

Lecture 15

Procedural Content Generation

Important Lessons for Today

- Procedural content is **harder**, not easier
 - You must already know your *design patterns*
 - Controlling *difficulty* is a potential challenge
 - *Unwinnable levels* are also a challenge
- Many procedural approaches are **ad hoc**
 - Designed for specific games
 - Limited adaptability to other games
- Procedural generation is a **stretch goal**

In the Beginning, There Was *Rogue*



@: You (dark) You now have chain mail <13> (g).
Health You dispatched the kobold, catching it unaware.
Nutrition You now have a scroll entitled "herba pus flem nidge" (h).
Str: 12 Armor: 2?
Stealth range: 4
!: A blue potion
*: 99 gold pieces

Roguelike Genre

- Classic RPG style
- Procedural dungeons
- **Permadeath**

-- Depth: 1 -- Explore Rest (z) Search (s) Menu Inventory

A Brief History of Roguelikes

- Precursors (1978)
 - *Beneath Apple Manor*
 - *Dungeon* (unfamous one)
 - Like *Rogue*, but less famous
 - Limited content generation
- *Rogue* (1980)
 - Multiplatform launch
- Immediate Copycats
 - *Hack* ('82), *NetHack* ('87)
 - *Moria* ('83), *Angband* ('90)
 - All very close in playstyle
 - Open source development
 - Middle Earth themed
- *Island of Kesmai* (1985)
 - *Legends of Kesmai* (1996)
 - Massively (~80) multiplayer
 - But content less procedural
- The Modern Revival
 - Relaxing RPG requirement

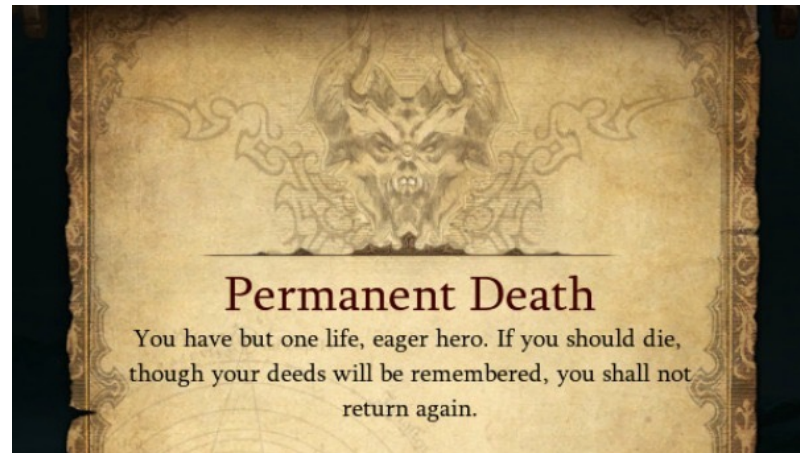
Changing Perspectives on Permadeath

Advantages

- Greater challenge
 - Used as a badge of honor
- Higher emotional stakes
 - Easy to instill fear & horror

Disadvantages

- Greater discouragement
 - Seen as a personal failure
- Missed game content
 - Cannot progress in story



Changing Perspectives on Permadeath

Advantages

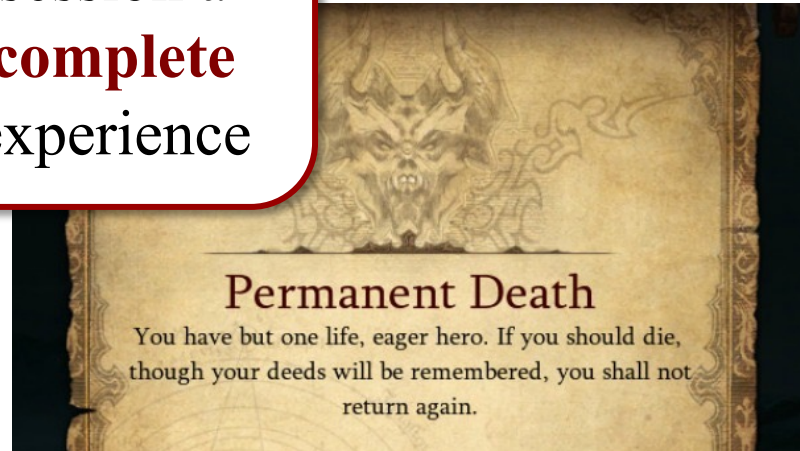
- Greater challenge
 - Used as a teaching tool
- Higher emotional investment
 - Easy to identify with the character

Make dying expected & **inevitable**

Make each session a **complete** experience

Disadvantages

- Greater discouragement
 - Seen as a personal failure
- Missed game content
 - Cannot progress in story



Changing Perspectives on Permadeath

Advantages

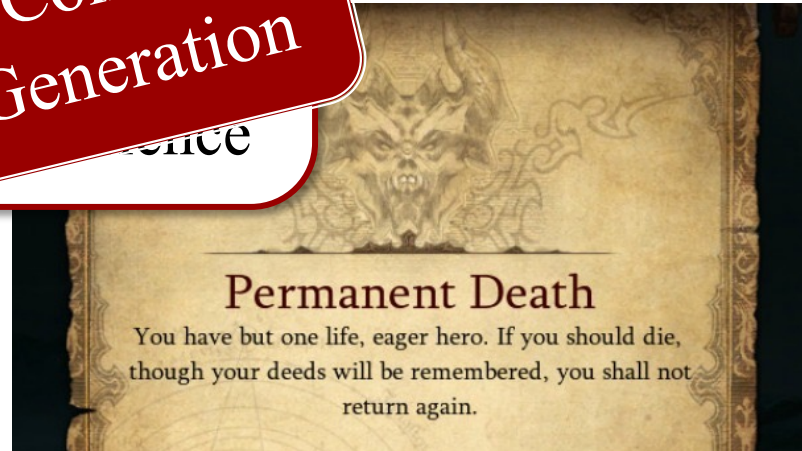
- Greater challenge
 - Used as a teaching tool
- Higher emotional investment
 - Easy to integrate

Make dying expected & **inevitable**

Content Generation

Disadvantages

- Greater discouragement
 - Seen as a personal failure
- Missed game content
 - Cannot progress in story



Issues with Roguelikes

- Design is often **horizontal**
 - Many verbs, game elements
 - Little coupled behavior
- Each play is a **slice**
 - Access to limited elements
 - Work with what you get
- “Expensive” to create
 - Requires a lot of content
 - But historically just text
- Difficult to balance

WEAPON (Table 1)					
Dagger	COST	WGT	PROB	MATL	APPEARANCE
orcish dagger	\$4	10	12	IRON	crude dagger
dagger	4	10	30	IRON	--
silver dagger	40	12	3	SILV	--
athame	4	10	0	IRON	--
elven dagger	4	10	10	WOOD	runed dagger
Knife	COST	WGT	PROB	MATL	APPEARANCE
worm tooth	2	20	0	NONE	--
knife (shito)	4	5	20	IRON	--
stiletto	4	5	5	IRON	--
scalpel	6	5	0	METL	--
crysknife	100	20	0	MINL	--
Axe	COST	WGT	PROB	MATL	APPEARANCE
axe	8	60	40	IRON	--
battle-axe	40	120*	10	IRON	double-headed axe
Pick-axe	COST	WGT	PROB	MATL	APPEARANCE
pick-axe	50	100	tool	IRON	--
dwarvish mattock	50	120*	13	IRON	broad pick
Short sword	COST	WGT	PROB	MATL	APPEARANCE
orcish short sword	10	30	3	IRON	crude short sword

Issues with Roguelikes

- Design is often **horizontal**
 - Many verbs, game elements
 - Little coupled behavior
- Each play is a **slice**
 - A
 - V
- “Expensive” to create
 - Requires a lot of content
 - But historically just text
- Difficult to balance

WEAPON (Table 1)

Dagger	<u>COST</u>	<u>WGT</u>	<u>PROB</u>	<u>MATL</u>	<u>APPEARANCE</u>
orcish dagger	\$4	10	12	IRON	crude dagger
dagger	4	10	30	IRON	--
silver dagger	40	12	3	SILV	--
athame	4	10		IRON	--
					runed dagger
					<u>APPEARANCE</u>
					--
				5 IRON	--
scalpel	6	5	0	METL	--
crysknife	100	20	0	MINL	--
Axe	<u>COST</u>	<u>WGT</u>	<u>PROB</u>	<u>MATL</u>	<u>APPEARANCE</u>
axe	8	60	40	IRON	--
battle-axe	40	120*	10	IRON	double-headed axe
Pick-axe	<u>COST</u>	<u>WGT</u>	<u>PROB</u>	<u>MATL</u>	<u>APPEARANCE</u>
pick-axe	50	100	tool	IRON	--
dwarvish mattock	50	120*	13	IRON	broad pick
Short sword	<u>COST</u>	<u>WGT</u>	<u>PROB</u>	<u>MATL</u>	<u>APPEARANCE</u>
orcish short sword	10	30	3	IRON	crude short sword

Procedural Content for Modern Games?

