# **Procedural Generation Beyond Visuals: Dynamic Economic Gameplay and Systemic Outbreak Scenarios**

## **I. Advancing Gameplay Dynamics: Procedural Generation Beyond Visuals**

Procedural generation (PG) has become an increasingly influential force in game development, referring to the algorithmic creation of game content rather than manual design.1 This approach utilizes mathematical formulas and algorithms to dynamically produce a wide array of game elements, from environments and characters to levels and narratives.1

### **A. The Evolution of Procedural Generation in Gaming**

The initial applications of procedural generation in the gaming industry were predominantly focused on visual elements. Developers employed PG techniques to generate vast game worlds, intricate terrains, diverse textures, and complex level structures.1 This focus was largely driven by the need for efficiency in creating expansive digital landscapes and reducing the manual labor involved in populating them.2 Seminal titles such as *Minecraft* and *No Man's Sky* brought this aspect of PG into the mainstream, showcasing its power to create seemingly endless and varied environments for players to explore.2 These games demonstrated the capability of algorithms to produce a large volume of visual assets, a task that would be prohibitively time-consuming and expensive if done manually.

However, the application of procedural generation is progressively transitioning beyond these visual domains. There is a growing trend towards utilizing PG for non-visual elements, including the dynamic generation of game content such as characters, narratives, quests, game mechanics, and interactive events.1 This shift marks a significant maturation of PG techniques and reflects a desire within the industry for deeper, more systemic, and endlessly replayable gameplay experiences. Academic research and industry development are increasingly exploring areas like procedural narrative generation, as seen in systems using GPT-based agents to enrich player experiences with personalized stories 8, and dynamic quest generation systems that adapt to player actions and world states.9 Furthermore, procedural content generation (PCG) is now being applied to the creation of core game mechanics and rules, fundamentally altering how games can be designed and experienced.6 For instance, procedural narrative techniques leverage grammar-based generation and Markov chains to create evolving storylines and dialogues.5

The trajectory of procedural generation, from its initial visual applications to its current exploration of non-visual elements, indicates a broader shift in game development. This evolution reflects a move towards more systemic design philosophies and an emphasis on player-driven narratives. Early PG methodologies primarily addressed the challenge of content quantity, particularly for visual assets.2 The current focus on non-visual aspects—such as quests, economic simulations, and dynamic event systems 1—targets the creation of varied and profound gameplay experiences. This suggests that developers are increasingly looking to PG not just to build game worlds, but to populate those worlds with dynamic systems that foster unique interactions and stories, moving away from static, pre-scripted narratives towards experiences that are more personalized and unpredictable.8

### **B. The Imperative of Non-Visual PG: Replayability, Scalability, and Emergence**

The drive towards non-visual procedural generation is underpinned by several key advantages that directly impact player experience and development efficiency.

* **Enhanced Replayability:** A primary benefit of PG is its capacity to create unique experiences with each playthrough. By algorithmically generating content, games can offer novel challenges, scenarios, and narratives every time a player engages with them, significantly extending the game's lifespan and appeal.2 This is especially true for non-visual elements like quests or dynamic events, as these directly influence gameplay variability and ensure that returning players encounter fresh content.
* **Content Scalability and Efficiency:** Procedural generation empowers developers to create vast amounts of content algorithmically, which drastically reduces the manual workload and associated development costs.2 This efficiency is particularly crucial for large-scale games or titles that require ongoing content updates, such as live service games.18 The economic benefits are substantial; for example, a single manually crafted quest might take a week to develop and test, whereas PG can generate numerous quest variations in a fraction of that time.17
* **Emergent Experiences:** By generating systems and scenarios rather than fixed, pre-determined content, PG can lead to emergent gameplay. Emergence occurs when unexpected and novel interactions, behaviors, or narratives arise from the interplay of various procedurally generated elements and player actions.11 This capacity for surprise and unpredictability is a key driver for deeper player engagement and can lead to unique stories that feel personal to each player. Games like *Dwarf Fortress* are renowned for their ability to generate detailed world histories and interconnected events, fostering complex emergent narratives.19
* **Resource Conservation:** An often-overlooked benefit of PG is its potential for resource conservation. Because content can be generated algorithmically at runtime, the storage requirements for game assets can be significantly reduced.2 This not only saves disk space but can also lead to faster load times and improved performance, making games more accessible across a wider range of hardware.

The increasing complexity of generating believable narratives, balanced economies, or coherent dynamic events necessitates more sophisticated algorithmic approaches and often requires interdisciplinary expertise. Creating a compelling procedural narrative, for instance, involves challenges such as maintaining emotional depth and avoiding repetitiveness.8 Similarly, generating a stable and engaging game economy requires careful balancing of numerous interconnected variables.4 This has led to the adoption and exploration of advanced AI techniques, including Large Language Models (LLMs) for narrative and dialogue 6, agent-based modeling for simulating complex systems like economies 17, and constraint satisfaction programming for ensuring coherence and adherence to design rules.20 The need for "guardrails" in procedural narrative generation to maintain focus and appropriateness 8, or for systems to ensure "balance" in procedurally generated economies 4, highlights the necessity of combining raw algorithmic power with careful design intent and systems thinking.

Despite the immense benefits, a significant challenge in non-visual PG is ensuring "meaningfulness." There is a risk that procedural variety can mask a lack of genuine consequence or depth, leading to what has been termed an "algorithmic alibi" 19—where impressive complexity deflects scrutiny from a lack of meaningful player impact. Issues such as a lack of emotional depth in narratives or repetitive gameplay loops can diminish the player experience if the procedural systems are not carefully designed.8 Therefore, successful non-visual PG must focus not merely on the *generation* of content, but on creating systems that *react* meaningfully to player choices and the evolving state of the game world, thereby fostering a genuine sense of agency.

## **II. Procedural Generation for Economic Gameplay: Dynamic Contracts and Missions**

The application of procedural generation to economic gameplay elements, particularly the creation of dynamic contracts and missions, offers a pathway to more reactive and engaging game economies. Instead of static mission boards or fixed trade opportunities, PG can enable systems where tasks related to selling products, resource acquisition, and other economic activities emerge dynamically from the game world's state.

### **A. Core Principles: Generating Missions for Selling Products and Economic Activities**

Procedural economic missions are tasks generated algorithmically that involve economic activities such as resource acquisition, production, trade, or the delivery of goods and services.11 The core request is to explore "contract/mission generation for selling products," which falls squarely within this domain.

Several key parameters must be considered for the procedural generation of such missions:

* **Objectives:** These define what the player needs to accomplish. Examples include delivering a specific quantity of a product to a designated location, acquiring a particular resource, establishing a new trade route, or protecting a valuable shipment. Complex objectives can often be broken down into smaller, manageable steps or sub-quests. The STRIPS (Stanford Research Institute Problem Solver) formalism, for instance, can define quest events like 'go', 'get', and 'use', which can be combined to form more elaborate economic tasks.16
* **Targets/Destinations:** These are the NPCs, locations, or entities involved in the contract. Their selection can be driven by the current world state, existing faction relationships, or specific economic needs within the simulation.23 For example, a mission to deliver goods might target a city experiencing a procedurally generated shortage.
* **Conditions/Constraints:** These encompass various factors that shape the mission, such as time limits for completion, required items or skills (e.g., a specific vehicle for transport, negotiation skills for better terms), and risk factors like dangerous travel routes or potential interference from competitors. The mission system in *Anarchy Online*, for example, features sliders like "EASY <-> HARD" which influence mob difficulty and "OPEN <-> HIDDEN" affecting level layout and the presence of obstacles like locked doors.24
* **Rewards:** Successful completion of economic missions typically results in rewards such as monetary payment, valuable items, reputation changes with factions or individuals, or access to new markets, resources, or more lucrative contracts. *Anarchy Online*'s system includes a "MONEY <-> XP" slider, allowing players to influence the type of reward they receive.24

Effective procedural economic mission generation relies heavily on integration with underlying simulated systems:

* **Market Conditions:** The availability of missions, the pricing of goods involved, and the demand for specific services can be dynamically tied to a simulated economy's supply and demand mechanics.4 For instance, if a particular region is experiencing a procedurally generated famine, missions to deliver food to that region might become more frequent and offer higher rewards. Agent-based economic models can simulate these dynamics, with prices and agent roles adapting to changing conditions.17
* **Player Progression:** The difficulty, complexity, and potential rewards of generated missions can be scaled according to the player's level, skills, reputation with relevant factions, or current economic standing.9 Systems like the Panoptyk Engine aim to generate quests based on real-time player information and their previous actions within the game world.9
* **Resource Availability:** Missions can be generated to address shortages of essential resources or to capitalize on surpluses within the game world.4 Rule-based systems can be employed to determine resource spawn rates and locations, contributing to a balanced and dynamic economy that can then feed into mission generation.4

### **B. Techniques and Algorithms for Economic Mission Generation**

Several techniques and algorithmic approaches can be employed to generate dynamic economic missions, each with its own strengths and weaknesses.

