Procedural Scenario Generation

Jeremy Hummel
Computer Science Department
Western Washington University
Bellingham, WA, US
hummelj3@wwu.edu

Steven Marantz
Computer Science Department
Western Washington University
Bellingham, WA, US
marants@wwu.edu

Ian Ritter
Computer Science Department
Western Washington University
Bellingham, WA, US
ritteri@wwu.edu

***Abstract*— a one paragraph summary with a sentence each on the objective, approach, results, and main conclusions.**

Keywords—Artificial Intelligence, ...

I. Introduction

CONTEXT

Medical students need extensive training before going into the field and medical professionals often need ongoing training throughout their careers. This training is crucial to a medical professional's career and gives medical students the proper experience to be able to treat real people with varying conditions. However, that training must be extensive and varied to be effective, it requires proper equipment and it needs to be carried out by highly trained domain specialists and qualified instructors which results in this training being expensive and less accessible to training programs with low budgets. This has pushed the development of scenario-based training. By utilizing multiple aspects of simulation, scenario-based training is designed to provide an experience that effectively mimics a real situation. This provides the most convincing experience but, due to its complexity, it is expensive and time-consuming to develop [1][5]. There is value in an automated solution that can rapidly produce quality medical scenarios for virtual training. Virtual training offers the advantages of self-directed learning, facilitates trainees developing skills at their own pace, and provides opportunities for unlimited repetition in a safe learning environment.

PROBLEM

Scenario generation is a mathematical algorithm that contains random or unknown data points. Generally, these data points are described as continuous random variables (CRV), which essentially is a variable that can take on infinitely many values. Due to the infinite possibilities of these CRV's, a discrete distribution with a finite number of scenarios must be created in order to constrain the infinite possibilities of the parameters. This discrete distribution is the most important part of scenario generation as we need to obtain a realistic set of scenarios from a set of parameters that can take infinitely many forms.

Our focus is on generating a realistic scenario in a simplistic environment, with a procedurally generated patient that has a rich medical history. This background may include traits passed down to offspring, and acquired traits throughout a given patient's life. An example of a medical background could include that the patient is allergic to peanuts, and is prone to diabetes from their grandmother who passed that trait to the patient's father. We are focusing on creating an automated solution that generates complex patient profiles rapidly in a valid manner, and free of human intervention. The goal of the generated scenario is to determine if digital scenario generation is an efficient tool at training medical students.

CHALLENGES

Many existing solutions to training medical students are not digital. Training people to act as a patient [6] is an existing solution but has many drawbacks. This solution is not only very costly and time consuming, but is also easily prone to human error. Alongside the overhead in training and potential for errors, these simulated patients are not easily distributable as they are human beings that cannot be sent over an internet connection.

One of the challenges associated with our problem is to address the ambiguity of the medical field. Many symptoms are not directly indicative of a single ailment, but rather a long list where common symptoms are shown. For example, a common headache could just be a common headache, or it could be an early sign of something more severe such as trigeminal neuralgia. This ambiguity is so challenging to overcome as the medical field is always expanding with new treatments, and the discovery of new diseases. In order to get past the challenge, the scope of the solution must be defined. Determining what the set of ailments and symptoms are that can be applied to a patient is pivotal to creating a realistic and believable patient.

Another challenge associated with our problem is the ability to provide useful feedback to the user. We must provide objective feedback that will tell the user what the most likely diagnosis would be for a given patient, alongside a list of less likely but still possible ailments for the given patient. The reason that this feedback is a challenge is because building a list of ailments where new ones are created all the time, may mean that our problem does not adequately simulate the real world. For example, before COVID-19 was a well known disease it may have been common for patients to be wrongly diagnosed with the Flu, as they share many of the same symptoms. As the medical field is always growing and expanding, an exhaustive list of all ailments and symptoms is not a goal that is not in the scope of our solution.

STATE OF THE ART

The most standard medical student training medium is through actors or different types of high and low-fidelity medical simulation manikins, both of which are expensive, not scalable, and not accessible to programs with low budgets. To solve these problems, as with many things nowadays, virtual solutions have been proposed and created starting the move of medical training to the virtual world.

There are several examples of virtual medical training simulators such as Shadow Health [9] that offer an extensive suite of healthcare simulation products for training in healthcare fields that are supposed to provide an effective, affordable, scalable, and patient-centered learning experience. These virtual solutions solve many of the problems introduced by actor or manikin-based training however they come with their own set of problems such as they often have fixed scenarios, they are expensive and time-consuming to develop and some solutions accidentally train students to do the wrong thing by introducing certain biases.

OBJECTIVE

In our work towards automatic generation of quality training scenarios, we will utilize procedural generation to produce patients with simulated histories. This will include recent health history such as their fitness, BMI, diet, mental health, and living and working environments. It will also include the generation of several members of their immediate family with similar characteristics in order to provide subjects that may contribute to the story of the patient. This may lead to information including hereditary conditions and behavioral predispositions. We will develop a model of patients to represent ailment characteristics, such as symptoms and sources, that we can generate patients from with variable overlapping symptoms. A PCG algorithm will employ the information in this model to generate patients who will be able to provide detailed information about not only their current health concerns but the context with which they are taking place. This will provide a much richer training experience as a diagnosis may require that the trainee dig deeper and face potentially contradicting information from which they will need to weigh in order to assign a diagnosis.

To test this system, we will build a prototype environment in which a virtual patient will be generated. The trainee seeing the patient will have access to a chart that shows the typical patient information standard to a hospital visit. The patient will then convey their concerns about their health including what symptoms they've been experiencing. After which, the trainee can begin asking questions to determine the perceived lifetime of the symptoms and their intensity. From there, they may need to ask further questions about the patient's environment and family history in order to see if there may be other factors in play.

Once the trainee has provided their diagnosis and stated their confidence, they will be presented with a patient report. This report will display all the information about the patient and show a list of ailments that match at least one of their symptoms rated from most to least relevant. The purpose of this list is to show the trainee how close they were to the most likely ailment. Their diagnosis and their confidence in it will help them to identify common errors. With this knowledge, they can focus their studies more effectively and develop improved confidence in their decisions. A user study will be performed to evaluate the effectiveness of this prototype.