Modern Roguelikes: *Spelunky*



Modern Roguelikes: *FTL*



Modern Roguelikes: *Roundguard*



Main Types of Procedural Content

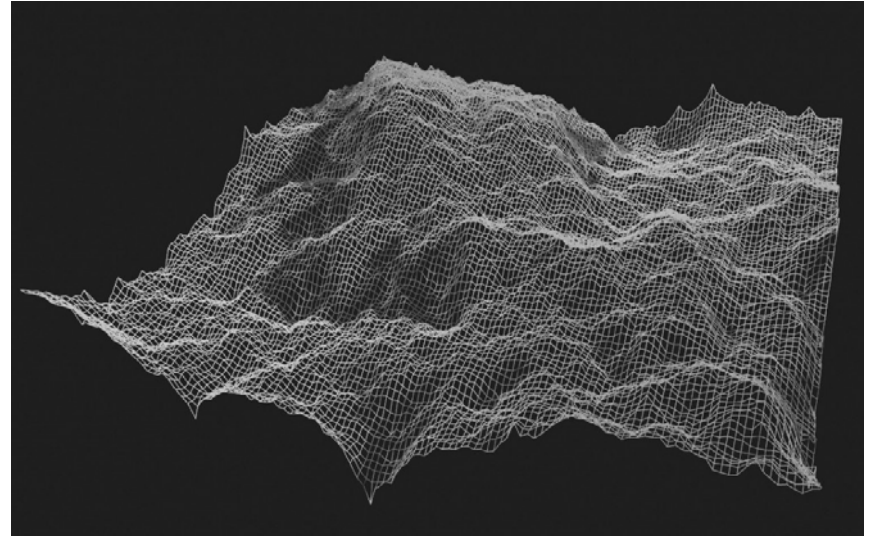
- Simulation
- World Generation
- Puzzle Generation
- Story Generation
- Dynamic Challenges
- Adaptive Difficulty



Procedural Content Wiki:
<http://pcg.wikidot.com>

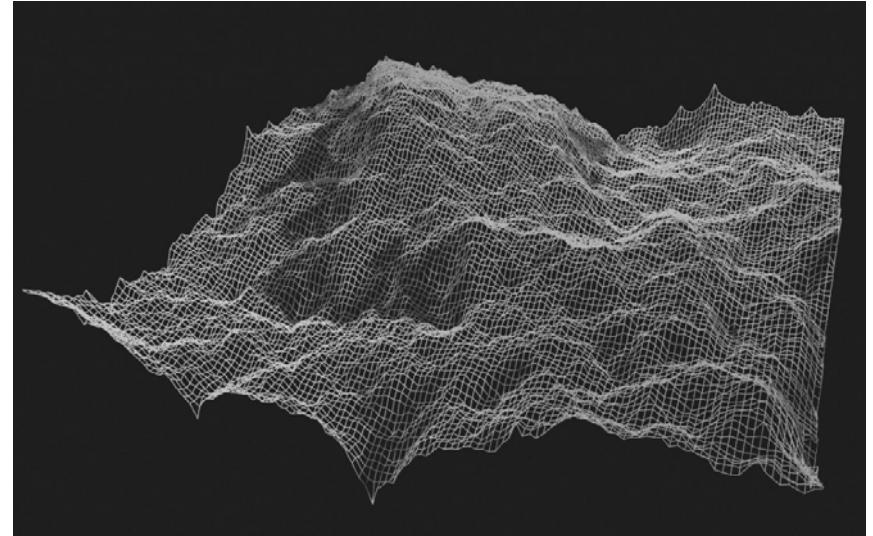
Simulation

- Complexity appears random
- Often a physical process
 - Fires, Fluids, Weather
 - Terrain generation
 - Artificial life
- **Teleological**
 - Run the full simulation
 - Accurate; hard to control
- **Ontological**
 - Create reasonable output
 - Inaccurate; easy to control



Simulation

- Complexity appears random
- Often a physical process
 - Fires, Fluids, Weather
 - Terrain generation
 - Artificial life



- **Teleological**

• **Scientific Computing**
• Difficult to control

- **Ontological**

• **Ad Hoc Algorithms**
• Easy to control

Simulation

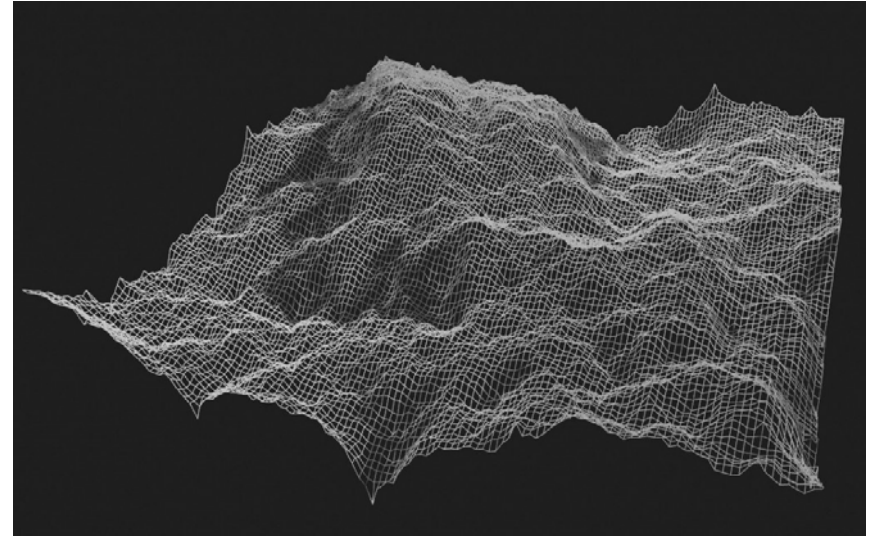
- Complexity appears random
- Often a physical process
 - Fires, Fluids, Weather
 - Terrain generation
 - Artificial life

- **Teleological**

• **Scientific Computing**
• Difficult to control

- **Ontological**

• **Ad Hoc Algorithms**
• Easy to control



- Minimal effect on gameplay
 - Often largely aesthetic
 - Hard to control difficulty
- Lot of work for little payoff

World Generation

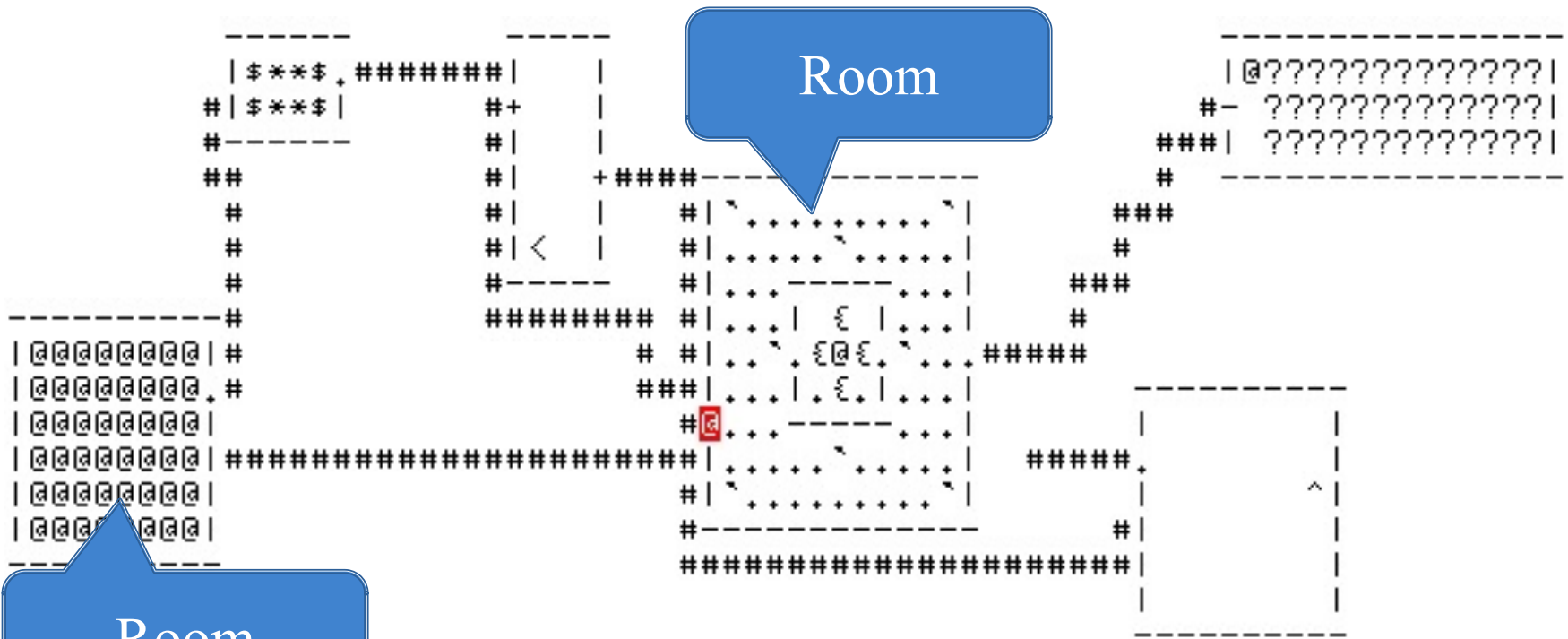
- Often thought of as map generation
 - But really generation of game *geography*
 - Particularly broad category of PCG
- **Basic Format**
 - Start with basic geography building blocks
 - Include combination rules for blocks
 - Build until reach a stopping point
- Algorithms vary widely