**Table 1: Comparative Overview of PG Techniques for Dynamic Economic Missions/Contracts**

| **Technique** | **Core Mechanism** | **Primary Application in Economic Missions** | **Strengths** | **Weaknesses** | **Key Snippet Examples** |
| --- | --- | --- | --- | --- | --- |
| **Rule-Based Systems** | Defines explicit rules and parameters (e.g., IF-THEN logic) to guide content creation. | Basic contract logic, resource distribution rules, simple economic event triggers. | Control, predictability, easier to implement for simpler scenarios. | Can become repetitive, scalability issues for highly complex economies. | 4 |
| **Agent-Based Modeling (ABM)** | Simulates individual agents (NPCs, factions) with economic goals, production/consumption needs; missions emerge from agent interactions. | Dynamic supply/demand chains, emergent trade routes, missions based on agent needs. | High potential for emergent complexity, realistic economic behavior. | Computationally intensive, balancing can be challenging, potential for un-fun emergent scenarios. | 17 |
| **Large Language Models (LLMs)** | Utilizes AI trained on vast text data to generate natural language for descriptions, dialogue, and potentially contract terms. | Generating contract text, NPC dialogue for trade, dynamic mission briefings, adapting terms based on data. | Highly varied and nuanced textual content, potential for personalized missions. | Can "hallucinate" or produce illogical content if unconstrained, ethical concerns, context limitations. | 8 |
| **Constraint Satisfaction (CSP) / Answer Set Programming (ASP)** | Defines missions as variables and constraints; a solver finds assignments satisfying all rules. | Generating complex mission parameters adhering to balance/narrative rules, ensuring logical consistency. | Guarantees valid/coherent parameters, good for complex interdependencies. | Defining constraints can be complex, solver performance issues for large spaces or real-time. | 20 |

1. Rule-Based Systems:

Rule-based systems operate by defining explicit parameters and logical rules that guide the creation of contracts and the broader economic environment.4 A simple rule might be: "IF region\_A\_demand\_for\_tools > supply\_of\_tools THEN generate\_mission(objective=deliver\_tools\_to\_region\_A, reward\_modifier=high)". These systems are often used to determine resource distribution patterns, establish the fundamental logic for basic contracts (e.g., "deliver X quantity of item Y to location Z for reward A"), and trigger simple economic events within the game world.4 One of their primary applications is in ensuring fair resource distribution within a procedurally generated economy, preventing imbalances that could negatively impact gameplay.4

The main strengths of rule-based systems lie in the level of control they offer designers and their relative predictability, making them easier to implement and debug for simpler economic scenarios. However, if the rule sets are not sufficiently complex or dynamic, the generated missions can quickly become repetitive. Furthermore, scaling rule-based systems to manage highly complex and interconnected economies can become unwieldy.

2. Agent-Based Economic Modeling:

Agent-based modeling (ABM) takes a bottom-up approach by simulating individual agents—such as NPCs, corporations, or factions—each with its own economic goals, production capabilities, resource needs, and behavioral patterns.17 Contracts and missions then emerge organically from the interactions, needs, and conflicts of these autonomous agents. For example, an NPC blacksmith agent whose simulated inventory of iron ore is running low might autonomously generate a "buy iron ore" contract or a mission to secure a new mining claim.

ABM excels at creating dynamic supply and demand chains, fostering the emergence of complex trade routes, and generating missions that are intrinsically tied to the simulated needs of the game world's inhabitants.17 This approach has a high potential for generating emergent complexity and more realistic, adaptive economic behavior. The primary challenges include the potential computational cost of simulating many agents in detail, the difficulty in balancing such a dynamic economy, and the risk that emergent behavior might lead to undesirable or un-fun gameplay scenarios if the system is not carefully designed and constrained.

3. Large Language Models (LLMs) and AI:

Large Language Models are increasingly being explored for their potential in generating the qualitative aspects of economic missions.8 LLMs can produce natural language text for contract descriptions, mission briefings, and dynamic NPC dialogues related to trade negotiations. They might even be used to generate the specific terms of a contract by processing structured input or natural language requests.25 Beyond text generation, broader AI techniques can analyze simulated economic data to suggest optimal mission parameters, identify emerging economic opportunities that could form the basis of new missions, or adapt existing mission templates to current market conditions.29 For instance, an AI system could monitor player trading patterns and generate personalized missions that align with their established economic activities or offer them challenges in new market sectors. 30 describes AI generating market reactions like social media posts based on simulation data, which could be adapted to generate mission-related intelligence or rumors.

The strength of LLMs lies in their ability to produce highly varied, nuanced, and contextually relevant textual content, which can significantly enhance the immersion and perceived uniqueness of economic missions.25 However, LLMs are prone to "hallucinations" (generating incorrect or nonsensical information) if not properly prompted and constrained.8 Ensuring the logical consistency of LLM-generated contract terms with underlying game mechanics is a critical challenge. Other concerns include the ethical implications of LLM-generated content and potential biases 7, as well as technical limitations like context window size and the difficulty of integrating LLMs with real-time external data sources without specialized frameworks.31

4. Constraint Satisfaction Programming (CSP) / Answer Set Programming (ASP):

CSP and ASP are declarative programming paradigms where a problem is defined as a set of variables, each with a domain of possible values, and a set of constraints that specify the allowed relationships between these variables.20 A solver then searches for assignments of values to variables that satisfy all constraints. In the context of economic missions, variables could represent item type, quantity, source, destination, reward amount, risk level, and required skills. Constraints could ensure, for example, that the reward is proportional to risk and distance, that the required item is actually available at the source location, or that the destination has a genuine need for the item.

These techniques are particularly useful for generating complex mission parameters that must adhere to specific game balance rules or narrative requirements.20 ASP, for instance, allows for the explicit description of a "design space," giving designers fine-grained control over the properties of the generated missions.34

The main advantage of CSP/ASP is the guarantee that generated missions will be valid and coherent according to the defined constraints, making them well-suited for managing complex interdependencies. However, formulating the constraints for a complex mission system can be a challenging task in itself. Solver performance can also be a concern for very large search spaces or if real-time mission generation is required, necessitating careful optimization. Introducing sufficient randomness to ensure variety while satisfying all constraints also requires deliberate design.20

A synthesis of these approaches often yields the most robust results. For example, rule-based or CSP/ASP frameworks can provide the logical structure and ensure balance for economic mission parameters. Agent-based systems can then introduce dynamic triggers for these missions based on simulated supply, demand, and agent needs. Finally, LLMs can overlay rich narrative flavoring, generate engaging contract text, and facilitate dynamic NPC interactions related to these missions. This layered approach leverages the strengths of each technique while mitigating some of their individual weaknesses, moving towards a system where economic missions are not just algorithmically generated but are also deeply integrated into a living, reactive game world.

### **C. Case Studies and Examples in Economic Mission Generation**

Several existing games and research projects illustrate various approaches to procedural economic mission generation:

* **1. Anarchy Online:** This MMORPG features a mission system where players interact with terminals to request missions. They can adjust parameters such as difficulty (EASY <-> HARD), alignment (GOOD <-> BAD), mob type (ORDER <-> CHAOS), and reward type (MONEY <-> XP).24 These settings directly influence the generated mission's mobs, layout, and objectives. The "MONEY <-> XP" slider is a direct example of player-influenced economic incentive in procedural mission generation. The difficulty slider impacts the quality level (QL) of item rewards, linking mission challenge to economic gain.24 This system demonstrates early principles of player-driven parameterization for tailoring procedurally generated tasks.
* **2. RimWorld:** This colony simulation game procedurally generates a variety of quests, some of which have strong economic implications. "Trade Request" quests involve a faction asking the player's colony to supply specific items (e.g., clothing, food, components) in return for rewards like silver, rare resources, or goodwill.37 These requests specify item types, quantities, quality requirements, and delivery deadlines. Other quests, such as "Item Stash" or "Bandit Camp," often yield valuable resources or items upon completion, which directly feed into the colony's economy and trade potential. This system showcases procedurally generated objectives tied to resource needs and inter-faction economic relationships.
* **3. Panoptyk Engine (Conceptual Research):** The Panoptyk Engine is designed around information-driven gameplay, where quests are generated by NPC faction leaders based on their current knowledge, pre-defined goals, and real-time events occurring in the game world.9 For example, the Thieves Guild leader might generate an "Item Retrieval Quest" to loot valuable items it has learned about, or a "Gift Delivery Quest" to reward an agent who has been helpful.38 This illustrates how a dynamic flow of information—such as the discovery of a valuable artifact or the actions of other agents—can trigger procedurally generated economic missions that are contextually relevant to the ongoing state of the simulation.
* **4. Dwarf Fortress (Emergent Economic Objectives):** While *Dwarf Fortress* may not have a formal "contract generation" system in the traditional sense, its deep simulation of resources, production chains, and the needs of various entities (dwarves, caravans, civilizations) leads to emergent economic objectives.19 The arrival of a trade caravan creates an opportunity (and often a necessity) to exchange goods. The creation of valuable artifacts within the player's fortress can trigger "quests" from external entities (villains, other civilizations) seeking to acquire them, either through negotiation, demand, or theft.42 Furthermore, the planned systems for bounties on stolen items represent a form of procedurally generated contract ("recover this stolen item for a reward").41 This demonstrates how a highly detailed procedural world can naturally generate economic pressures and opportunities that function as implicit missions for the player, driven by the simulated needs and desires of the world's inhabitants.