II. Related Work

A. Procedural Generation in Relation to Training

The goal of training is to provide a trainee with experience so that they are better prepared when facing real-life situations. However, that training must be extensive and varied to be effective. This has resulted in the development of scenario-based training. By utilizing multiple aspects of simulation, scenario-based training is designed to provide an experience that effectively mimics a real situation. This provides the most convincing experience but, due to its complexity, it is expensive and time consuming to develop. Similar to our project, **The Use of Functional L-Systems for Scenario Generation in Serious Games** [1] seeks to address this issue by exploring the possibilities of procedural generation to automate the scenario development process. Having determined that, while many scenario generation methods exist, they are domain specific, they designed a system that aims to be domain independent. This system utilizes Functional L-Systems, a form of procedural generation, to automatically develop military scenarios. Their resulting work shows progress towards this goal but no experiments were reported to evaluate the effectiveness of their approach.

3D Character Generation using PCGML [2] presents a method to procedurally generate the Non- Playable Characters (NPCs) in video games using a modified Style-based generative adversarial network (StyleGAN) which is a type of neural network. Procedural Content Generation using Machine learning (PCGML) is the most cost- effective method for game content generation, and was employed to reduce production effort and to save storage space. Their approach adapts the use of PCGML with StyleGAN to generate NPCs that are unique in both appearance and behavior. The properties or traits influence the generation of characters making the game environment diverse and interesting for the players. In our case we have to generate patients and it would be best if patients all felt unique in appearance and medical history.

Data Driven Approach to Multi-Agent Low Level Behavior Generation in Medical Simulations. Creating a realistic and believable environment full of NPC's doing different jobs and tasks is a complex task. Using sensors in a real-world hospital [3] aimed to produce a realistic hospital with many NPCs that perform different duties, where the goal of the simulation is to help educate basic handwashing and hygiene. These sensors log movement data captured within a hospital ward that was then used to fill the simulation with believable characters. Looking at their results they found that making a believable simulation was effective at training basic hand hygiene and heavily relied on the fact that the hospital was a believable simulation. This is important for our project to understand that the simulation will yield better results if the simulation can be taken as a real-life scenario.

In **Automated scenario generation: toward tailored and optimized military training in virtual environments** [4] they show the advantages of using an automated scenario generator for virtual training. An automated scenario generator is a system that creates training scenarios from scratch, augmenting human authoring to rapidly develop new scenarios, providing a richer diversity of tailored training opportunities, and delivering training scenarios on demand. They introduce a combinatorial optimization approach to scenario generation to deliver the requisite diversity and quality of scenarios while tailoring the scenarios to a particular learner's needs and abilities. They also propose a set of evaluation metrics appropriate to scenario generation technologies and present preliminary evidence for the suitability of their approach compared to other scenario generation approaches. Similar to this study we wish to achieve believable training scenarios that require little input to return a quality output.

In **Automatic scenario generation through procedural modeling for scenario-based training** [5] they discuss how the creation of validated, effective scenarios is time-consuming and must be carried out by highly trained domain specialists, qualified instructors, and technology experts and is therefore expensive. They also touch on the issue of how scenario libraries often fail to cover the optimal range of possible mission contexts and how to solve it by automating the process of scenario generation. Automated scenario generation must be supported by appropriate algorithmic modeling approaches, as well as research-based heuristics to ensure that scenario outputs are domain-valid and pedagogically effective. They explore this concept from the perspective of computer science and training science. From the perspective of CS, they review procedural modeling techniques that can support automated scenario generation and from the perspective of training science they discuss guidelines and research approaches for developing objective guidelines to ensure scenario effectiveness. Like this study we also aim to create validated and effective procedurally generated scenarios for training purposes.

B. Non-Procedural Generation Training Simulations

The real-life patients described in **Use of Simulated Patients in Medical Education** [6] fulfill a role very similar to that of our procedurally generated patient. They are professionally trained to know all necessary aspects of the conditions they simulate and are used extensively in medical training. Their use is valued for providing 8 areas where real patients with the ailment are not available. The benefit of using simulated patients is explored in "**Are Simulated Patients Effective in Facilitating Development of Clinical Competence for Healthcare Students?**" [7]. This scoping study explored the results of 33 studies on the subject and found that 24 of them showed that the use of simulated patients resulted in improved clinical competence in students.

Scenario generation for emergency rescue training games [8] is related to our project through the use of procedural generation to create a simulation in order to train and educate workers in an industry. Their goal was to train emergency firefighters and rescuers when clearing a collapsed building. Their idea was to procedurally generate the buildings using a set of rules where rooms can be collapsed based upon the structural integrity of other walls. The reason this is important is to make the simulation as realistic as possible. Where this applies to our project is to make sure the ailments make as much sense as possible and when applied change different aspects of the model. if the character is missing a leg, they may need crutches or a wheelchair to walk. Knowing these ailments and understanding how they will impact the NPC is very important.

Shadow Health [9] is an extensive suite of healthcare simulation products for nursing and allied health care fields that are supposed to provide an effective, affordable, and scalable path to experiential and patient-centered learning. Shadow Health offers users the ability Interview the virtual patients by engaging in open-ended conversations to gather subjective data and practice patient-centered communication. They also offer users the ability to perform tests and use instruments on the virtual patients to gather and record objective patient data. Shadow Health's approach is similar to ours in that they focus on a user using gathered information on a virtual patient to properly diagnose them; however unlike our approach Shadow Health does not utilize procedural content generation .

Online Digital Education for Post registration Training of Medical Doctors: Systematic Review by the Digital Health Education Collaboration [10] reviews the results of 93 studies involving the effectiveness of online digital education (ODE) for medical doctors. The focus was to see if ODE was effective in improving practicing medical doctor's knowledge, skills, attitude, and satisfaction. The results of the review were that ODE is generally comparable to self-directed or face-to-face learning. However, the studies that were reviewed were of inconsistent quality showing that further research in this area is required.