Example: NetHack

```
-----
| $$$, #####| |
#| $$$| #+ |
#----- #| |
## #| +####-----
# #| | #| \ . . . . . \ | ###
# #| < | #| . . . . \ . . . . | #
# #----- #| . . . - - - . . . | ###
-----# ##### #| . . . | { | . . . | #
| @ @ @ @ @ @ @ @ | # # # | . . . { @ { . . . | #####
| @ @ @ @ @ @ @ @ | # | . . . | { . | . . . |
| @ @ @ @ @ @ @ @ | ##### | . . . - - - . . . |
| @ @ @ @ @ @ @ @ | # | . . . \ . . . \ | #####
| @ @ @ @ @ @ @ @ | # | \ . . . . . \ | #
-----#----- #|
##### |
| | |
| | ^ |
| | |
-----
```

Izchak the Curator St:18/11 Dx:16 Co:17 In:18 Wi:18 Ch:17 Lawful
Dlv1:8 \$:94041 HP:217(234) Pw:190(195) AC:7 Exp:30

Example: NetHack



Izchak the Curator St:18/11 Dx:16 Co:17 In:18 Wi:18 Ch:17 Lawful
 Dlv1:8 #:94041 HP:217(234) Pw:190(195) AC:7 Exp:30

Example: NetHack

The image displays a section of a NetHack ASCII art map. A character, represented by a red square with a white 'Q', is positioned in a room. The room is bounded by walls and contains various objects like a book and a chest. A hallway leads away from the room. Three callout boxes are overlaid on the map: a blue box labeled 'Room' pointing to the character's current location, a yellow box labeled 'Hallway' pointing to the adjacent passage, and another blue box labeled 'Room' pointing to a larger room on the left side of the map. The map uses standard ASCII characters for walls, floors, and objects.

| \$\$\$, ##### | |
#| \$\$\$ | #+ |
#- | # |
| # | +####-----
| # | #| \ |
| # | #| |
| # | #| < | #| |
| # | #| # | \ |
-----# | # | #| |
| @ @ @ @ @ @ @ @ | # # | . . . | { | . . . | #
| @ @ @ @ @ @ @ @ | # ## | . . . | { @ { . . . | #####
| @ @ @ @ @ @ @ @ | # @ | . . . | { . | . . . | #
| @ @ @ @ @ @ @ @ | ##### | # | . . . | \ | #####
| @ @ @ @ @ @ @ @ | # | \ | #
| @ @ @ @ @ @ @ @ | # | # | | #
-----# | # | # | | #
|
#####

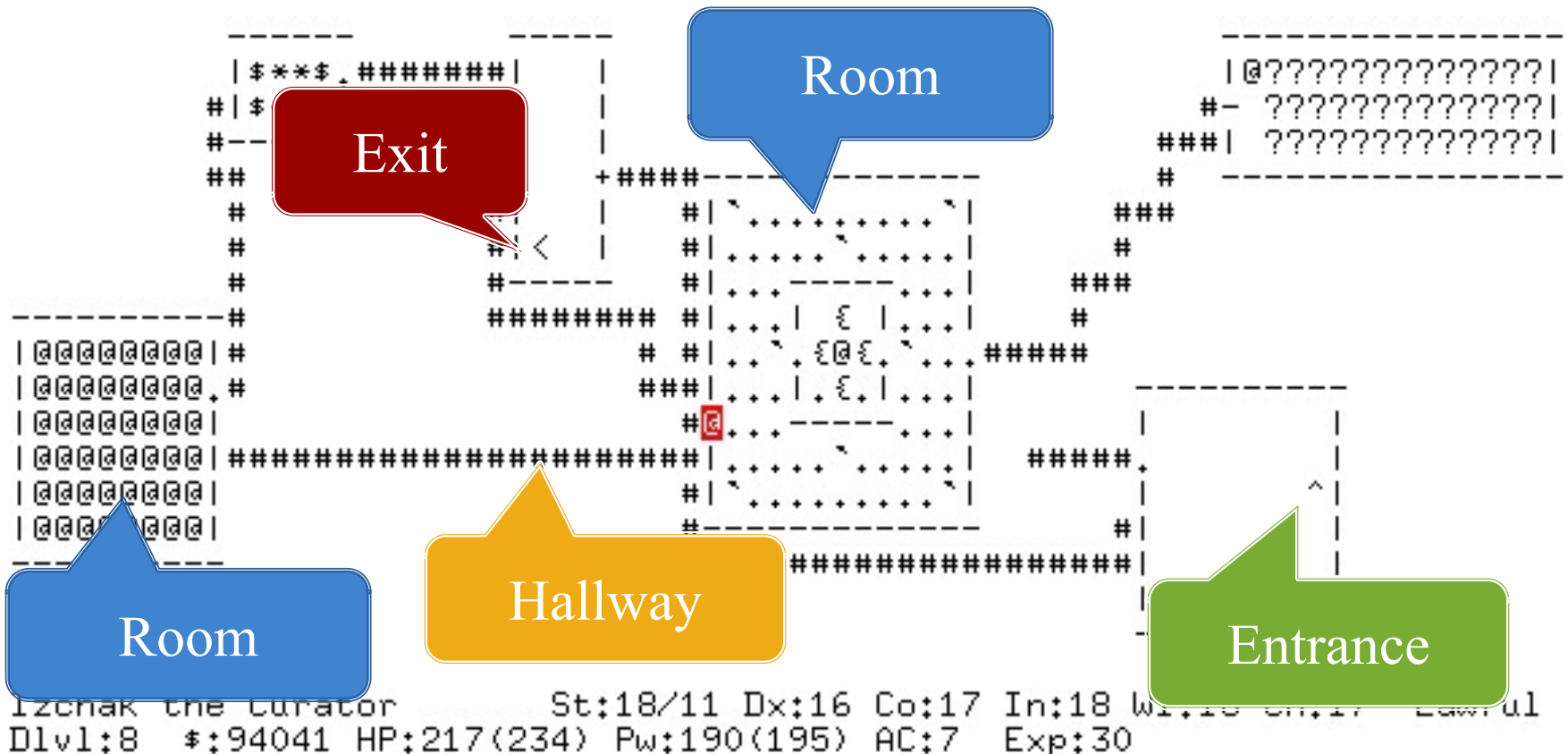
Room

Hallway

Room

Izchak the Curator St:18/11 Dx:16 Co:17 In:18 Wi:18 Ch:17 Lawful
Dlv1:8 #:94041 HP:217(234) Pw:190(195) AC:7 Exp:30