These examples highlight a spectrum of approaches, from explicit player-parameterized mission generation to more emergent systems where economic tasks arise from the complex interactions within a simulated world. Procedurally generated economic missions can thus serve as more than just tasks for the player; they can become a powerful tool for implicit world-building and storytelling. The types of contracts available, the goods in demand, the factions offering them, and the associated risks can reveal much about the state of the game world's economy, inter-faction relationships, and societal needs, all conveyed dynamically through the tasks offered to the player. This makes the mission system itself a dynamic narrator of the game world's evolving state, enhancing immersion and player engagement.

### **D. Challenges in Procedural Economic Mission Generation**

Despite the potential, several challenges must be addressed when designing systems for procedural economic mission generation:

* **Balancing and Fairness:** A critical challenge is ensuring that generated contracts are balanced in terms of risk versus reward and that they are fair to the player. The system must avoid creating exploitable economic loopholes or missions that are impossible to complete profitably.4 This requires careful calibration of generation parameters and potentially dynamic adjustment based on the evolving economy.
* **Meaningful Choice and Impact:** Procedurally generated economic missions should offer meaningful choices to the player and have a tangible impact on their economic standing or the game world.45 They need to transcend the feeling of generic "fetch quests" or "delivery tasks" by incorporating elements of strategy, risk assessment, and consequence.8
* **Avoiding Repetitiveness:** Like all PG content, economic missions can become repetitive if the underlying generation system lacks sufficient variety in parameters, objectives, narrative flavor, and potential complications.8 Techniques like LLMs for varied descriptions 27 or more complex multi-stage generation logic 16 can help mitigate this.
* **Narrative Coherence:** Integrating procedurally generated economic missions into an overarching narrative or a believable world context is essential.8 Players are more likely to engage with missions that feel connected to the world's story, factions, and ongoing events. Why is this particular contract available now? Who benefits from its completion? What are the broader implications?
* **Scalability of Complexity:** As the economic models become more sophisticated and mission parameters more numerous and interconnected, ensuring that the procedural generation system remains manageable, performant (especially if generating missions in real-time), and relatively bug-free becomes increasingly difficult.22 The risk of generating unplayable or broken missions due to unforeseen interactions between parameters is a constant concern.

A key aspect of overcoming these challenges lies in tightly integrating player progression and reputation into the procedural contract generation system. This creates a sense of a living, reactive economy rather than a static mission board. While basic systems might scale mission difficulty with player level, as seen in *Anarchy Online* 24, more advanced implementations could unlock new types of contracts, offer better terms, or trigger unique economic opportunities as a player's reputation with a merchant guild, corporation, or faction evolves. Conversely, negative actions could lead to blacklisting or rival entities offering counter-missions. Such reactivity ensures that the player's economic actions have lasting and meaningful consequences, fostering a more engaging long-term gameplay loop and directly addressing the challenge of providing meaningful choice. However, a fundamental hurdle remains: determining what constitutes a "good" or "interesting" economic mission from the player's perspective, beyond simple profitability. While algorithms can be tuned for economic balance 4, player enjoyment is also influenced by novelty, appropriate challenge, narrative context, and perceived fairness. The issues of "lack of emotional depth" and "repetitiveness" noted in procedural narrative 8 apply equally to economic missions. This suggests a need for more sophisticated player modeling, dynamic feedback mechanisms to guide the generation process, or for designers to meticulously craft the constraints and heuristics that define "interesting" and "engaging" economic gameplay.

## **III. Procedural Generation of Pestilence: Crafting Unique Outbreak Scenarios**

Beyond economic systems, procedural generation is also being explored for creating unique and dynamic pest or disease outbreak scenarios in games. This involves algorithmically defining the emergence, characteristics, spread, and impact of these biological threats, offering rich possibilities for systemic gameplay and emergent challenges.

### **A. Core Principles: Simulating Emergence, Spread, and Impact of Pests/Diseases**

Procedural outbreak generation involves the algorithmic creation of disease or pest events, encompassing their origin, specific characteristics, mechanisms of transmission, and their effects on the game world and its inhabitants.12

Key characteristics for the procedural generation of such scenarios include:

* **Pathogen/Pest Type:** The nature of the threat—be it viral, bacterial, fungal, insectoid, parasitic, or even a fictional blight—fundamentally dictates its behavior, vulnerabilities, and potential countermeasures.12 For example, *Outbreak Simulator* includes pre-parameterized scenarios for influenza and SARS-CoV-2, with plans for Ebola and Measles 12, while *RimWorld* features insectoid infestations with specific creature types like Megascarabs and Spelopedes.49
* **Transmission Vectors:** How the pestilence spreads is crucial. Common vectors include airborne transmission, direct contact, vector-borne transmission (e.g., carried by insects or animals), or environmental contamination (e.g., through water sources or infected surfaces).12 The chosen vector heavily influences spread patterns and effective containment strategies.
* **Symptoms & Severity:** The effects on afflicted entities are defined by parameters such as incubation period, infectivity rate (how easily it spreads), lethality rate, specific debuffs (e.g., reduced work speed, combat effectiveness), and potential long-term consequences.12 Compartmental models often categorize populations into states like Susceptible, Exposed, Infected, Recovered, and Deceased (S, E, I, R, D) to track these effects.12 *Covid Simulator*, for instance, models outcomes like illness, long-term disability, and death.47
* **Environmental Triggers/Factors:** The emergence and spread of outbreaks can be tied to environmental conditions. Factors like temperature, humidity, population density, sanitation levels, or the presence of specific resources (e.g., stagnant water for mosquito-borne diseases, overhead mountain terrain for *RimWorld*'s insect infestations) can influence outbreak likelihood and severity.49 Research into forest pest susceptibility utilizes environmental covariates such as land surface temperature, precipitation, and vegetation indices (NDVI, NDMI) to model and predict outbreak locations.52

Modeling population dynamics and environmental interaction is also central:

* **Population Effects:** The outbreak's impact can vary across different population groups based on factors like age, species, pre-existing health conditions, or even genetic predispositions. *Covid Simulator* notes that its simulation bases outcomes on US CDC statistics which consider age, gender, health, and ethnicity.47
* **Environmental Alteration:** The outbreak itself can alter the game environment. This could manifest as contaminated zones requiring cleansing, destruction of crops leading to famine, or even the creation of new environmental threats, such as the insectoid hives that emerge and expand during infestations in *RimWorld*.49

### **B. Techniques and Algorithms for Outbreak Scenario Generation**

A variety of techniques can be employed to procedurally generate and simulate pestilence scenarios.