Training for our Digital Future: A Human-centered Design Approach to Graduate Medical Education for Aspiring Clinician-innovators [11] explains how the need for better health care at lower costs is driving the need for innovation in care models. However, many aspects of current models actively resist improvement for reasons ranging from lack of available resources to attachment to outdated technologies. The proposed solution is employing a design methodology that emphasizes empowering the trainee with the skills and experience to develop and implement innovation. Our project is driven by the desire to support and facilitate this innovation in medical training.

Development of Patient Scenario Generation which can reproduce characteristics of the patient for simulating real-world conditions of task for airway management training system WKA-3 – Generating training scenarios is a complex problem especially in the field of medicine as these scenarios in the real world may have severe consequences. [12] lists some factors that can be measured in order to determine success: the scenarios must represent a real-world condition, provide objective assessments to the trainee's, and allow for useful feedback for further training. However, this may be useful to a trainee, to a beginner some of the steps followed may not be clear to them and would need further instruction in order to accomplish the scenario. This is important for us to understand as if ailments are too complex, we won't be able to provide an objective assessment of how well they performed.

C. Other Procedural Generation Applications

In [13] **A survey on the procedural generation of virtual worlds**, the researchers explore a multitude of procedural generation concepts and explain how they may be used in practice. We can use this paper to help us define procedural generation as a topic. They claim that procedural generation is the generation of an asset based on an exposed set of input parameters. Our goal is to generate a model with different appearances and procedurally generated ailments that will then be treated by our user. Also of interest are the discussions on Humans in this paper, as it lists out some possible inputs such as stride length, body tension, and corpulence.

Rule-based procedural generation of item in role-playing game – Procedural generation has been frequently used by programmers for different types of content. The most obvious type of generation is terrain, but this is not the only use of PCG. Rule based item generation is based on a subset of rules such as drop chance, number of items, quality, durability, rarity and many more. As shown in [14], a rule-based randomized approach can be more effective at procedurally generating items than a totally randomized algorithm. They show that a completely random algorithm does not produce adequate results in creating fun and unique items. Generating a set of rules can easily curb this randomization problem and is easily expandable to add more rules in the future.

Let CONAN tell you a story: Procedural quest generation [15] is a procedural quest generator that uses an AI planning approach for dynamic story generation. The engine in this project takes in a world description represented as a set of facts, including characters, locations, and items, and generates quests according to the state of the world and the preferences of the characters. This is similar to our goals as it pairs basic elements together to build stories or quests that make sense from given inputs. We need to build a system that pairs symptoms together intelligently to mimic real world illnesses just like this system might make a quest by intelligently pairing items, characters and events it's given into something that is coherent.

Toward Supporting Stories with Procedurally Generated Game Worlds [16] is an approach to procedurally generating, rendering, and making playable novel games based on a priori unknown story structures. These stories may be authored by humans or by computational story generation systems. For example, an ailment, like a quest, has certain basic properties that combine to make it unique like a quest might be composed of solving a puzzle and turning in an item. For us those properties may be something such as a sore throat paired with a headache or rash paired with a fever, and like a quest there are things that need to be done to complete a quest or in our case treat a patient.

III. Approach

Our approach is to develop an algorithm utilizing procedural content generation that can create NPC patients with several layers of detail that are meaningfully connected. One challenge is ensuring that the system generates meaningful results. This necessitates having a model that describes how patient attributes are related. For example, the quality of the patient's relationships with relatives they live with can affect mental health and their income can affect their diet which, in turn, can affect their BMI.

A second challenge is defining a small subset of potential ailments and clearly defining them. The ailments will need to be broken down into a collection of symptoms and characteristics that we can use as a pool to pull from when generating NPCs. An automated algorithm will need to be able to assign symptoms automatically in such a way that their correlation may be relevant to the assigned ailment.

With this web of interrelated characteristics that make up the patient's story, we can provide a realistic representation of a patient. A prototype virtual environment will demonstrate that the generated patients can provide more useful diagnosis training for medical students. A patient will have one or more symptoms, at least one of which has become disruptive enough to persuade them to visit a doctor. The medical trainee's task will be to start with the symptoms and their history. Then dig deeper to see if they can find a cause. This is where the patient's complexity can pay off in a training scenario. The doctor can ask questions like how the patient has been sleeping, if they regularly exercise, how stressed they feel, etc. And the patient will be able to provide reasonably realistic answers. The trainee must then decide what characteristics qualify as correlations and search their knowledge to find the most appropriate diagnosis.

One challenge in demonstrating a solution is the massive scope of potential ailments. Our approach is therefore limited to a subset of respiratory ailments to restrict the scope of ailments for proof of concept. Interactions with a patient are limited to a dialogue to focus on diagnosis rather than interactions. Patient generation focuses on the ailments and medical history of a patient, not their physical traits or visual elements. To this end, our system consists of a diagnostic patient model to capture patient medical history, an engine for procedural generation of the patient history, and a prototype virtual environment to explore the generated content. The intended application is not the treatment of ailments, it is on testing a student's knowledge.

Our design consists of a model to represent patients, a procedural generation engine to dynamically generate patient data, and a prototype system to demonstrate our approach (ref. Fig 1).

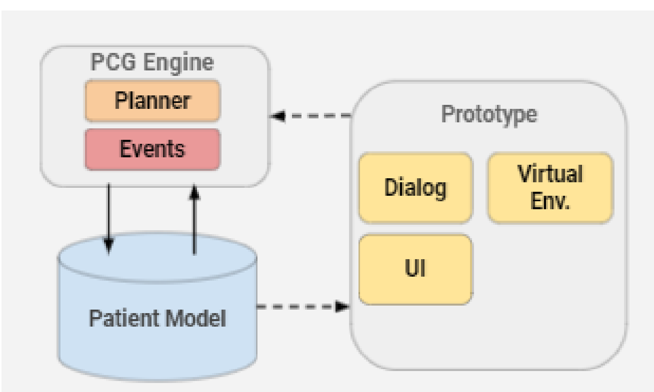


Fig 1. System diagram.

