soft issues
    - Engagement, fun, hesitation
Personal Game Design Experience

Drive-by CTF
AjaxWar
Katamari Damacy Text Adventure
the.cubing.game
the.discrete.gardender
Sequence Sleuth
Troy
fusepuck
T++

others I’ve forgotten…
Models of Discovery

- Two common domains:
  - Natural science
    - physics, chemistry, genomics, virology
  - Mathematics
    - graph theory, number theory

- Two common goals:
  - Explain historic discoveries
  - Produce new knowledge

- Unifying vocabulary for discovery:
  (Shrager and Langley 1990)
  - Knowledge structures
  - Processes
Discover Systems

Early Systems

- **DENDRAL** (Feigenbaum 1965)
- **BACON** (Langley 1977)
- **AM** (Lenat 1977)
- **EURISKO** (Lenat 1985)
- **CYRANO** (Haase 1987)

Refinements

- **GT** (Epstein 1988)
- **Graffiti** (Fajtlowicz 1988)
- **HR** (Colton 1999)

Modern components

- Abductive and inductive logic learning
- Inductive process modeling
- Statistical-relational learning
Computational Creativity

Theoretical Models

- Conceptual spaces (Boden)
- Domain, individual, field, interaction (DIFI) (Feldman)
- Curiosity (Saunders)
- Perceptual Creativity (Colton)
- ...

Aspects

- Artifacts
- Processes
- Expectation
- Emotion
- Socialization
- Novelty and value
- Generate and test loop
- ...

expressiveintelligencestudio
Creative Art Systems

- AARON (Cohen)
- NEvAr (Machado)
- Digital Clockwork Muse (Saunders)
- EMI (Cope)
- MINSTREL (Turner)
- The Painting Fool (Colton)
Recap of Related Work

- Game Design
  - Textbook + Call for more structure
  - Game studies + AI

- Discovery and Creativity
  - Models of Discovery + Systems
  - Computational Creativity + Systems
**Tableau Machine**
Logical game design
Game generation
Elementary discovery in game design

PRIOR WORK
Tableau Machine

- Experience formalizing an expressive domain

- Generate-and-test
  - Design grammars
  - Image analysis

- Learn long-term patterns in sensor data to stay relevant
BIPED: Computational support for play testing game sketches

- Designer
  - Analysis Engine
  - Game Sketch
  - Game Engine
  - formal rule system
  - playable prototype
  - design insight
    - play traces implied properties exploits puzzle solutions
    - play traces engagement fun hesitation
  - constraints and queries
  - human subjects
Example Game: *DrillBot 6000*

```
happens(mine(a1), 0).
happens(drain, 1).
happens(drain, 2).
happens(trade, 3).
happens(mine(a2), 4).
happens(mine(a0), 5).
happens(down_to(a), 6).
happens(mine(corps), 7).
happens(mine(c0), 8).
happens(down_to(c), 9).
happens(down_to(f), 10).
happens(up_to(c), 11).
happens(up_to(a), 12).
happens(down_to(c), 13).
happens(down_to(f), 14).
```
Movement mechanic from *DrillBot 6000*

```prolog
pos(base). pos(a). pos(b). pos(c). ...

game_state(position(P)) :- pos(P).

game_event(up_up(P)) :- pos(P).
game_event(down_do(P)) :- pos(P).

initiates(down_to(P),position(P)) :- pos(P).
terminates(down_to(_),position(Prev)) :-
    holds(position(Prev)).

initially(position(base)).
```
Logical User-Interface Programming

- UI bindings in *DrillBot 6000*

```prolog
ui_title('DrillBot 6000').
ui_space(P) :- pos(P).
ui_space(inventory).
ui_token(db6k).
ui_location(db6k,P) :- holds(position(P)).
ui_triggers(ui_click_space(P), down_to(P)).
ui_triggers(ui_click_space(base), refuel).
```
Capabilities

- Syntactic properties
  - Design validation

- Semantic properties
  - Trace harvesting
  - Rule set debugging
  - Win-ability verification
  - Reachability analysis
  - Uniqueness verification of puzzle solutions
  - Testing a game before you ever make a UI

- Induction on semantics
  - Player-model construction
Game Generation

- BIPED-tech is great for testing game ideas, but who generates them in the first place?
  - Need a “design grammar” for games

- Propositional game generator experiment
  - Generation of rule systems is feasible.
  - Needs higher-level building blocks:
    - Multi-clause rules
    - Multi-rule mechanics
    - Higher-level mechanics
“Movement between underground caverns” in *Drillbot 6000*

% positions (caverns)
pos(base).
pos(a).
pos(b).
pos(c).
% links (drillable routes)
link(base,a).
link(a,b).
link(a,c).
% event preconditions
possible(down_to(Dst)) :-
  holds(position(Src)),
  link(Dst,Src).
...