**Table 2: Comparative Overview of PG Techniques for Pest/Disease Outbreak Scenarios**

| **Technique/Model** | **Core Mechanism** | **Primary Application in Outbreaks** | **Strengths** | **Weaknesses** | **Key Snippet Examples** |
| --- | --- | --- | --- | --- | --- |
| **Compartmental Models (e.g., SIR, SEIR)** | Divides population into states (Susceptible, Exposed, Infected, Recovered, etc.); models flow between states using differential equations or discrete time steps. | Simulating overall epidemic curve, infection rates, impact of interventions at a macro level. | Epidemiologically sound for many diseases, good abstraction, computationally manageable for large populations. | Assumes homogeneous mixing, may need adaptation for complex social structures or pest dynamics. | 12 |
| **Agent-Based Modeling (ABM)** | Simulates individual agents with health states and behaviors; spread occurs via agent-to-agent or agent-to-environment interactions. | Modeling localized outbreaks, complex transmission paths, impact of individual agent behavior, spatial dynamics. | Captures population heterogeneity and complex interactions, good for emergent behaviors. | Computationally expensive for many agents, requires detailed behavior modeling. | 47 (implicit) |
| **AI-driven Prediction (RF, GB, MaxEnt, Regression)** | Uses machine learning/statistical models trained on environmental/historical data to predict outbreak likelihood, susceptibility, or initial locations. | Determining initial outbreak locations, predicting spatial spread based on risk factors, dynamic parameter adjustment. | Data-driven realism, contextually relevant outbreak starts, adaptive potential. | Requires significant training data, models can be black boxes, predictions are probabilistic. | 51 |
| **Rule-Based Systems / Cellular Automata (CA)** | Defines rules for origin, spread, and impact based on game state (Rule-Based). Grid cells change state based on neighbors and rules (CA). | Simulating spatial spread on maps, growth of infested areas, environmental changes due to outbreak, specific pest/disease behaviors. | Simple to implement, computationally efficient for grid spread, can create complex emergent patterns. | Can look artificial if rules are too simple, may not easily model complex agent behaviors or long-range transmission. | 4 |

1. Compartmental Models (e.g., SIR, SEIR):

These models are a mainstay in epidemiological research and have been adapted for game simulations. They divide the game's population into distinct compartments—such as Susceptible (S), Exposed (E), Infected (I), Recovered/Resistant (R), and sometimes Deceased (D)—and use systems of differential equations or discrete time-step calculations to model the rate at which individuals transition between these states.12 Outbreak Simulator, for example, employs an S, E, I, V (Vaccinated), R, D model 12, while the game SARS Wars uses an adapted SEIR model to simulate its pandemic scenario.48

Compartmental models are effective for simulating the overall progression of an epidemic, often visualized as an "epidemic curve," and for estimating key metrics like infection rates and the potential impact of large-scale interventions (e.g., mass vaccination).12 Their strengths lie in their well-established theoretical basis in epidemiology, their ability to provide a good abstraction of disease spread dynamics at a population level, and their computational manageability even for large populations. However, a primary limitation is the common assumption of homogeneous mixing within the population (i.e., every individual has an equal chance of interacting with every other individual), which may not accurately reflect the complex spatial layouts, social structures, or movement patterns found in many game worlds.12 These models may also require significant adaptation to accurately represent the dynamics of pest infestations, which can differ substantially from infectious diseases.

2. Agent-Based Modeling (ABM):

In contrast to the top-down approach of compartmental models, ABM simulates outbreaks from the bottom up. Each individual agent (e.g., a game character, a creature, a plant) is modeled with its own health state, behaviors (such as movement patterns, social interactions, hygiene practices), and susceptibility to the pestilence. The disease or pest then spreads through direct agent-to-agent interactions or via agents interacting with a contaminated environment. Games like RimWorld, with its detailed simulation of individual colonists and animals, implicitly use agent-based principles for events like disease spread 49, and Covid Simulator models individual workers who can become infected and spread the virus within a workplace.47

ABM is particularly well-suited for modeling localized outbreaks, tracing complex and non-obvious transmission paths that depend on specific agent behaviors and social networks, and observing the impact of individual agent actions (or inactions) on the overall spread. This approach excels at capturing population heterogeneity and complex interaction patterns, making it ideal for simulating detailed spatial dynamics and allowing for rich emergent outbreak behaviors. The main drawbacks are its computational expense, which can become prohibitive for simulations involving very large numbers of agents, and the need for detailed behavior modeling for each agent type to achieve realistic outcomes.

3. AI and Statistical Modeling for Prediction and Spatial Dynamics:

Artificial intelligence and statistical modeling techniques can be employed to predict where and when outbreaks are most likely to occur, and to simulate their spatial dynamics. Machine learning algorithms such as Random Forest (RF), Gradient Boosting (GB), and Maximum Entropy (MaxEnt), along with statistical models like logistic regression, can be trained on historical outbreak data and relevant environmental covariates (e.g., temperature, humidity, vegetation density, population movements) to identify areas of high susceptibility.51

These models can be used to procedurally determine initial outbreak locations by identifying regions with the highest predicted risk based on current environmental conditions.52 For example, research on forest pest susceptibility in Honduras successfully used RF, GB, and MaxEnt algorithms with environmental covariates like land surface temperature, precipitation, and NDVI to create spatial susceptibility maps with high accuracy (up to 92% for RF and GB).52 Once an outbreak begins, AI can also inform models of spatial spread, using diffusion models, gravitation models that account for population movement between zones 12, or learned patterns from past outbreaks. Furthermore, these systems can dynamically adjust outbreak parameters (e.g., infectivity, spread rate) in response to procedurally generated changes in environmental conditions.

The strengths of these AI-driven approaches include their potential to create more realistic and contextually relevant outbreak starting points and their capacity for dynamic adaptation. However, they typically require substantial amounts of data for training (e.g., historical records of outbreak locations, detailed environmental data layers). The models themselves can sometimes be "black boxes," making it difficult to understand the exact reasoning behind their predictions, and their outputs are inherently probabilistic rather than deterministic.

**4. Rule-Based Systems and Cellular Automata:**

* **Rule-Based Systems:** These involve defining explicit rules that govern how a disease or pest originates, spreads, and impacts entities based on the current game state. For example: "IF entity\_A\_is\_infected AND entity\_B\_is\_adjacent\_to\_entity\_A AND entity\_B\_is\_susceptible THEN entity\_B\_infection\_chance = X%". The infestation mechanics in *RimWorld* are heavily rule-based, with factors like temperature thresholds and the presence of "overhead mountain" terrain dictating where insectoid hives can spawn.49
* **Cellular Automata (CA):** CA model the game world as a grid of cells, where each cell can exist in one of several states (e.g., healthy, infected, infested, immune, barren). The state of each cell evolves over discrete time steps based on the states of its neighboring cells, according to a predefined set of rules.4 This technique is well-suited for simulating the spatial spread of an outbreak across a map, the growth of infested or contaminated areas, and environmental changes resulting from the pestilence. Both rule-based systems and CA are relatively simple to implement and can be computationally efficient, especially for grid-based simulations of spread. They are capable of generating complex emergent patterns from simple, localized rules. However, if the rules are too simplistic, the resulting spread patterns can look artificial or predictable. Standard CA models may also struggle to easily account for complex, non-local agent behaviors or long-range transmission mechanisms without significant modifications or hybridization with other techniques.

5. Algorithmic Determination of Initial Conditions & Dynamic Event Triggers:

Beyond the core spread model, procedural generation can define the initial conditions of an outbreak and trigger dynamic events as it progresses.

* **Initial Conditions:** Algorithms can procedurally set key parameters at the outset of an outbreak. This includes the pathogen's intrinsic virulence or the pest's reproductive rate, the initial number and location of infected individuals or infestation sites 52, and the initial availability of resources or knowledge for combating the threat. Statistical distributions, such as the negative binomial distribution, can be used to simulate baseline daily case counts in surveillance data, with outbreak signals (e.g., single-day spikes or multi-day increases with varying duration, peak day, and peak size) superimposed to initiate the event.51
* **Dynamic Event Triggers:** During an ongoing outbreak, algorithms can trigger new events based on its progression, specific player actions (or inactions), or elapsed time. Examples include the pathogen mutating to become more virulent or drug-resistant, the discovery of a new critical symptom, the arrival of external aid, or the emergence of secondary threats (e.g., other opportunistic infections, social unrest due to resource scarcity). The quest system in *RimWorld*, for instance, often incorporates timers and specific threat triggers that activate under certain conditions.37

The most robust procedural outbreak systems often emerge from a blend of these techniques. For instance, statistical or AI models might determine *where* and *when* an outbreak is most likely to initiate.51 Compartmental or agent-based models can then simulate its *spread* and impact on the population.12 Rule-based systems can govern specific local interactions, environmental effects, and the mechanics of player interventions.49 This layered approach allows each technique to contribute its strengths, leading to more comprehensive and dynamic outbreak scenarios.

### **C. Integrating Player Agency and Intervention in Outbreak Scenarios**

A crucial aspect of procedurally generated outbreak scenarios is providing players with meaningful agency to influence the course of the pestilence. This involves designing game mechanics that allow for various forms of intervention and ensuring the simulation reacts dynamically to player choices.