A. Diagnostic Patient Model

A model refers to an abstract representation of the attributes and relationships to describe an entity in a particular problem domain. It's a representation of how different properties fit together and relate to one another. For example, a cat can be represented as having four legs, whiskers, likes to catch mice, meows, and has an owner. The reason we need these models is that we need to understand how a real patient would be structured, what information a doctor or medical student would have access to, what they would need to know about a patient, and what could affect a patient's medical history. The intended effect of this model is that the patient may be generated with a complexity similar to that of a real patient.

In order to generate a patient with complexities and ambiguities similar to a real patient, the characteristics that define a patient must first be established. This includes the patient's vitals when they visit the doctor, details about their personal and professional relationships that could relate to their physical and mental health, careers, habits, income and healthcare access. *Vitals* are a simple data structure containing heart rhythm, respiratory rate, blood pressure and temperature. *StaticInfo* contains all things that don't change throughout a patients life such as name and sex. *Habits* store data such as whether a patient smokes and the quality of the patient's diet. *Metrics* contain things such as height and weight. *MedicalHistory* stores past maladies and the *ChiefComplaint*. The *ChiefComplaint* is what the patient is going to the doctor for, so this data structure contains all information relating to the malady the patient is inflicted with. These attributes of the patient model are shown in the diagram below (ref. Fig. 2).

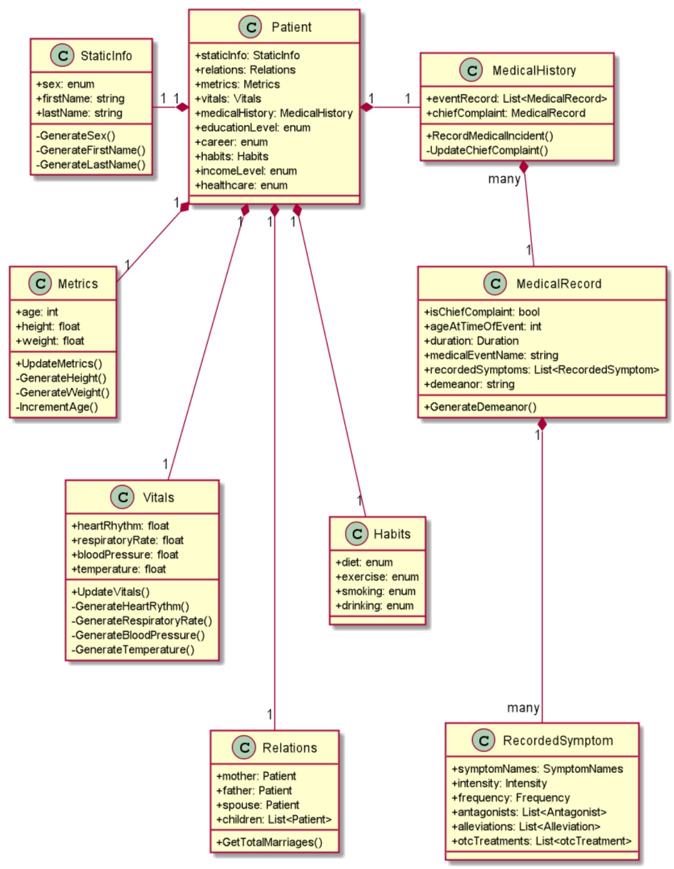


Fig 2. Diagram of the Diagnostic Patient Model.

The condition of the patient will be established based on their relationships, habits, careers, past maladies, as these have both immediate and long-term effects on the patient's physical and mental health. This includes predispositions towards various ailments based on environmental factors such as a patient's at-home life, jobs or career, relationships, and/or hereditary factors such as particular ailments run in the family like Cystic Fibrosis and some cancers. The model is designed to track and store information for all such concerns. As an example application of our model, assume we want to represent a patient named Fred Ward. Fred is allergic to peanuts, has a wife and two kids, and gets regular exercise. His kids expose him to cold and flu fairly regularly and he has a higher risk of getting diabetes because his mother had it. The name 'Fred Ward' would be stored in *StaticInfo* and his allergy would be recorded in *MedicalHistory*. Fred's wife and two kids would be stored in *Relationships*. History of the cold and flu would be stored in the *MedicalHistory* of the patient model. Freds high likelihood to develop Diabetes would be decided based on his relationships and also stored in *MedicalHistory.* All of Fred's past Maladies (cold, flu, diabetes if he contracted it, etc.) would be stored in *MedicalRecord* under *MedicalHistory*. Any symptoms associated with these illnesses would then be captured in *RecordsSymptoms*.

B. *Procedural Content Generation Algorithm*

Populating the patient model automatically, while ensuring patients that are generated are diverse and have a history that makes sense, is a significant challenge. Random generation can provide diversity but outcomes are, by definition, unpredictable. To ensure that results are realistic, a more sophisticated system is needed. AI planning provides a way to guide this randomness to more predictable results. Our approach utilizes AI Planning by treating all possible changes a patient can be subject to as a collection of operations. These changes represent the patient's life experiences and are referred to as events (ref. Fig. 3).

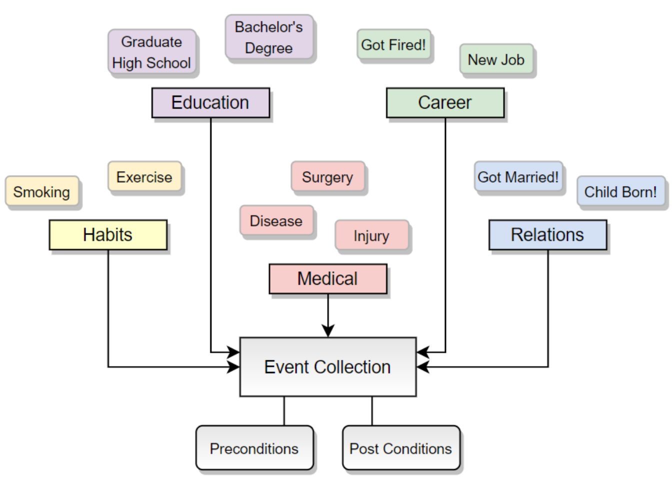


Fig 3. Building the Event collection.