A general “network navigation” design pattern at the code level
- Setting: predicate `room(R)`
- State: `location(R)` such that `room(R)`
- Setting: `doorway(R1,R2)` such that `room(R1)` and `room(R2)`.
- Event: `move_to(R)` is possible only if `room(R)` and you `location` adjacent room, as judged by `doorway`
Recap of Prior Work

- *Tableau Machine*
- Logical game design
- Game generation
- Elementary discovery in game design
New Theories
System Architecture
Experimental Validation
Timeline

PROPOSED WORK
Reviewing the Knowledge Level
A Game Design Meta-Theory
A Knowledge-Level Account of Game Design

- **Agents**
  - Designers!

- **Actions**
  - Game actions
  - Play actions
  - Design actions

- **Goals**
  - Discovery of design-level knowledge
Reflexive Creativity

- My conjecture on creativity:
  - Creativity is the **rational pursuit of curiosity** that results in a surprise.

- Mash up some theories:
  - If game designer aim to discover...
  - And discovery is way to satisfy curiosity...
  - **Maybe creative game designers make games to help them discover!**
System Overview: Exterior

Core Discoverer

System Author

Designer Peers

Design Tools

Human Players
System Overview: Interior

- Operational Knowledge
- Discovery Notebook
- Artifact Library
- Design Theory

Core Discoverer
Symbol-level Implementation

- Implement as a production system
  - Facts, rules, and a rule engine/executive

- Operational knowledge:
  - Fixed rule set

- Design theory:
  - Mutable rule set, Mutable fact-base

- Artifact library:
  - Append-only fact-base

- Discovery Notebook
  - Mutable fact-base
Operational Knowledge

- Remember those scientific **knowledge structures** and scientific **processes**? I’ve translated them to game design.

- Layered mapping:
  - Scientific knowledge structures and processes
  - Game design knowledge structures and processes
  - Data structures and operations
  - Production rules and facts
% entry for an generated sequel to DrillBot 6000
game(db6k_mk2, [
  construction(expand_map(drillbot6000)),
  rules({'BIPED-compatible rule set'}),
  design_annotations(expand_map_seed(12391))]).

% a player model
player(energy_hog, [
  construction({'production rule used to produce this player'}),
  rules({'internal state, predicate transformers, BIPED-compatible play-hook clauses'}),
  design_annotations({'...'})).

% entry for an instance of play
play_instance(pairing23423,[
  game(db6k_mk2),
  player(energy_hog),
  pairing_rules({'choices the system had to make to glue the player to the game'}).
Design Theory

- What-is knowledge
  - Design patterns (named and detailed game and play structural elements)
    - Recall the “network navigation” pattern
  - Trace predictors
    - “If the game contains pattern X, then Y should be found in the trace”

- How-to knowledge
  - *When-to-always* and *when-to-never* perform certain design actions under certain conditions
Raw Design Theory Examples

% a movement mechanic
mechanic(movement_between_rooms(R,D),[
  dungeon_map(R,D),
  game_state(at(agent/1,R)),
  game_event(moves_to(agent/1,R)),
  \{trigger logic\}
]).

% player’s view of the game state as a player character
player_construct(pc_avatar(PcPred),[
  binder(pc_avatar(PcPred)),
  pc_avatar(Pc),
  pp_mapper(in(Pc,X),out(X))\}].

% trace property predictor
trace_implication(happens(victory,T2),happens(boss_kill,T1)) :-
  T2 <= T1, mechanic(boss_kill_victory).