**Designing Game Mechanics for Management:**

* **Research & Development:** Players might need to research the specific pathogen or pest to understand its characteristics, vulnerabilities, and transmission pathways, leading to the development of countermeasures such as vaccines, antidotes, pesticides, or specialized containment equipment.50
* **Containment & Control:** This can involve a range of actions, from implementing quarantines and lockdowns for infected areas or individuals, to culling infected populations (in the case of animals or pests), to cleaning and decontaminating affected environments.12 *Covid Simulator* allows players to directly enact mask and vaccine mandates 47, while *RimWorld* players employ strategies like temperature control and chokepoint defenses against insectoid infestations.49
* **Resource Management:** Combating an outbreak invariably involves managing limited resources, such as medical supplies, research personnel, financial budgets, and specialized equipment.56 Players must make strategic decisions about how to allocate these resources for maximum effect. Studies using experimental gaming simulations have explored biosecurity investment decisions in agricultural contexts, where participants allocate funds to upgrade facility biosecurity levels.57
* **Public Health Policies/Communication (for human diseases):** In scenarios involving human populations, players might make decisions regarding public information campaigns, the implementation of mandatory health measures, and managing population morale, trust, and compliance with these policies.46 The game *SARS Wars*, for example, includes a "motivation gauge" that is affected by player decisions, representing the population's willingness to adhere to implemented health measures.48

**Procedurally Generating Consequences of Player Actions:**

The game's procedural systems must dynamically reflect the consequences of player interventions. The success or failure of these actions should tangibly alter the outbreak's progression. For example, an effectively implemented quarantine should visibly slow the spread rate in the simulation, while failed research into a cure might lead to a worsening situation or the emergence of new, more dangerous strains.

Furthermore, the system can generate unintended consequences. Overuse of broad-spectrum pesticides might lead to the evolution of resistant pests or cause collateral damage to beneficial organisms and the wider ecosystem. Excessively strict or prolonged lockdowns could trigger economic collapse, resource shortages, or social unrest, creating secondary crises that the player must manage. The underlying simulation models (compartmental, agent-based, etc.) should be designed to react to these player-driven interventions by adjusting their parameters and behavior accordingly. In SARS Wars, player decisions directly apply multipliers to the virus's basic reproduction number (R0​) and modify the population's motivation gauge, demonstrating a direct feedback loop between player choice and systemic outcomes.48

This interaction between player agency and a reactive procedural outbreak system is critical. It moves beyond simply presenting a problem ("here is a disease") to creating a dynamic decision-making space where players must grapple with complex trade-offs, uncertain outcomes, and cascading consequences, mirroring the complexities of real-world crisis management. The "narrative" of such a procedurally generated outbreak then emerges not from a pre-scripted plot, but from the unique interplay of the pathogen's generated characteristics, the environment's inherent vulnerabilities, and the player's (often imperfect and evolving) attempts to understand and manage the crisis. This creates fertile ground for unique player stories of struggle, adaptation, unexpected failure, and hard-won triumph.

### **D. Case Studies and Examples in Pestilence PG**

Several games and research projects provide insights into the procedural generation and simulation of pestilence:

* **1. Outbreak Simulator / Symphony / Sonata:** These research-oriented simulators use compartmental models (SEIR variants), incorporate spatial data layers and movement models (diffusion, gravitation) to simulate infectious disease spread.12 They allow users to intervene by adjusting parameters like disease biology and virulence, and by implementing public health measures such as travel restrictions, lockdowns, masking, and vaccination campaigns, with outcomes visualized through dynamic data displays.12 These tools emphasize sophisticated simulation for public health planning and education.
* **2. Covid Simulator:** This commercially available game offers a sandbox environment where players can create procedurally generated (or custom-designed) workspaces and experiment with the spread of COVID-19.47 Players can customize viral strains (Alpha, Delta, Omicron, or custom creations) and directly implement or remove interventions like mask mandates and vaccine policies, observing the impact on infection rates among simulated workers. Its focus is on visualizing workplace spread and the direct consequences of different intervention strategies.
* **3. SARS Wars:** This serious game, developed by the University of Liège, models a pandemic using an adapted SEIR model.48 Gameplay revolves around decision-making through card choices, where each player response applies multipliers to the virus's R0​ (basic reproduction number) and affects a "motivational gauge" representing population compliance. This highlights the interplay between epidemiological modeling and the socio-behavioral aspects of managing an outbreak.
* **4. RimWorld ("Infestation" events):** This colony simulation game features procedurally generated insectoid pest outbreaks ("Infestations").49 The spawning of insectoid hives and their defenders is governed by a rule-based system considering factors like the presence of "overhead mountain" terrain, ambient temperature, and light levels. The initial size of an infestation scales with "raid points," linking its severity to the player's game progression. Insect AI dictates their behavior (defending hives, digging, attacking colonists). Players employ a variety of countermeasures, including environmental temperature control (freezing or burning out infestations), creating chokepoints for melee combat, kiting insects in open areas, and building "lure rooms" to attract infestations to heavily trapped areas. This provides a clear example of procedurally generated pest outbreaks deeply integrated into core gameplay, offering diverse strategic challenges and player-driven management solutions.
* **5. "Pest Friends" Board Game / Agricultural Simulations:** "Pest Friends" is an educational board game designed to teach the principles of Integrated Pest Management (IPM).55 Players assume the role of pest managers in a fictitious crop, making decisions about scouting, research, habitat modification, and the use of pesticide or non-pesticide control methods over a series of rounds that simulate a growing season. Other research in agricultural science uses experimental gaming simulations to study biosecurity investment decisions in response to livestock disease outbreaks, often incorporating factors like information uncertainty about infection spread and neighbor behaviors.57 These examples demonstrate the application of outbreak and pest management simulation principles in agricultural and educational contexts, emphasizing strategic decision-making and learning through experiencing consequences.

These case studies illustrate a range of approaches, from detailed epidemiological simulations for research and education to more abstracted but highly engaging gameplay mechanics. A common thread is the potential for procedural outbreak generators to serve a pedagogical function. By allowing players to interact with and manage complex systems—be it disease ecology, integrated pest management, or public health policy—these games can foster a deeper understanding of these phenomena through direct, interactive experience, often proving more engaging than traditional learning methods.12

### **E. Challenges in Procedural Outbreak Generation**

Developing compelling and effective procedural outbreak scenarios presents several distinct challenges:

* **Balancing Realism with Gameplay:** Achieving a balance between scientific realism and engaging gameplay is crucial. Highly realistic simulations can become overly complex, slow-paced, or difficult for players to understand and influence.47 Conversely, overly simplistic models may lack believability or fail to provide meaningful strategic depth. The disclaimer in *Covid Simulator* acknowledges that "It is impossible for a simple game to fully simulate the intricacies of every day human life".47
* **Managing Complexity and Information Overload:** Outbreaks are inherently complex phenomena involving numerous interacting variables (pathogen characteristics, environmental factors, population behaviors, intervention effects). Presenting this information to the player in a clear, understandable, and actionable manner is a significant design challenge.12 Effective data visualization and user interface design are critical.
* **Meaningful Player Feedback and Agency:** Players must be able to understand how their actions and decisions are impacting the course of the outbreak. The game needs to provide clear feedback mechanisms and ensure that player choices feel consequential, rather than the outbreak unfolding independently of their efforts.46
* **Creating Varied and Unpredictable Scenarios:** A core goal of procedural generation is to provide variety. For outbreak scenarios, this means ensuring that generated pestilences offer genuine differences in terms of the challenges they pose and the optimal strategies for managing them. This helps avoid situations where players discover a single dominant strategy that works for all outbreaks, diminishing replayability.
* **Ethical Considerations:** Particularly when simulating real-world diseases or sensitive public health issues, developers must exercise caution. There is a responsibility to avoid spreading misinformation, trivializing serious health crises, or creating content that could be perceived as insensitive or harmful.47 Clear disclaimers and a focus on educational or awareness-raising aspects can be important in such cases.

## **IV. Advanced Methodologies and Future Horizons in Non-Visual PG**

As procedural generation continues to mature beyond visual content, several advanced methodologies are gaining prominence, and new research horizons are emerging. These developments promise to create even more sophisticated, dynamic, and meaningful non-visual gameplay experiences, particularly in the realms of economic interactions and systemic outbreaks.

### **A. The Expanding Role of Large Language Models (LLMs)**

Large Language Models are rapidly transforming the landscape of procedural content generation, offering powerful new capabilities for creating nuanced non-visual elements.