Each event has associated preconditions that determine if the event can logically occur and post conditions that will be applied, transforming the state of the patient, if the preconditions are met (ref. Fig. 4). Probability, a precondition that is shared by all events, is determined by the patient's current information allowing for a degree of configurable randomness. For example, poor health care or a smoking or drinking habit may make the patient more likely to get sick while a good diet and exercise may have the opposite effect.

The planner generates a patient by cycling through each year of their life, applying events as they age. It is this chronological approach that allows each new event to be influenced by all events before it. That is, each event occurs within the unique context of the patient's life at the time the event occurs.

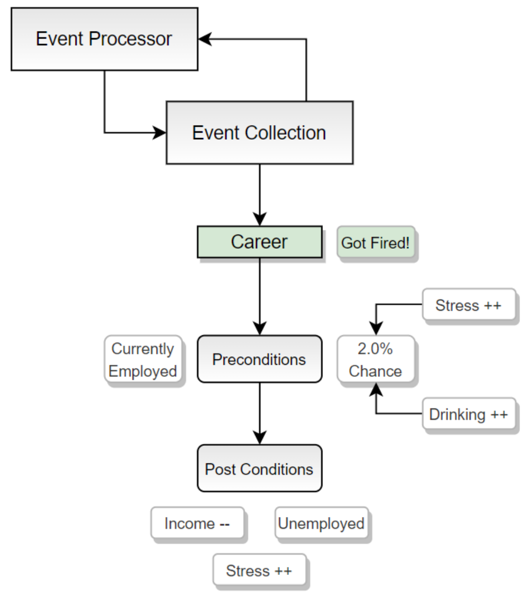


Fig 4. Event Structure.

To determine the patient's hereditary vulnerabilities, the patient's parents are generated first. The patient is then generated with predispositions resulting from a combination of their parent's medical histories. From that point, the family ages together, having their lives altered by events as they grow. As the patient gets married and has children, these new family members are also procedurally generated and grow along with the family from the point of their introduction on (ref Fig. 5).

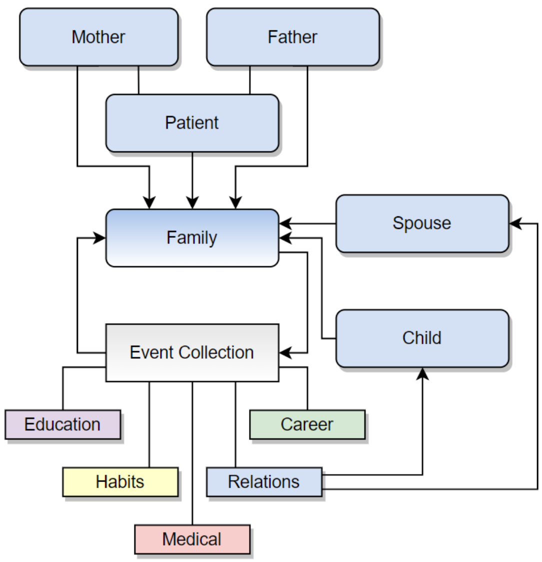


Fig 5. Family Generation Flow.

This process concludes when the patient reaches a predetermined age and catches a predetermined disease. The specifics of the disease and the overall health of the patient are determined by the context of their procedurally generated life. Bringing this complexity to diagnosis training allows the simulation to present a more realistic patient scenario where the patient's personal experience with disease mimics the ambiguity of real life. The medical trainee interacting with the system must deal with incomplete, or muddled information, perform tests to dig deeper, and ultimately make their best, educated guess at what disease the patient is suffering from.

C. Prototype

To provide a proof of concept, a prototype training system set in a 3D virtual medical clinic was developed (ref Fig. 6). Included in this prototype are a user interface and a dialogue system for interacting with a patient.

Fig 6. Prototype Environment.

The dialogue system is the primary avenue for interacting with the patient. This system facilitates interaction between the doctor and the patient through a question and answer process. A user can ask questions of the patient and get answers, based on the data in the underlying patient model. This system's behavior is defined in a node-based manner, where dialogue is specified as a graph representation (ref. Fig 7). Nodes represent dialogue text and node coloring indicates who is speaking. Blue nodes indicate that the doctor is asking the patient a question, and grey nodes indicate that the patient is responding to the doctor's question. Edges between nodes indicate possible paths for the conversation to go next. For example, when the doctor asks the question, "Do you drink at all" the outward edge indicates which node will be the response from the patient.

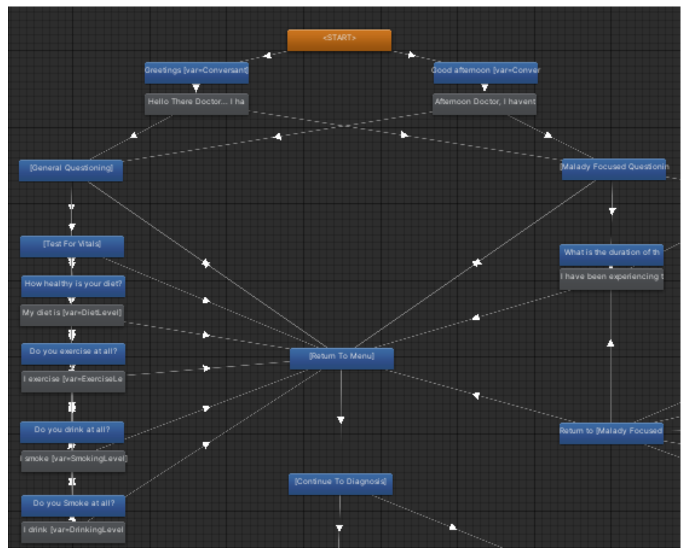


Fig 7. Visual Representation of Dialogue.

The dialogue system enables conversation-like interaction between the user and the patient. Such a conversation is split up into two major categories referred to as "General Questioning" and "Malady Focused Questioning".

General questioning is a section about the metrics and habits of the patient. For instance, how much do they drink, do they smoke at all, do they have a balanced diet, and more. This facilitates gathering background information about a patient that may or may not be present on a medical record. Questions in this section are purely informational, and populate elements within the information clipboard. This clipboard has the patient's name, their sex, when they were born, their height, their weight, alcohol & drug usage, as well as the patients vitals. Parts of this information are populated when the doctor asks a specific question. For example, when the doctor asks "How much do you drink", the clipboard (ref. Figure 8) will populate the "Drinks:" field with the response from the patient. This allows for information to be present within a UI element, without having the user remember the patient's answer or take notes manually.