Example: NetHack



Room

Exit

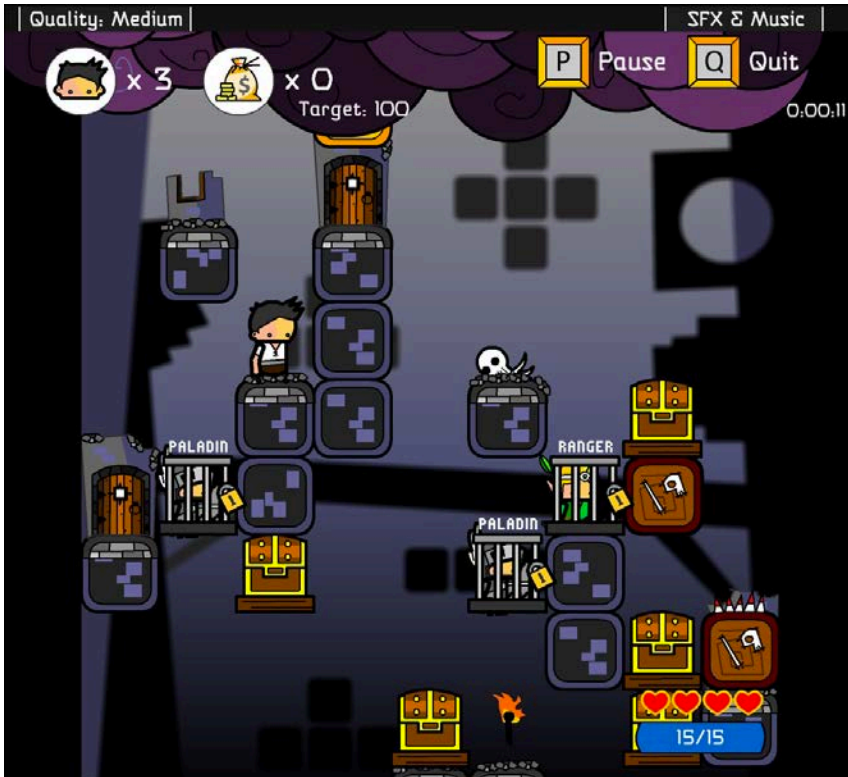
Room

Hallway

Entrance

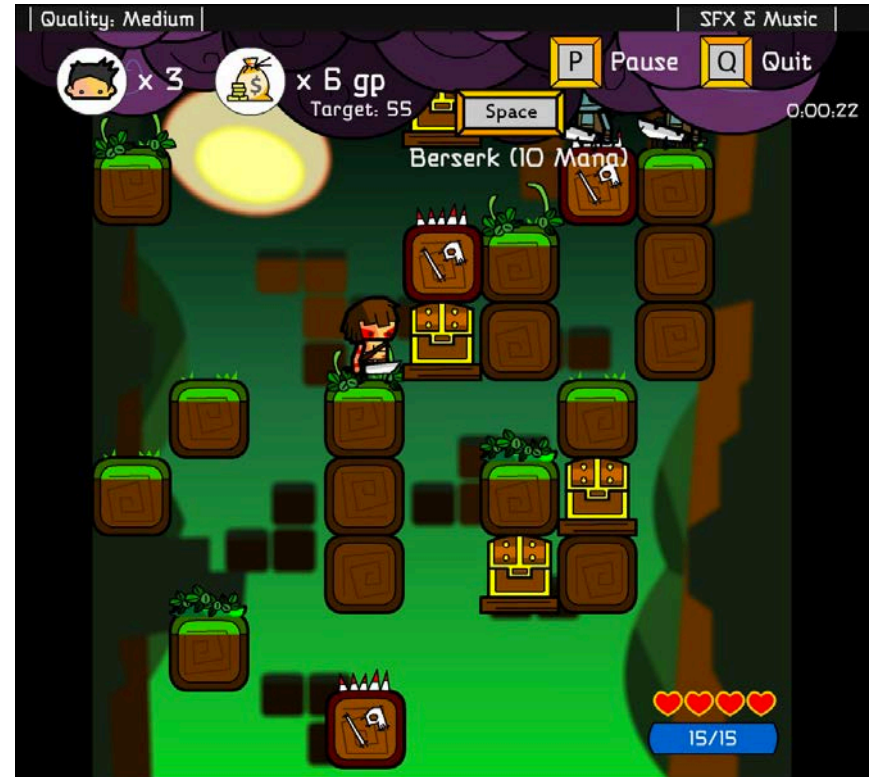
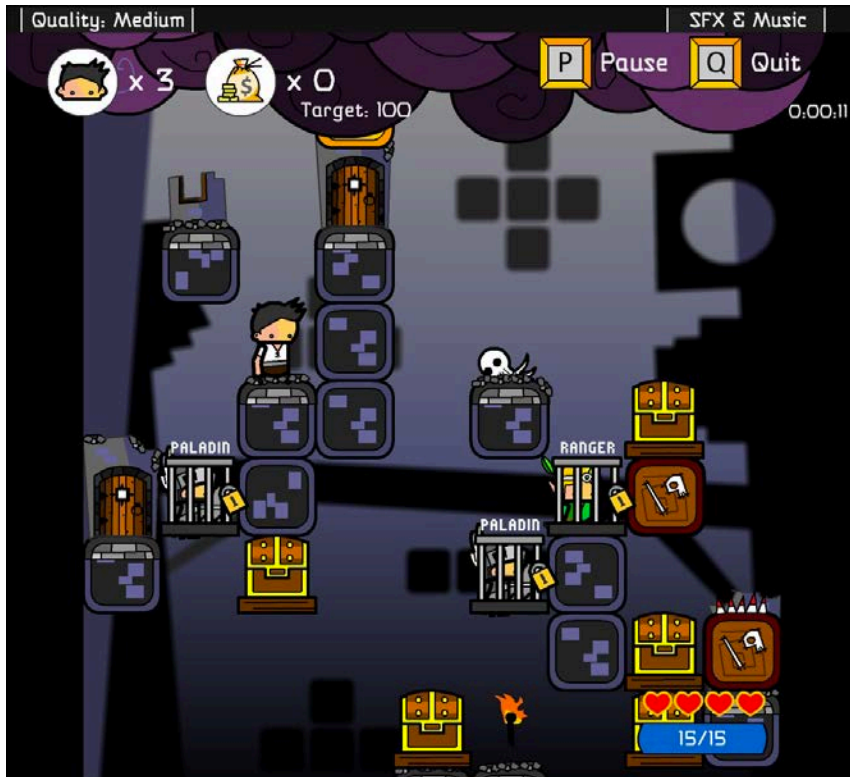
Izchak the Curator St:18/11 Dx:16 Co:17 In:18 W:18 S:17 Lawful
Dlv1:8 #:94041 HP:217(234) Pw:190(195) AC:7 Exp:30

Example: *Vertical Drop Heroes*

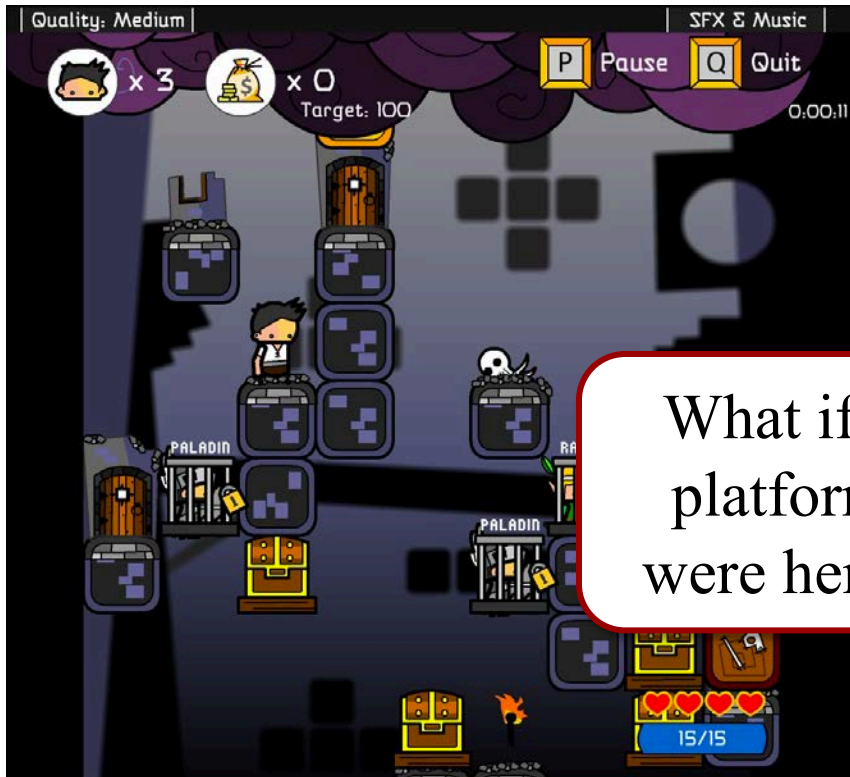


- **Movement**
 - Can move left-right
 - Down arrow to stomp/fall
 - Cannot jump at all!
- **Combat**
 - Space to fire weapon
 - Weapon depends on class
 - Free cage to switch class
- **Goal**
 - Collect treasure
 - Reach (a possible) exit