* **Nuanced Narrative and Dialogue:** LLMs excel at generating human-like text, making them exceptionally well-suited for creating diverse, context-aware, and stylistically varied narratives, quest descriptions, NPC dialogue, and in-game lore.6 For economic missions, LLMs can draft detailed contract proposals or generate dynamic dialogue for NPCs negotiating trade terms. In outbreak scenarios, they can produce news reports, diary entries from affected individuals, or scientific briefings that evolve with the crisis. Research has explored using GPT-based agents to foster richer narrative experiences and more dynamic NPC interactions.8 Academic surveys confirm the significant role of LLMs in generating game scenarios, including conversations, stories, and quests.7
* **Complex Rule Set Generation:** Beyond narrative text, LLMs are being investigated for their potential to generate game mechanics and rule sets.6 In the context of economic systems, this could involve LLMs proposing terms for complex contracts, generating dynamic pricing rules based on simulated market data, or even designing the rules for entirely new economic mini-games. For outbreak scenarios, LLMs might generate plausible behaviors for novel pathogens or draft societal response protocols to unique crisis situations.
* **LLM-Based Agents:** A significant emerging area is the development of LLM-based agents capable of human-like decision-making and action within complex game environments.26 These agents, using an LLM as their core reasoning engine, could dynamically create or adapt missions based on the evolving game state, negotiate intricate contract terms with players or other NPCs, or simulate complex, emergent NPC reactions to an unfolding outbreak based on learned knowledge and situational understanding. A conceptual architecture for such agents typically includes modules for memory, reasoning, and input/output processing.26
* **Transforming Instructions to Content:** LLMs can interpret high-level natural language instructions from designers (or potentially even players) and translate them into concrete game content, such as dynamic game scenes, specific NPC interactions, or branching dialogue trees.8 This capability can significantly streamline the design and iteration process for dynamic events and quests.

Despite their immense potential, LLMs present challenges. "Hallucinations" (generating factually incorrect or nonsensical information), maintaining long-term coherence in narratives or rule sets, accurately capturing domain-specific subtleties, and managing context length limitations are ongoing areas of research and development.7 Ethical considerations, such as biases learned from training data or the potential for generating inappropriate content, also require careful attention.7 Techniques like fine-tuning LLMs on domain-specific data, employing sophisticated prompting strategies, and developing hybrid approaches that combine LLMs with traditional symbolic reasoning tools or human-in-the-loop oversight are being explored to mitigate these issues.25

### **B. Leveraging Constraint Solvers (CSP/ASP)**

Constraint Satisfaction Programming (CSP) and Answer Set Programming (ASP) are powerful declarative paradigms for generating content that must adhere to complex sets of rules, interdependencies, and designer-specified constraints.20

* **Ensuring Coherence and Validity:** These techniques are invaluable for ensuring that procedurally generated non-visual elements are logically sound, balanced, and playable.
  + For **economic missions**, CSP/ASP can ensure that generated contracts are internally consistent (e.g., delivery locations are reachable, required goods exist), economically viable (e.g., rewards appropriately reflect risk, distance, and item value), and adhere to global game balance parameters.20
  + For **outbreak parameters**, they can define plausible combinations of pathogen characteristics (e.g., a highly lethal virus is unlikely to also have an extremely long asymptomatic infectious period if it's to spread widely), ensure spread patterns respect environmental constraints, or model intervention effects that follow logical rules.
  + For **dynamic events**, they can generate sequences of occurrences that are narratively coherent and mechanically sound, avoiding contradictions or dead-end states.
* **Explicit Design Space Definition:** ASP, in particular, allows designers to explicitly and declaratively define the "design space" of all possible valid content instances.34 Instead of coding the step-by-step generation procedure, designers specify the properties that valid artifacts must possess. The ASP solver then explores this space to find solutions (i.e., generated content) that meet all specified criteria. This offers a high degree of control over the characteristics and quality of the generated output.
* **Combining with Other Techniques:** CSP/ASP are not mutually exclusive with other PG methods. They can be used to validate or refine content initially drafted by other techniques, such as LLMs or simpler rule-based systems. For example, an LLM might generate the narrative text for a contract, while a CSP ensures the economic parameters are balanced. Alternatively, CSP/ASP could generate a structured skeleton for a mission or event, which is then fleshed out with details by other procedural methods or even hand-authored components.

The primary challenges in using constraint solvers include the complexity of formulating intricate real-world problems into the formal language of variables and constraints. Solver performance can also become a bottleneck for very large or complex design spaces, especially if real-time generation is required, though ongoing research continually improves solver efficiency.20 Introducing meaningful randomness and variety while satisfying a tight set of constraints also requires careful strategic implementation within the solver's search process.20

The convergence of LLMs for their semantic understanding and generative capabilities with CSP/ASP for their logical rigor and constraint enforcement represents a particularly powerful future direction. Imagine an LLM drafting the narrative elements and initial terms of a complex trade agreement, while an ASP solver simultaneously verifies its economic viability, ensures it doesn't conflict with existing game rules or faction relationships, and perhaps even suggests modifications to optimize its balance or interest. This synergistic approach could mitigate the weaknesses of each individual technique—the potential for LLM incoherence or factual errors, and the lack of natural language fluency in pure CSP/ASP—to produce non-visual content that is both highly complex, deeply coherent, and richly nuanced.

### **C. Ensuring Meaningful Emergence and Player Agency**

The ultimate goal of much non-visual PG is to create "meaningful emergence," where complex and engaging gameplay arises from the interaction of systems, and where players feel a strong sense of agency.

* **Reactive Systems:** The foundation of meaningful PG lies in creating systems that are not static but dynamically respond to player actions and changes within the game world.19 This means that generated missions, prevailing economic conditions, or the dynamics of an outbreak should visibly and logically change based on what the player does or fails to do.
* **Consequential Choices:** Player decisions made within these procedurally generated scenarios must have lasting and understandable consequences.10 For example, successfully completing a series of dangerous trade contracts for a struggling faction might visibly improve that faction's economic status and military strength, leading to new geopolitical opportunities or threats. Conversely, failing to contain an outbreak should lead to clear, escalating problems that reflect the player's shortcomings. The Panoptyk Engine's design, for instance, aims for players to experience follow-up consequences from their actions in dynamically generated quests.10
* **Balancing Randomness and Authored Control:** Pure, unconstrained randomness can often feel chaotic, arbitrary, and ultimately meaningless to players. Highly successful PG systems frequently blend procedural elements with hand-crafted "anchors," designer-guided heuristics, or curated content libraries to ensure a baseline level of quality, coherence, and alignment with the overall game vision.19 This provides the "human touch" or design intent that can be lost in purely algorithmic generation.22
* **Avoiding the "Algorithmic Alibi":** As PG systems become more complex, there's a risk that their intricacy can mask a fundamental lack of true interactivity or meaningful consequence.19 The goal should be to create not just variety but also depth and responsiveness, where the player feels their choices genuinely shape their experience and the world around them.
* **Information-Driven Systems:** Approaches like those prototyped in the Panoptyk Engine 9, where quests and events are generated based on the dynamic flow of information and events within the game world itself, represent a strong pathway towards achieving emergent and player-relevant content. In such systems, missions aren't simply "rolled" from a table but arise organically from the simulated needs, actions, and histories of entities within the world. This shifts the focus of PG from "generating a mission" to "simulating a situation that necessitates a mission," making the generated content feel more grounded and purposeful. This philosophy is also evident in the deep simulation ambitions of games like *Dwarf Fortress*, where complex events and objectives emerge from the interactions of its myriad procedural systems.19

This pursuit of meaningful emergence is pushing procedural generation towards the creation of more deeply simulated game worlds. In these worlds, procedural systems are not merely external content generators but are integral components of the world's underlying cause-and-effect fabric, constantly reacting to and being shaped by player actions and other systemic changes.

As non-visual PG becomes more sophisticated, the role of the game designer evolves. Instead of focusing solely on direct content creation, designers increasingly become "meta-designers" or "curators" of these generative systems.13 Their primary task shifts to defining the rules, constraints, goals, evaluation functions, and ethical boundaries that guide the AI and algorithms.8 This requires a higher level of abstraction and a deeper understanding of systemic interactions, as designers must learn how to steer complex systems towards producing desired *types* of experiences rather than hand-crafting specific instances. This also introduces new challenges in testing, balancing, and debugging these inherently unpredictable systems.

Furthermore, the increasing reliance on complex AI, particularly LLMs, for non-visual PG raises significant questions regarding accessibility and ethics for the broader game development community. Access to the most powerful LLMs and the specialized expertise required to effectively fine-tune and integrate them into game engines may be limited, potentially creating a divide between large studios and smaller independent developers.25 Concurrently, issues of bias embedded in LLM training data, the originality and copyright implications of AI-generated content, and the potential for LLMs to produce harmful, undesirable, or simply low-quality output necessitate careful consideration and the development of robust "guardrail" systems and ethical guidelines.7 The "black box" nature of some advanced AI models can also complicate debugging efforts and make it difficult to ensure fairness and transparency in how content is generated and presented to players.

## **V. Conclusion and Strategic Recommendations**

The exploration of procedural generation techniques beyond visual content opens up new frontiers for creating dynamic, emergent, and deeply engaging gameplay. The capacity to algorithmically generate economic missions, contracts, and unique pestilence or disease outbreak scenarios allows for game worlds that are not only vast but also systemically reactive and endlessly replayable.