Malady Focused Questioning is a section centered around the idea that the patient is suffering from a specific malady. The doctor needs information to determine exactly what malady the patient is suffering from. Questions such as “How intense is the pain” or “how long have you been experiencing these symptoms” are primarily focused on the symptoms the patient is experiencing. This section's main purpose is to address the chief complaint that the patient has which includes information such as the symptoms, the intensity, what makes the pain worse, what makes the pain better, and if the patient has tried anything already that the doctor can rule out.

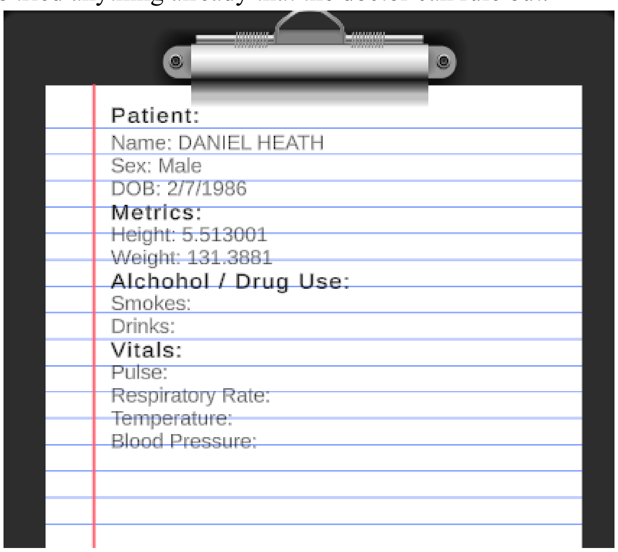


Fig 8. Information Clipboard.

Choosing a diagnosis is also an important part of the prototype, which is done through a simple menu after the conversation with the patient has ended. When the conversation ends, an event gets fired which opens a menu with a multitude of buttons that represent a user's options for diagnosis . Due to the ambiguity of the medical field, diagnosis is a very hard problem to solve since the diagnosis of a malady is usually done by running extensive tests on the patient. It is important to note that the menu to choose what malady the patient is suffering from is a placeholder, in its current state the menu is far too rigid to reflect an actual diagnosis.

IV. Results

A. Experimental Setup

Our prototype system was built in Unity Version 2019.4.11f1. The system testing was performed on a Windows 10 machine, with a HP EX900 SSD, 16 gb of DDR4 ram, AMD Ryzen 5 3600 6-Core 3.59 ghz processor, and a 2060 Nvidia GTX gpu.

B. Evaluation Plan

Evaluation of our prototype system will start with running performance tests. For performance testing, we will generate three sets of patients: 1k, 15k, and 100k. These tests will give us data points to determine the growth rate of our system’s performance as the problem size increases. Performance indicators of memory management, CPU utilization, framerate, and overall runtime will be gathered using Unity’s built-in profiler. Performance testing will be iterative, running tests, reviewing the results, and making code optimizations where appropriate to address bottlenecks.

Another area of evaluation is validity as verification that the system’s procedurally generated patients do not violate the rules specified by the preconditions. For instance, a patient cannot have a bachelor's degree if they never graduated high school. To evaluate validity, we will generate a set of 10k patients and manually check a statistical sample (~5%) of those generated patients.

The major evaluation piece focuses on the diversity of the generated patients. Diversity refers to the degree of variance amongst a generated set of patients. Instrumentation was added to the prototype to create a snapshot of patient data when generated that is written to a spreadsheet. Statistical analysis tools will then compute averages and standard deviations to analyze the numerical data. Non-numerical data will be analyzed via occurrence rate, determining what portion of patients have a given attribute. Calculations will also be performed to analyze how often a given single patient reoccurs, and the uniqueness of said patient. Finally, a generalized diversity value is generated to directly compare patients. This value is generated from all of the attributes, habits, and relationships associated with a single patient. Each of these values has a numerical value equivalent, and when summed, will result in a generalized diversity value.

C. Analysis

Performance

The initial results of performance testing were not as promising as expected. Generating seven or more patients took significantly longer than the first six. Using timers, Unity’s Profiler, and code review, it was determined that the source of performance loss was excessive object deletion which resulted in the garbage collector (GC) becoming overloaded. This issue was solved by introducing object pooling. Additionally, the use of array-based lists caused further performance loss, and so was refactored to a model that redefined existing lists rather than adding and removing elements from them. The result was a significant improvement in performance of approximately 1,500 patients generated per minute with our hardware (ref. Fig 9).

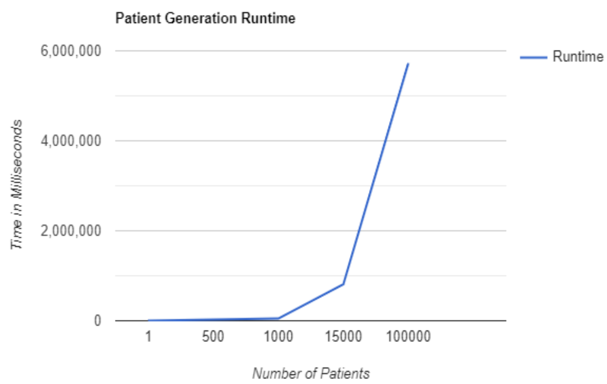


Fig 9. Runtime Graph

Validity

Due to the procedural techniques employed by the algorithm, the content that is generated follows the algorithm's predefined rules. Validity analysis took place by hand, and it must be noted that we are not the subject matter experts when it comes to a patient and their life story. Hand driven analysis of a small subset of patients were analyzed for valid attributes, logical relationships, and convincing life stories. Our analysis had found that none of the patients had violated the planners predefined rules.