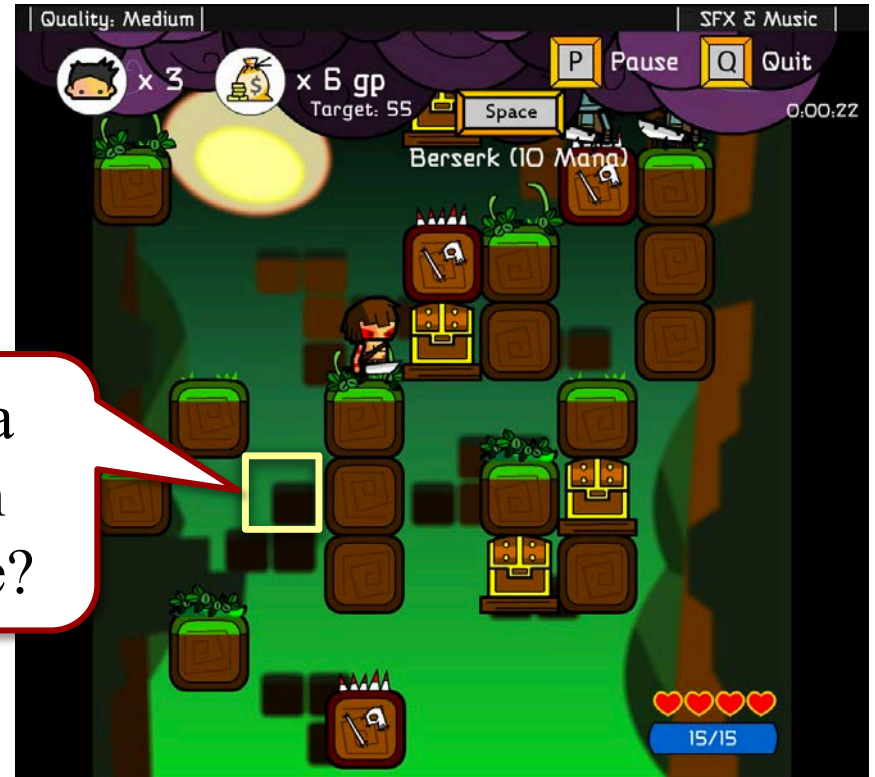
Example: *Vertical Drop Heroes*



Example: *Vertical Drop Heroes*

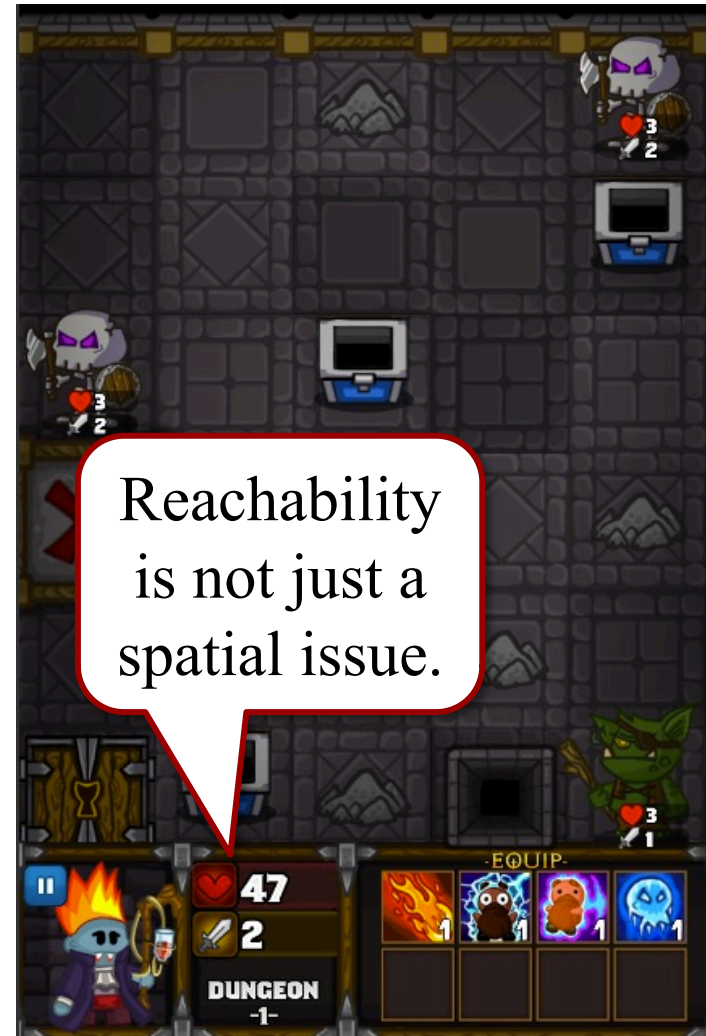


What if a platform were here?



The Reachability Problem

- Levels are effectively graphs
 - Edges are player choices
 - Choices are discretized
 - Fully **connected** (why?)
- PCG might make a graph
 - with a lot of dead ends
 - with a lot of backtracking
 - that is **unconnected**
- Need to remember goal
 - Should always be reachable
 - Else, reset must be painless



Example: *Card Crawl*

Panic
Button



Ensuring Reachability

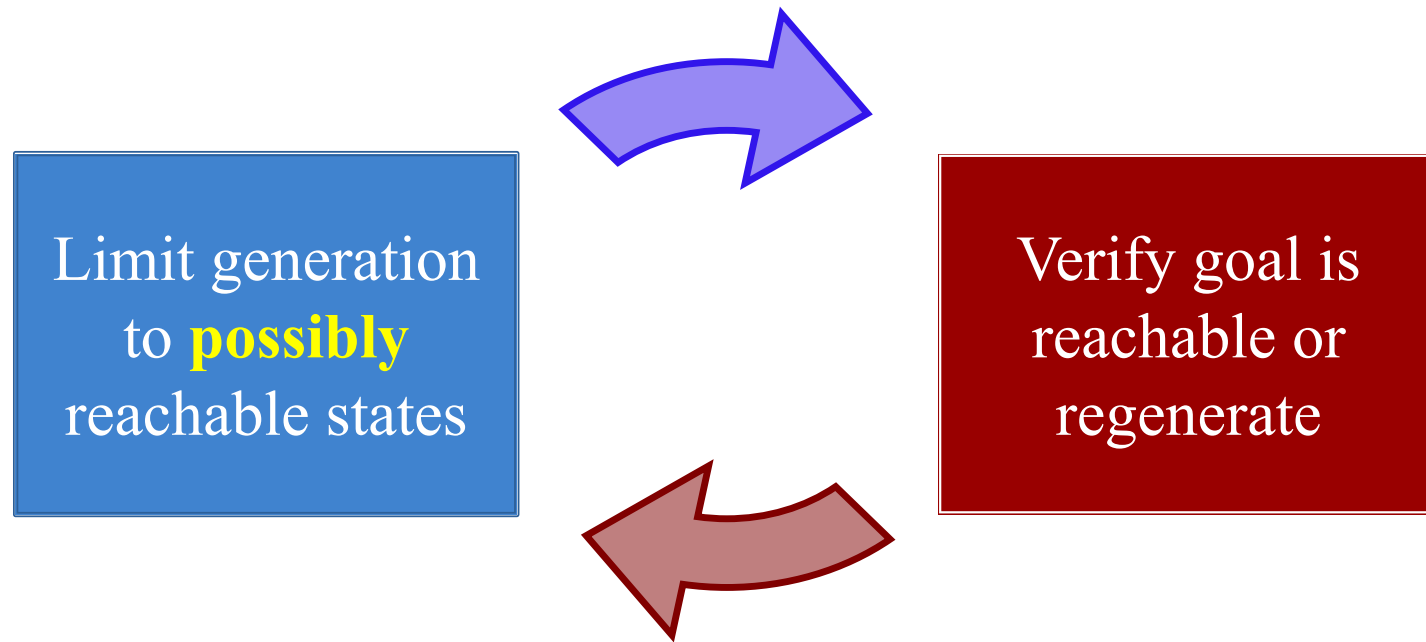
Two Options:

Limit generation
to reachable
game states

Verify goal is
reachable or
regenerate

Ensuring Reachability

Two Options:



Grammars: A Formal Approach

• Notation

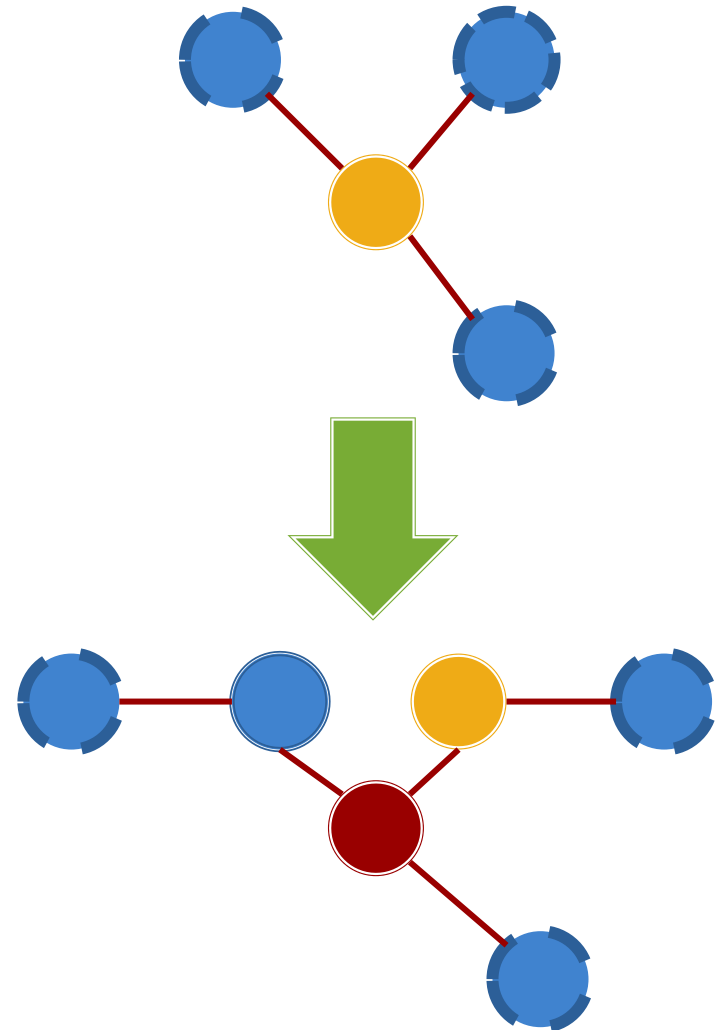
- Set \mathcal{N} of nonterminals
- Set Σ of terminal symbols
- Set \mathcal{P} of production rules
 - Have the form $A \Rightarrow B$
 - A, B are **words** of symbols
- To generate a value
 - Start with word XAY
 - Pick any rule $A \Rightarrow B$
 - Replace with XBY
 - Repeat until only terminals

Example

- $\mathcal{N} = \{S, B\}$
- $\Sigma = \{a, b, c\}$
- \mathcal{P} is the list of rules
 - $S \Rightarrow aBSc$
 - $S \Rightarrow abc$
 - $Ba \Rightarrow aB$
 - $Bb \Rightarrow bb$
- Possible **outputs**
 - $abc, aabbcc, aaabbbccc, \dots$

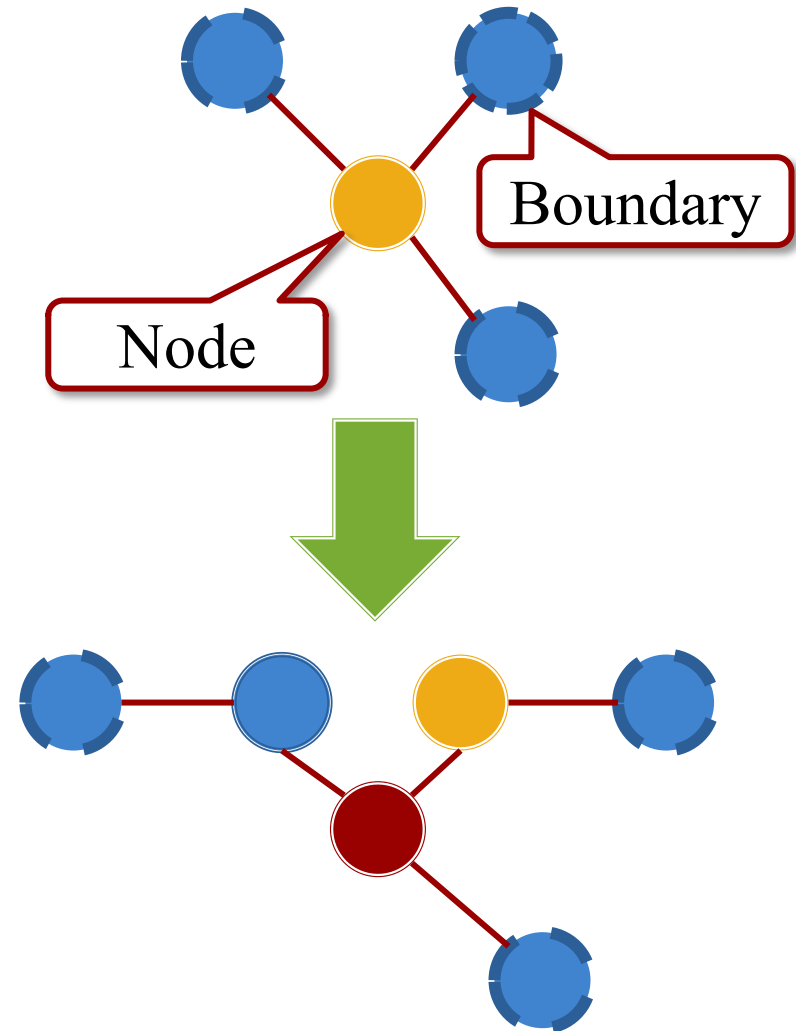
Grammars on Graphs

- Symbols are colored nodes
 - Either terminal or not
 - Edges replace word order
- Words are now graphs
 - Productions on subgraphs
 - LHS is node+boundary
 - RHS alters the node
- Output built as before
 - But rule matching harder
 - Graph equivalency



Grammars on Graphs

- Symbols are colored nodes
 - Either terminal or not
 - Edges replace word order
- Words are now graphs
 - Productions on subgraphs
 - LHS is node+boundary
 - RHS alters the node
- Output built as before
 - But rule matching harder
 - Graph equivalency

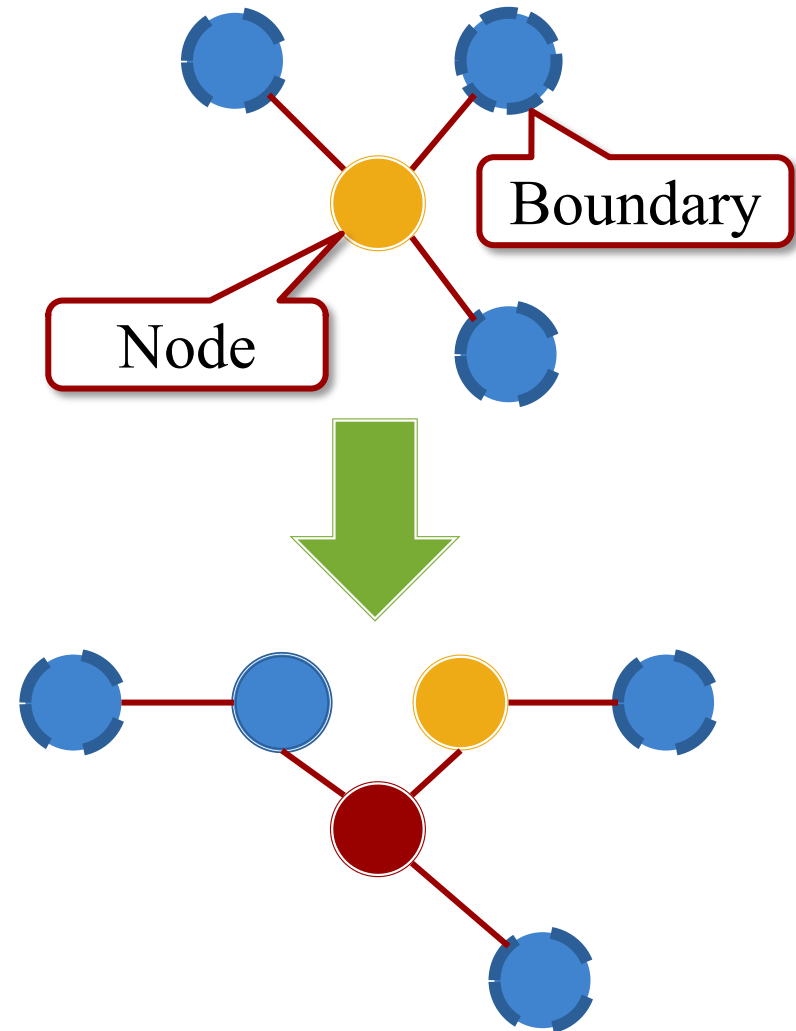


Grammars on Graphs

- Symbols are colored nodes
 - Either terminal or not
 - Edges replace word order
- Words are now graphs
 - Productions on subgraphs
- Output built as before
 - But rule matching harder
 - Graph equivalency