### **A. Recapitulation of Key PG Techniques for Non-Visual Gameplay**

This report has examined several core methodologies applicable to the procedural generation of non-visual gameplay elements. For **economic missions and contracts**, techniques include:

* **Rule-Based Systems** for foundational logic and parameterization.
* **Agent-Based Modeling** for simulating dynamic economies from which missions emerge.
* **Large Language Models (LLMs)** for generating rich narrative text, contract details, and NPC interactions.
* **Constraint Satisfaction Programming (CSP/ASP)** for ensuring logical coherence, balance, and adherence to complex design rules.

For **pestilence and disease outbreak scenarios**, key approaches involve:

* **Compartmental Models (e.g., SIR/SEIR)** for simulating epidemiological curves at a population level.
* **Agent-Based Modeling** for detailed simulation of individual agent infection and localized spread.
* **AI and Statistical Modeling** for predicting outbreak susceptibility and determining initial conditions based on environmental and historical data.
* **Rule-Based Systems and Cellular Automata** for governing specific pest/disease behaviors and spatial spread patterns.

Across both domains, the trend is a clear shift from PG as a tool for primarily visual asset creation towards its use in crafting systemic, experiential gameplay that responds dynamically to player actions and the evolving game state. Hybrid approaches, combining the strengths of multiple techniques, are often the most effective for achieving both complexity and coherence.

### **B. Strategic Recommendations for Developers**

Developers seeking to implement procedural generation for non-visual elements should consider the following strategic recommendations:

1. **Start with Clear Design Goals:** Before selecting specific PG techniques, clearly define the desired player experience. What constitutes a "meaningful" contract or an "engaging" outbreak scenario within the specific context of the game? What are the core emotions, challenges, and choices the procedural content should evoke? Having well-defined design pillars will guide the selection and implementation of appropriate PG systems.
2. **Embrace Hybrid Approaches:** No single PG technique is a panacea. The most robust and compelling non-visual content often arises from combining the strengths of different methodologies. For instance, use structured approaches like rule-based systems or CSP/ASP to establish core logic, ensure balance, and maintain constraints. Then, leverage agent-based modeling for dynamic emergence and AI/LLMs for richness, variety in textual presentation, and adaptive NPC behavior.
3. **Prioritize Player Agency and Feedback Loops:** Design PG systems that are highly reactive to player choices. The consequences of a player's actions—whether in fulfilling a contract or attempting to manage an outbreak—should be clear, tangible, and feed back into the procedural systems, influencing future generated content. This fosters a sense of agency and makes the game world feel alive and responsive.
4. **Iterate and Test Rigorously:** Procedurally generated systems, by their nature, can produce a vast range of outputs, including unexpected and potentially undesirable or game-breaking outcomes.22 Implement a robust iterative development process with extensive testing. Consider using AI-driven test agents or developing specific analytical tools to evaluate the balance, fairness, and engagement potential of the generated content across many permutations.
5. **Invest in "Guardrails" and "Curated Randomness":** Especially when working with powerful but less predictable techniques like LLMs or complex emergent systems, it is crucial to implement strong "guardrails" or guiding heuristics.8 These mechanisms help ensure that generated content remains aligned with the design intent, maintains a minimum quality standard, and avoids problematic outputs. Pure randomness is rarely as engaging as "curated randomness" that operates within well-defined creative and mechanical boundaries.
6. **Cultivate the "Meta-Designer" Role:** Recognize that designing with advanced PG systems shifts the designer's role from direct content authoring to that of a "meta-designer".13 This involves shaping the generative processes themselves—defining rules, constraints, evaluation functions, and ethical guidelines for the algorithms. Provide designers with the tools, training, and mindset to effectively manage and curate these complex systems.

The success of future non-visual PG will likely depend significantly on the development of sophisticated authoring tools. These tools need to empower designers, who may not be AI experts, to effectively manage the complexity and harness the expressiveness of these generative systems. Without intuitive interfaces for defining constraints, setting behavioral goals, and curating the output, the broader adoption of advanced PG techniques could be hindered.3

### **C. Future Research Directions and Potential Innovations**

The field of non-visual procedural generation is ripe with opportunities for future research and innovation:

1. **Explainable AI (XAI) in PG:** Developing PG systems where the underlying reasons for generated content are more transparent. For instance, an XAI system could explain to a designer why a particular contract was generated with specific terms, or why an outbreak was predicted to start in a certain location. This transparency can aid in debugging, balancing, and building trust in complex PG systems.
2. **Co-creative PG Tools:** Expanding on current research into designer- controllable PG 3, future tools could empower players to actively participate in the co-creation of procedural content, such as designing mission templates, defining parameters for custom outbreaks, or influencing the rules of an economic simulation, all within carefully defined boundaries set by the developers.
3. **Long-term Narrative and Systemic Progression:** Creating PG systems capable of generating coherent and dynamically evolving economic landscapes or unfolding outbreak narratives over very long playtimes. Such systems would need to adapt to the cumulative impact of player actions and world events, maintaining interest and challenge across potentially hundreds of hours of gameplay.
4. **Ethical Frameworks for AI-driven PG:** As AI plays an increasingly significant role in content generation, particularly with LLMs shaping narratives and potentially influencing player understanding or behavior, the development of clear ethical guidelines and best practices is paramount. This includes addressing issues of bias, originality, and the responsible deployment of AI in interactive entertainment.
5. **Cross-Modal Procedural Generation:** Fostering tighter integration between non-visual PG systems and visual PG. For example, a procedurally generated economic downturn (a non-visual event) could dynamically trigger changes in the visual appearance of game environments (e.g., cities becoming more dilapidated, NPCs wearing simpler clothing), leading to a more holistic and immersive procedurally generated world.

A fundamental consideration that will continue to shape the field is the inherent tension between the desire for truly emergent, unpredictable procedural outcomes and the practical need for reliable, balanced, and consistently "fun" gameplay experiences.22 Unconstrained emergence can be fascinating but can also lead to broken, unfair, or frustrating gameplay. Therefore, ongoing research must bridge PG techniques with deeper understanding of player psychology, engagement models, and user experience design to ensure that procedural systems not only generate content but also craft compelling experiences.

Finally, the relentless increase in computational power and the growing availability of data (including telemetry from player interactions) will undoubtedly fuel the development of even more powerful and data-driven PG techniques.53 This could lead to highly personalized and adaptive game experiences, where economic missions or outbreak challenges are tailored in real-time to individual player preferences, skills, and play styles, far exceeding current approaches to dynamic difficulty adjustment. While this prospect offers exciting possibilities for player engagement, it also amplifies the importance of addressing the associated privacy and ethical considerations thoughtfully and proactively.