Diversity

After resolving the initial performance issues with the garbage collector, our initial diversity results were very promising but required tuning the procedural algorithm. One of the first problems we had recognized was that some patients had a higher chance of repetition due to the random assignment of a planner variable known as the “seed”. Given two patients with an identical seed, all of the associated values attached to the patient would be the same across both patients. Initially we had found that out of 15k patients, ~14k of them were unique appearances of a single patient. After adjustments to the algorithm, out of a set of 15k generated patients we found 15k unique appearances of a patient, with only ~500 repeat occurrences of the same name.

Statistical analysis of patient attributes was conducted to determine the rate of occurrence for each of the non-numerical attributes (ref. Fig 10). Some attributes such as education level found a disproportionate amount of Bachelor's degrees. Further tuning of the probability distributions utilized by the procedural algorithm can easily address this problem. Similar results can be found in the career and income fields, but instead of upwards of 50% being allotted to one value, it was in the lower to mid 30%. The generalized diversity value (shown in Fig. 11) is a generalized single number that represents the attributes of a single patient all summed together. Two patients with the same generalized diversity value may look completely different from under the hood. By analyzing the distribution of values taken from the graph we found an acceptable level of deviation from patient to patient.

Education	Count	Rate
LittleOrNone	873	0.06
HighSchool	582	0.04
Bachelors	8379	0.56
Masters	3429	0.23
Doctorate	1737	0.12

Diet	Count	Rate
VeryPoor	3504.0	0.23
Poor	3562.0	0.24
Medium	3013.0	0.20
High	2609.0	0.17
VeryHigh	2312.0	0.15

Career	Count	Rate
None	1404	0.09
Labor	2217	0.15
Sales	3605	0.24
Tech	5474	0.36
Entrepreneur	2300	0.15

Exercise	Count	Rate
VeryLow	3501	0.23
Low	3510	0.23
Moderate	3027	0.20
High	2645	0.18
VeryHigh	2317	0.15

Income	Count	Rate
Poverty	512	0.03
LowIncome	2718	0.18
MiddleClass	3894	0.26
UpperMiddleClass	5576	0.37
Wealthy	2300	0.15

Drinking	Count	Rate
None	2396	0.16
Little	2661	0.18
Medium	3110	0.21
High	3400	0.23
VeryHigh	3433	0.23

Smoking	Count	Rate
None	2852	0.19
Little	2984	0.20
Medium	3104	0.21
High	3139	0.21
VeryHigh	2921	0.19

Fig 10. Data Results

The results of our performance, validity, and diversity analysis were promising. Performance testing was easily optimized and resulted in an algorithm that could rapidly generate synthetic patient data much faster to a non-digital counterpart. Validity results show promise but require further analysis by a subject matter expert. Diversity results can be further tuned, but again show a high degree of variation and result in a unique set of patient scenarios.

Procedural generation follows a predefined set of rules, and the content generated is a valid byproduct of those rules. Random generation on the other hand has no defined ruleset or conditions it must follow. If we were to randomly generate a patient using the same patient data model, we would find that results don't follow a trend and are completely disjoint. For example, a patient named John Smith who is 39 years old, weighs 50 pounds and is 6’5”. This example is clearly not realistic as the weight of an adult male who is 6’5” should not be equal in weight to a large bag of dog food.

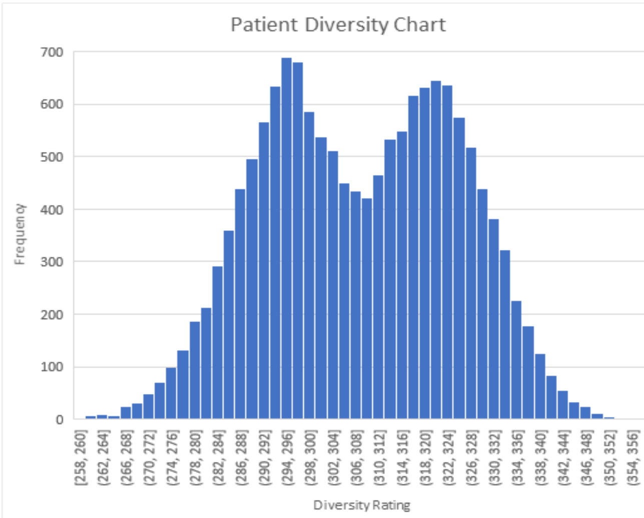


Fig 11. Patient Diversity Chart

V. Summary & Future Work

This paper presents a novel approach that employs procedural generation techniques to rapidly generate diverse patient scenarios for medical diagnostic training. A major contribution of this research is the procedural algorithm, which utilizes AI planning techniques to generate diverse patient scenarios. The two minor contributions of this project is a scalable patient model, which supports complex life stories and allows for a more realistic patient representation. Secondly, the prototype also provides a 3d test environment for a user to interact with the generated patients via a dialogue system. Evaluation of our system demonstrates that diagnostic training scenarios can be rapidly generated and provide valid and diverse patient data.

There are many avenues forward for this project, and one of the biggest ones to further evaluate our systems is a comparison to random generation. Randomly generating a patient and comparing it to procedurally generated will allow us to further evaluate the validity of the generated patient data. Another future work would be to include a wider range of potential maladies, rather than just respiratory, the expansion to include neurological, cardiovascular, and other types of maladies. Lastly, to finish our evaluation of the systems implemented, a subject matter expert should be consulted to evaluate the validity and usability of our system.

References

[1] Martin, Glenn A., et al. "The use of functional L-systems for scenario generation in serious games." Proceedings of the 2010 Workshop on Procedural Content Generation in Games. 2010.

[2] Balika J. Chelliah, Vijay Krishna Vallabhaneni, Saikiran Reddy Lenkala, Mithran J, M Kesava Krishna Reddy. “3D Character Generation using PCGML”. IJITEE, 2019.

[3] Gupta, Manan. “Data Driven Approach to Multi-Agent Low Level Behavior Generation in Medical Simulations.” (2012).

[4] Zook, Alexander, et al. "Automated scenario generation: toward tailored and optimized military training in virtual environments." Proceedings of the international conference on the foundations of digital games. 2012.