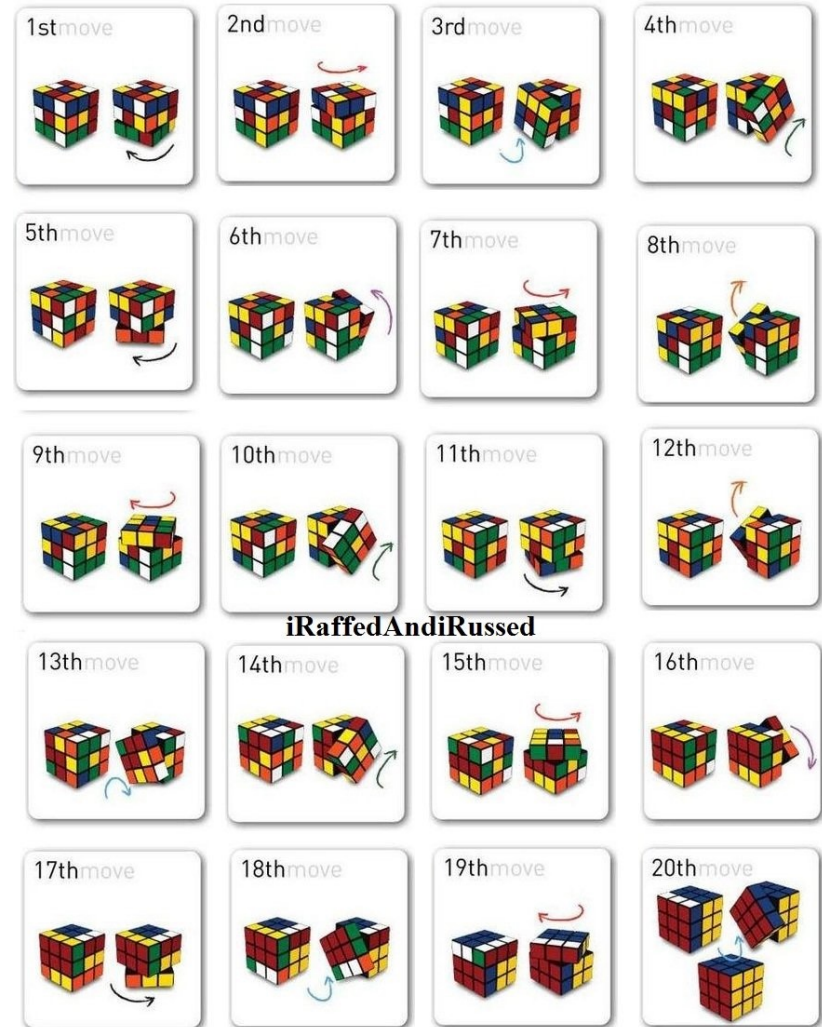
Game Geography is a graph

THIS alters the node

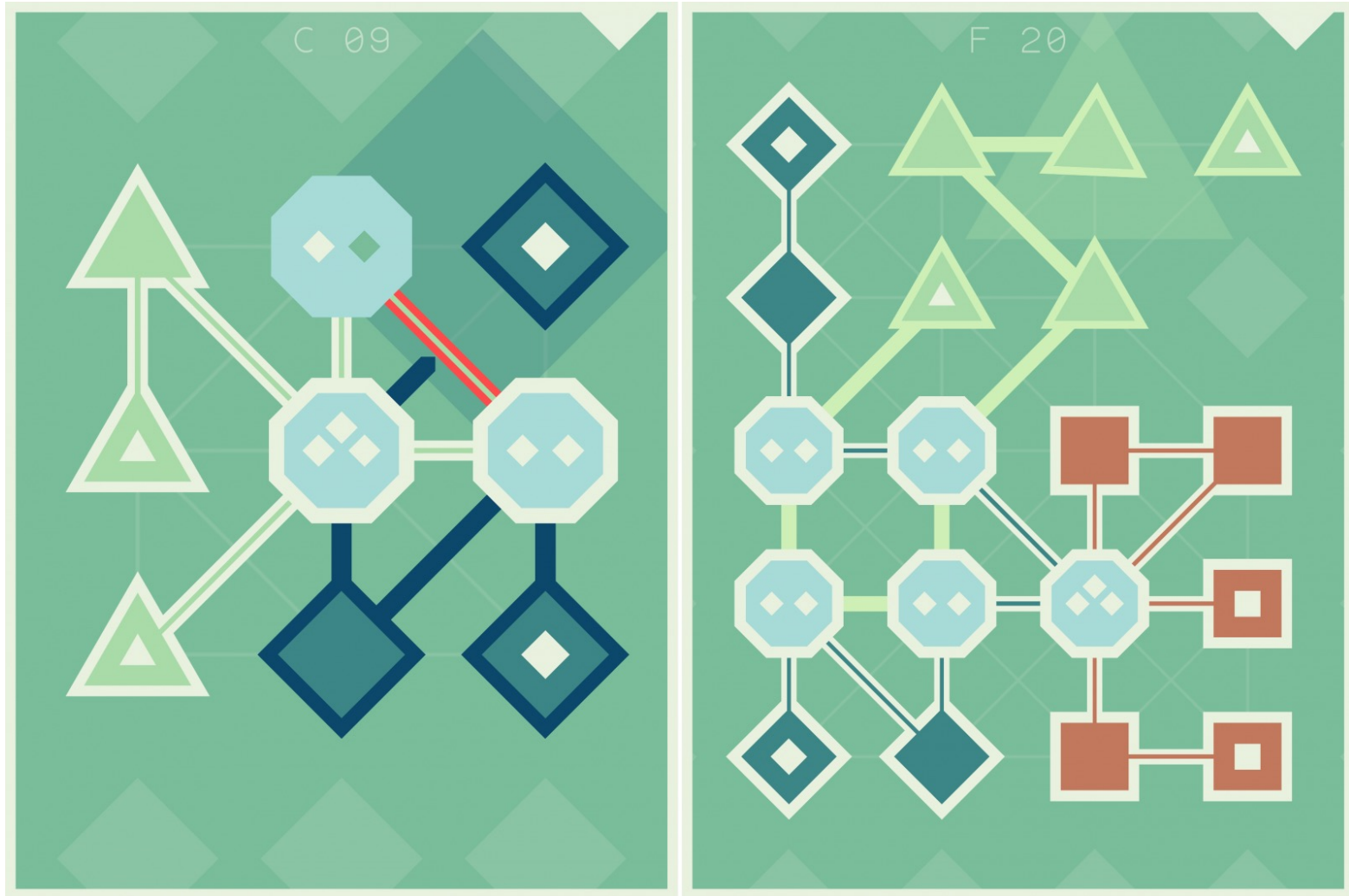


Puzzle Generation

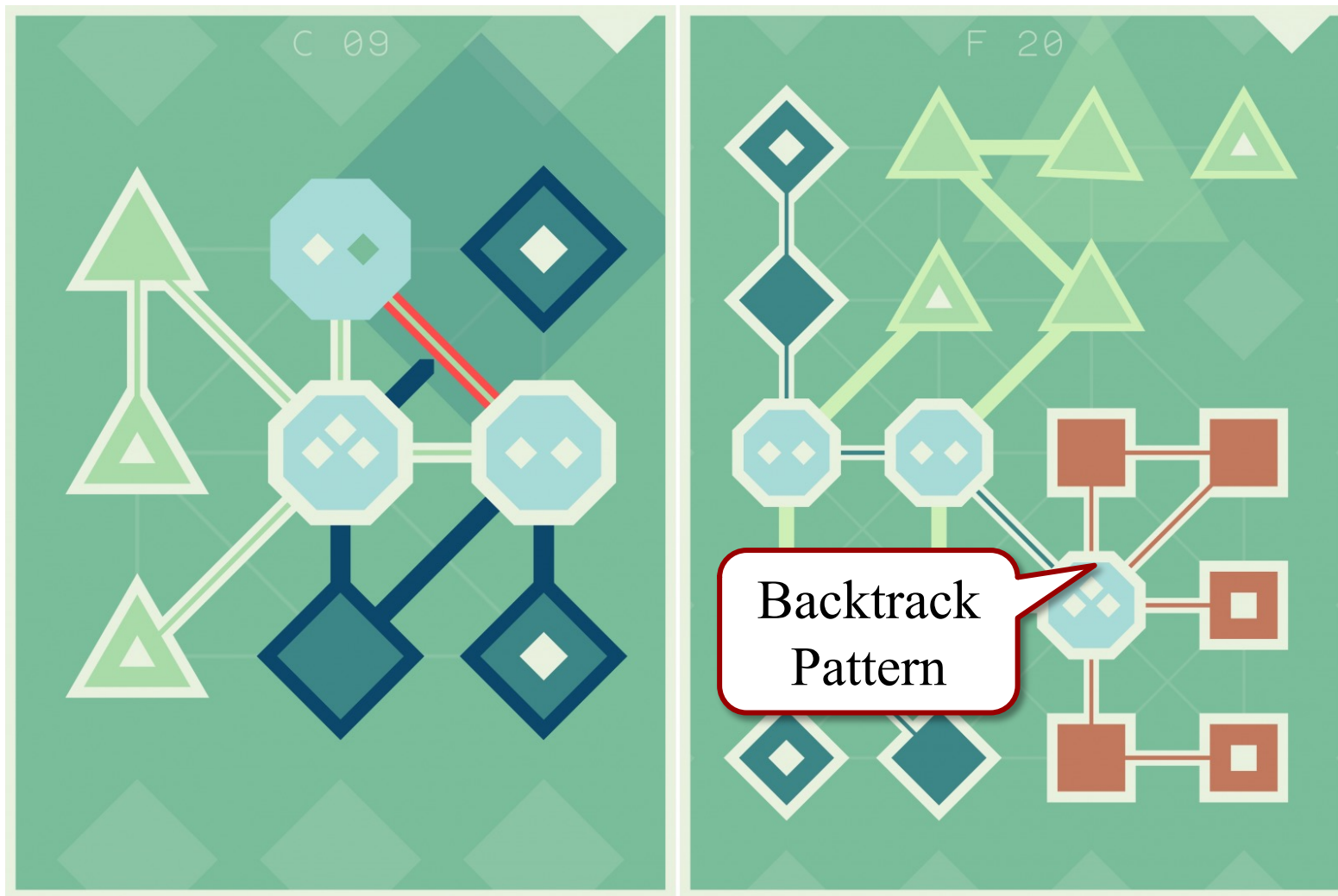
- Basic puzzle structure
 - Discrete actions/moves
 - Moves applied in sequence
 - **Goal:** get correct sequence
- Identify move sequences
 - Could be a loose category
 - Represent specific strategies
- Build up from sequences
 - Start from solved state
 - Invert moves (scrambling)
- Will require verification



Example: Lyne



Example: Lyne



Story Generation

- **Narrative** is tightly crafted
 - Must have emotional arc
 - Very hard to generate
- But **backstory** is looser
 - Collection of tales/subplots
 - Combine to form a story
 - Often displayed in a codex
 - Much easier to generate
- **Idea:** Create list of subplots
 - Pick some subset at a time
 - Mix with NLG techniques



Example: Dwarf Fortress



Natural Language Generation

- Function that outputs language
 - **Given:** complex set of data
 - **Outcome:** comment on data
 - Major area of CS research
- Comment requirements
 - Must be **simpler** than data
 - Should also be **natural**
- **Examples**
 - Sports commentary
 - Party combat chatter
 - Intelligent townfolk



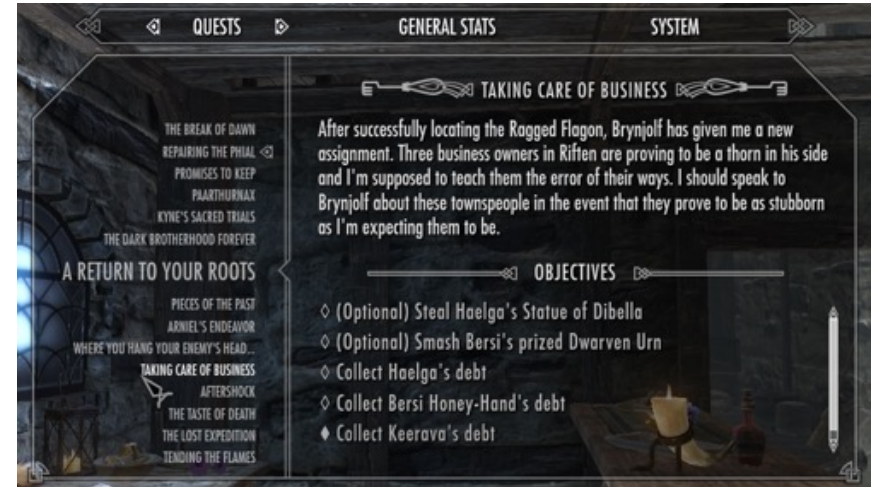
NLG and Story Dialogue

- Often a set of “canned” text
 - React to specific events
 - NPC picks text as appropriate
- Text is *parameterized*
 - “What do we do, <name>?”
 - “Someone killed <monster>!”
 - “That was <numb> days ago.”
- Choosing text to say
 - Favor important events?
 - Favor recent events?
 - Random (pull-toy)?



Skyrim's Radiant Quest System

- Geography includes NPCs
 - Mobile, removable location
 - Dialogue is also a space
- System “randomly” chooses
 - Quest giver
 - Quest location
 - Location's challenges
 - Quest redeemer
- Randomness is limited
 - Lists appropriate to quest
 - Depends on earlier actions



- Goals:
 - Send to unexplored areas
 - Adjust challenges to level
 - Can never be missed
- Largely a success

Skyrim's Radiant Quest System

- Geography includes NPCs
 - Mobile, removable location
 - Dialogue is also a space
- System “randomly” chooses
 - Quest giver
 - Quest location
 - Location's challenges
 - Quest redeemer



Guarantees reachability unexplored areas

- Randomness is limited
 - Lists appropriate to quest
 - Depends on earlier actions

- Adjust challenges to level
- Can never be missed
- Largely a success

But Sometimes a Problem



Dynamic Challenges

- Challenges that can change
 - Become easier or harder
 - Just be different