#### Works cited

1. polydin.com, accessed May 6, 2025, <https://polydin.com/procedural-generation-in-games/#:~:text=Procedural%20generation%20in%20games%20refers,levels%2C%20and%20even%20narratives%20dynamically.>
2. Procedural Generation | 3D Procedural Generation | Autodesk, accessed May 6, 2025, <https://www.autodesk.com/solutions/procedural-generation>
3. PROCEDURAL GENERATION: AN OVERVIEW AND HISTORICAL ANALYSIS OF ITS IMPACT ON GAME DESIGN - IRJMETS, accessed May 6, 2025, <https://www.irjmets.com/uploadedfiles/paper//issue_1_january_2024/48217/final/fin_irjmets1704966602.pdf>
4. Procedural Generation Techniques in Metaverse Game Worlds - SDLC Corp, accessed May 6, 2025, <https://sdlccorp.com/post/procedural-generation-techniques-in-metaverse-game-worlds/>
5. Exploring Procedural Generation in Games - Polydin, accessed May 6, 2025, <https://polydin.com/procedural-generation-in-games/>
6. Procedural Content Generation in Games: A Survey with Insights on Emerging LLM Integration - arXiv, accessed May 6, 2025, <https://arxiv.org/html/2410.15644v1>
7. arxiv.org, accessed May 6, 2025, <https://arxiv.org/abs/2410.15644>
8. Unveiling New Realms: Enhancing Procedural ... - SMU Scholar, accessed May 6, 2025, <https://scholar.smu.edu/cgi/viewcontent.cgi?article=1012&context=guildhall_leveldesign_etds>
9. Contextually Dynamic Quest Generation Using In-Session Player Information in MMORPG, accessed May 6, 2025, <https://digitalcommons.calpoly.edu/theses/2743/>
10. "Writing for Each Other: Dynamic Quest Generation Using in Session ..., accessed May 6, 2025, <https://digitalcommons.calpoly.edu/theses/2146/>
11. The Future of F2P Games and Hybrid Monetization - Gamigion, accessed May 6, 2025, <https://www.gamigion.com/the-future-of-f2p-games-and-hybrid-monetization/>
12. About Outbreak Simulator - Polymorphic Ed - University of Idaho, accessed May 6, 2025, <https://polymorphiced.uidaho.edu/aboutos.html>
13. Understanding Procedural Generation in Games | Lenovo US, accessed May 6, 2025, <https://www.lenovo.com/us/en/glossary/procedural-generation/>
14. Notes on Procedural Map Generation Techniques - Christian Mills, accessed May 6, 2025, <https://christianjmills.com/posts/procedural-map-generation-techniques-notes/>
15. Creating a Newer and Improved Procedural Content Generation (PCG) Algorithm with Minimal Human Intervention for Computer Gaming Development - MDPI, accessed May 6, 2025, <https://www.mdpi.com/2073-431X/13/11/304>
16. www.inf.puc-rio.br, accessed May 6, 2025, <https://www.inf.puc-rio.br/wordpress/wp-content/uploads/2019/11/196293.pdf>
17. digital.library.unt.edu, accessed May 6, 2025, <https://digital.library.unt.edu/ark:/67531/metadc700051/m2/1/high_res_d/dissertation.pdf>
18. Welcome to Day 1 of GDC 2025: Connecting the World Through Games! | News, accessed May 6, 2025, <https://gdconf.com/news/welcome-day-1-gdc-2025-connecting-world-through-games>
19. The Algorithmic Alibi: When Procedural Generation Steals Player Agency - Wayline, accessed May 6, 2025, <https://www.wayline.io/blog/algorithmic-alibi-procedural-generation-agency>
20. nornagon/constraint-solving-pcg: Using Constraints to ... - GitHub, accessed May 6, 2025, <https://github.com/nornagon/constraint-solving-pcg>
21. Procedural Generation Discussion - Handmade Hero, accessed May 6, 2025, <https://hero.handmade.network/forums/game-discussion/t/1581-procedural_generation_discussion>
22. Procedural Content Generation for C++ Game Development - Packt, accessed May 6, 2025, <https://www.packtpub.com/en-us/product/procedural-content-generation-for-c-game-development-9781785886713?type=print>
23. Procedural world generation oriented on gameplay features - Game Development Stack Exchange, accessed May 6, 2025, <https://gamedev.stackexchange.com/questions/14004/procedural-world-generation-oriented-on-gameplay-features>
24. General - Comprehensive Newbie Guide - Part 3 - Anarchy Online ..., accessed May 6, 2025, <https://www.ign.com/wikis/anarchy-online/General_-_Comprehensive_Newbie_Guide_-_Part_3>
25. Research directions for using LLM in software requirement engineering: a systematic review, accessed May 6, 2025, <https://www.frontiersin.org/journals/computer-science/articles/10.3389/fcomp.2025.1519437/full>
26. A Survey on Large Language Model Based Game Agents - arXiv, accessed May 6, 2025, <https://arxiv.org/html/2404.02039v2>
27. The case for adding AI to side quests - Inworld AI, accessed May 6, 2025, <https://inworld.ai/blog/adding-ai-to-side-quests>
28. Integrating Artificial Intelligence into the Game Development Process - The Data Scientist, accessed May 6, 2025, <https://thedatascientist.com/integrating-artificial-intelligence-into-the-game-development-process/>
29. Is AI Used for Procedural Content Gen in Gaming? - Lenovo, accessed May 6, 2025, <https://www.lenovo.com/us/en/gaming/ai-in-gaming/ai-in-procedural-content-gen/>
30. Beyond the Hype: How AI Enhances Learning in Business Simulations - Advantexe, accessed May 6, 2025, <https://www.advantexe.com/blog/beyond-the-hype-how-ai-enhances-learning-in-business-simulations>
31. From LLMs to LLM-based Agents for Software Engineering: A Survey of Current, Challenges and Future - arXiv, accessed May 6, 2025, <https://arxiv.org/html/2408.02479v2>
32. Procedural Generation using Constraint Satisfaction - YouTube, accessed May 6, 2025, <https://www.youtube.com/watch?v=gKNJKce1p8M>
33. Answer Set Programming for Procedural Content Generation: A Design Space Approach, accessed May 6, 2025, <https://www.researchgate.net/publication/224240207_Answer_Set_Programming_for_Procedural_Content_Generation_A_Design_Space_Approach>
34. course.ccs.neu.edu, accessed May 6, 2025, <https://course.ccs.neu.edu/cs5150f13/readings/smith_asp4pcg.pdf>
35. (PDF) Procedural Content Generation in Games (Guest Editorial), accessed May 6, 2025, <https://www.researchgate.net/publication/224258153_Procedural_Content_Generation_in_Games_Guest_Editorial>
36. List of games using procedural generation - Wikipedia, accessed May 6, 2025, <https://en.wikipedia.org/wiki/List_of_games_using_procedural_generation>
37. Quests - RimWorld Wiki, accessed May 6, 2025, <https://rimworldwiki.com/wiki/Quests>
38. www.foaad.net, accessed May 6, 2025, <http://www.foaad.net/sites/default/files/attachments/AIIDE_2022_Panoptyk_Mendonca.pdf>
39. Panoptyk: information driven MMO engine | Request PDF - ResearchGate, accessed May 6, 2025, <https://www.researchgate.net/publication/335590923_Panoptyk_information_driven_MMO_engine>
40. What in the world gen is procedural and what is random? - Bay 12 Games, accessed May 6, 2025, <http://www.bay12forums.com/smf/index.php?topic=162140.0>
41. Dwarf Fortress Development - Bay 12 Games: Dwarf Fortress, accessed May 6, 2025, <https://www.bay12games.com/dwarves/dev.html>
42. Artifact Quests in Fortress Mode - Bay 12 Games, accessed May 6, 2025, <https://www.bay12forums.com/smf/index.php?topic=176456.0>
43. Dwarf Fortress - Wikipedia, accessed May 6, 2025, <https://en.wikipedia.org/wiki/Dwarf_Fortress>
44. Dwarf Fortress is a single-player fantasy game. You ... - Bay 12 Games, accessed May 6, 2025, <https://www.bay12games.com/dwarves/features.html>
45. How can procedural generation be used to create meaningful quest structures in open-world games?, accessed May 6, 2025, <https://gamedev.stackexchange.com/questions/213231/how-can-procedural-generation-be-used-to-create-meaningful-quest-structures-in-o>
46. Ready, set, respond: How playing an outbreak simulation game ..., accessed May 6, 2025, <https://isr.nyas.org/2024/10/03/ready-set-respond-how-playing-an-outbreak-simulation-game-helps-scientists-prepare-for-the-next-pandemic/>
47. Covid Simulator on Steam, accessed May 6, 2025, <https://store.steampowered.com/app/1871620/Covid_Simulator/>
48. Developing a Video Game as an Awareness and Research Tool ..., accessed May 6, 2025, <https://pmc.ncbi.nlm.nih.gov/articles/PMC12016957/>
49. Infestation - RimWorld Wiki, accessed May 6, 2025, <https://rimworldwiki.com/wiki/Infestation>
50. Disease simulation | TPT, accessed May 6, 2025, [https://www.teacherspayteachers.com/browse?search=disease%20simulation](https://www.teacherspayteachers.com/browse?search=disease+simulation)
51. Predicting Outbreak Detection in Public Health Surveillance ..., accessed May 6, 2025, <https://pmc.ncbi.nlm.nih.gov/articles/PMC2656053/>
52. Comparison of Artificial Intelligence Algorithms and Remote Sensing ..., accessed May 6, 2025, <https://www.mdpi.com/2072-4292/17/5/912>
53. The power of artificial intelligence for managing pandemics: A primer for public health professionals - PMC, accessed May 6, 2025, <https://pmc.ncbi.nlm.nih.gov/articles/PMC11704850/>
54. Planning for a disease outbreak? There's a game for that ..., accessed May 6, 2025, <https://www.sciencedaily.com/releases/2016/02/160201220324.htm>
55. www.nacaa.com, accessed May 6, 2025, <https://www.nacaa.com/file.ashx?id=01804777-5a51-48f6-bbec-3def3cfea502>
56. Serious Game Design for Teaching University Students to Address Complexity Issues in the Healthcare Logistics System: Lessons from an Emergency Department Case Study - MDPI, accessed May 6, 2025, <https://www.mdpi.com/2079-8954/13/3/197>
57. Using experimental gaming simulations to elicit risk mitigation ..., accessed May 6, 2025, <https://scholarworks.uvm.edu/cgi/viewcontent.cgi?article=1059&context=calsfac>