% how-to
designers_never(game_apply(expand_map),game_apply(expand_map)).
Discovery Notebook

- Starts empty
- Contains
  - Outstanding experiments
  - Expectations
  - Agenda
  - ...
- General working memory
- Usage dictated by operational knowledge
Actions:

- **Design actions:**
  - *Manipulate* game, player, and play instance models
  - *Solicit* a game or play trace from an automated tool or a human player

- **Discovery actions:**
  - *Propose* new game, player, and play instance structural elements or production constraints
  - *Look* for examples of new patterns in old artifacts
  - *Verify* (prove?) trace predictors
Design Tools

- Design validator
- Gameness-check
- Trace-finder
- Human player trace harvester
- Logical debugger
- Misc. statistical-relational learning tools

Potential add-ons from Mark’s research:
- Alternate trace-finding back-ends
- Query suggester and answerer
- Rule visualizer
Experimental Validation – Q1

How does an intelligent game designer function?

- “games” – Ask some players if the machine-designed games felt like real games
  - Do the players think the games feel real?
- “intelligent” – Ask some expert designers to perform some discovery, and record the result (using same tools).
  - Does my system rediscover it?
  - Did my system discover something beyond it?
- “game design” – Ask some expert designers to design some games and record the result (using same tools).
  - Does the designer think the games feel real?
  - Does the system produce the same kind of games?
What does such a system imply for the relationship between discovery and expressive domains?

- Test the theory by testing the systems designed according to it.
  - Toggle various elements of the architecture to see what is really to blame for the interesting behavior

- Need implications in-hand to propose concrete experiments
Time Line

- **Year one:** focus on stretching game design into a science-like practice, automation comes at the very end
  - Summer 2009: play with more manual discovery
  - Fall 2009: implement scientific knowledge structures
  - Winter 2010: implement the scientific processes
  - Spring 2010: integrate the system, close the loop

- **Year two+:** focus on architectural experimentation, system evolution, and generalizing to my theoretical contributions
  - Summer 2010: plan the dissertation
  - Fall 2010: perform the experiments
  - Winter 2011: synchronize experimental results with plan
  - Spring 2011: dissertation writing
  - Summer 2011: final polish and defense
Recap of Proposed Work

- Improve upon some new theories
- Implement system according to proposed architecture
- Validate my intelligent game designer in experiments
- Generalize from working system to more general theories
EPILOGUE

Revisiting Buchanan’s criticisms
Epilogue

Bruce Buchanan (in AAAI-2000 Presidential Address) says of existing creative systems...

- (1) they do not accumulate experience, and thus, cannot reason about it;
- (2) they work within fixed frameworks including fixed assumptions, and criteria of success;
- and (3) they lack the means to transfer concepts and methods from one domain to another.
Thesis proposal: PROPOSED

THANKS
Detailed First Year Plan

- **Summer 2009**
  - Flesh out first system architecture (mostly complete)
  - Perform manual discovery with the raw tools and representations (already started)
    - Produce example outputs
    - Document the manual process
- **Fall 2009**
  - Implement knowledge structures
    - Games, player, play instances, trace predictors
    - Structural elements (setting constructs, mechanics, player predicate transformers)
  - Perform manual discovery with richer representations
- **Winter 2010**
  - Implement processes
    - Drivers/scripts for external tools
    - Action sequences (“get a trace, then induce a player model”)
    - Heuristic processes selection (“try verifying a trace prediction”)
  - Perform manual discovery using large-scale processes as the individual move
- **Spring 2010:**
  - Write up preliminary view of game design as a scientific domain (with structures and processes in-hand)
  - Plan expert and player evaluations
  - Create minimal closed-loop system
    - Theory goes in, improved theory comes out; also, games were produced
  - Improve system by building larger scale processes
Detailed Second Year Plan

- **Summer 2010**
  - Write up initial results and architecture of integrated system
  - Perform final literature review
    - Game design, discovery, and generation in expressive domains
    - Digital media (and other fields outside my own) for reference on expressive artifacts
  - Formulate initial implications between discovery and expressive domains
  - Design experiments to test implications
  - Produced detailed dissertation outline

- **Fall 2010**
  - Carry out experiments

- **Winter 2011**
  - Reconcile experimental results with theory, adjust claims
  - Start dissertation writing

- **Spring 2011**
  - Dissertation writing
  - Clarifying experiments

- **Summer 2011**
  - Final polish and defense