- **Example:** Autoleveling
 - NPCs have statistics
 - Adjust to character level
 - Difficulty always reasonable
 - Allows true “open” world
- Not always popular
 - Can lead to design recycling
 - Sense of risk is lost



Rat: Level 1

ATK	1
DFN	0
HP	5



Rat: Level 50

ATK	30
DFN	10
HP	90

Other Types of Dynamic Challenges

- **Composite Challenges**

- Encounter is a collection of NPCs, obstacles
- Add or remove individuals from encounter

- **Dynamic NPC AI**

- NPCs have a choice of AI scripts
- Choose one that matches the player

- **Player Boosting**

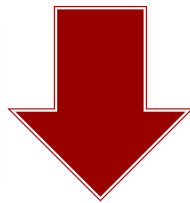
- Change result of player actions, interactions
- Modifications make challenges easier/harder

Assigning Dynamic Challenges

Player

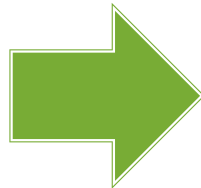


Extract feature
vector from
play history



$(a_1, a_2, a_3, \dots, a_n)$

Match the
challenge to
the play style



Procedural Content

Challenge



Parameterize
challenge
difficulty



$(b_1, b_2, b_3, \dots, b_k)$

Assigning Dynamic Challenges

Player



Challenge



Matching Function is
hardest to balance

Extract feature
vector from
play history



Match the
challenge to
the play style



Parameterize
challenge
difficulty

$(a_1, a_2, a_3, \dots, a_n)$



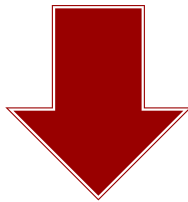
$(b_1, b_2, b_3, \dots, b_k)$

Adaptive Difficulty

Player

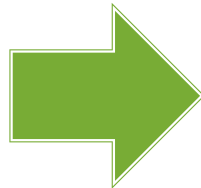


Extract feature
vector from
play history



$(a_1, a_2, a_3, \dots, a_n)$

Match via
machine
learning



Procedural Content

Challenge



Parameterize
challenge
difficulty



$(b_1, b_2, b_3, \dots, b_k)$

Adaptive Difficulty

- Manually define the **gameplay model**
 - Metrics that identify player behavior
 - Parameters that define challenge behavior
 - Also metrics to evaluate player success or failure
- **Goal:** Use learning to find player-challenge match-up
 - Use playtesting/beta to get a large training set
 - Create an initial model from these results
 - Adjust in the game according to current player
- Starting to really take off in the industry

Summary

- Procedural content started with Rogue(likes)
 - Tightly coupled with permadeath, horizontal design
 - Becoming fashionable once again
- Many applications to modern game design
 - World Generation
 - Puzzle Generation
 - Story Generation
 - Dynamic Challenges

Summary

- Procedural content started with Rogue(likes)
 - Tightly coupled with permadeath, horizontal design
 - Becoming fashionable once again
- Many
 - World
 - Puzz
 - Story Generation
 - Dynamic Challenges

Procedural Content Wiki:
<http://pcg.wikidot.com>