[5] Martin, Glenn, et al. "Automatic scenario generation through procedural modeling for scenario-based training." Proceedings of the Human Factors and Ergonomics Society Annual Meeting. Vol. 53. No. 26. Sage CA: Los Angeles, CA: SAGE Publications, 2009.

[6] Cleland, Jennifer A., Keiko Abe, and Jan-Joost Rethans. "The use of simulated patients in medical education: AMEE Guide No 42." Medical teacher 31.6 (2009): 477-486.

[7] Williams, Brett, and Jane Jee Yeon Song. "Are simulated patients effective in facilitating development of clinical competence for healthcare students? A scoping review." Advances in Simulation 1.1 (2016): 6.

[8] Hullett, Kenneth, and Michael Mateas. "Scenario generation for emergency rescue training games." Proceedings of the 4th International Conference on Foundations of Digital Games. 2009.

[9] Shadow Health. https://www.shadowhealth.com/

[10] George, Pradeep Paul, et al. "Online digital education for postregistration training of medical doctors: systematic review by the Digital Health Education Collaboration." Journal of medical Internet research 21.2 (2019): e13269.

[11] Carter, Jocelyn, Yanik J. Bababekov, and Maulik D. Majmudar. "Training for our digital future: a human-centered design approach to graduate medical education for aspiring clinician-innovators." NPJ digital medicine 1.1 (2018): 1-3.

[12] Noh, Yohan et al. “Development of Patient Scenario Generation which can reproduce characteristics of the patient for simulating real-world conditions of task for airway management training system WKA-3.” 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (2010): 331-336.

[13] Freiknecht, Jonas, and Wolfgang Effelsberg. "A survey on the procedural generation of virtual worlds." Multimodal Technologies and Interaction 1.4 (2017): 27.

[14] On, Chin Kim, et al. "Rule-based procedural generation of item in role-playing game." Int. J. Adv. Sci. Eng. Inf. Technol 7.5 (2017): 1735.

[15] Vincent Breault, Sebastien Ouellet, Jim Davies. “Let CONAN tell you a story: Procedural quest generation”. IEEE, 2018.

[16] Ken Hartsook, Alexander Zook, Sauvik Das, and Mark O. Riedl. “Toward Supporting Stories with Procedurally Generated Game Worlds”. Georgia Institute of Technology.

[17] Shaker, Noor, Julian Togelius, and Mark J. Nelson. Procedural content generation in games. Switzerland: Springer International Publishing, 2016.

Appendix A

	Asthma	COPD	Chronic Bronchitis	Emphysema	Lung Cancer	Cystic Fibrosis	Pneumonia	Plueral Effusion	COVID-19
Coughing	x	x	x	x	x	x	x	x	
Wheezing	x	x	x	x	x	x			
Shortness of Breath	x	x	x	x	x	x	x	x	x
Chest Tightness	x		x	x				x	
Difficulty Breathing	x								
Excess Plegm		x							
Trouble with Deep Breath		x							
Whistling / Sqeaky Breathing			x	x					
Chest Pain					x		x		
Coughing up Blood					x				
Fatigue					x				x
Unexplained Weight Loss					x				
Salty Skin						x			
Poor Growth						x			
Difficult Bowel Movement						x			
Male Infertility						x			
Frequent Lung infections						x			
Fever							x		x
Chills							x		x
Nausea / Vomitting							x		x
Inability to lie flat								x	
Inability to exercise								x	
Diarrhea							x		x
Generally feeling unwell								x	
Muscle Aches									x
Loss of Taste									x
Sore Throat									x
Congested / runny nose									x
Headache									x

Symptom Chart

Cold: Really common, children get this 6 - 10 times a year, adults get 2 - 3 a year. The older you get the fewer colds you get. Curable. More info can be found here.

- Fever or chills (although most people with colds do not develop a fever)
- Cough.
- Sore throat.
- Congestion or runny nose.
- Sneezing.
- Post-nasal drip.
- Watery eyes.

Flu: ~ 5% to 20% get this every year. Curable. More info can be found here.

- Fever or feeling feverish/chills
- Cough
- Shortness of breath or difficulty breathing
- Fatigue (tiredness)
- Sore throat
- Runny or stuffy nose
- Muscle pain or body aches

• Headache
• Some people may have vomiting and diarrhea, though this is more common in children than adults

Asthma: 1 in 13 people have asthma, there are different types. Some asthma goes away as one gets older, some only appear when one gets older. Can go away and can be permanent. More info can be found here.
• Wheezing
• Cough
• Shortness of breath or difficulty breathing
• Chest tightness
• Trouble sleeping caused by shortness of breath, coughing, and wheezing.
• All symptoms can be worsened by smoking, exercise, and/or respiratory ailments.
• Can also be situational, i.e. allergies, exercise, or occupation can cause it.

COPD: Chronic obstructive pulmonary disease, has 4 stages: Mild, Moderate, Severe, Very Severe. The prevalence of COPD in our adult population (≥45 years) was 3.6% and it increased greatly with age (1.9% in the age group 45–64, 4.8% in 65–74, 6.8% in 75–84 and 5.6% in ≥85 years). COPD appeared more common in males than in females (4.1% in males and 3.1% in females. Permanent condition. More info can be found here.
• Frequent coughing or wheezing
• Excess phlegm or sputum
• Shortness of breath
• Trouble taking a deep breath

Chronic Bronchitis: ~0.27 % have this (8.9 mill out of 323 mill people in US 2016). Is a form of COPD. Permanent condition. More info can be found here.
• Cough, often called smoker's cough
• Coughing up mucus (expectoration)
• Wheezing
• Chest discomfort

Emphysema: ~ 0.00959752321% people above 45 get this. Is a form of COPD. Permanent condition. More info can be found here.
• Frequent coughing or wheezing.
• A cough that produces a lot of mucus.
• Shortness of breath, especially with physical activity
• A whistling or squeaky sound when you breathe
• Tightness in your chest

Lung Cancer: ~ 0.0016749226% people above 45 get this, 15% of smokers get this. Curable. More info can be found here.
• A new cough that doesn't go away.
• Coughing up blood, even a small amount.
• Shortness of breath.
• Chest pain.
• Hoarseness.
• Losing weight without trying.
• Bone pain.
• Headache.

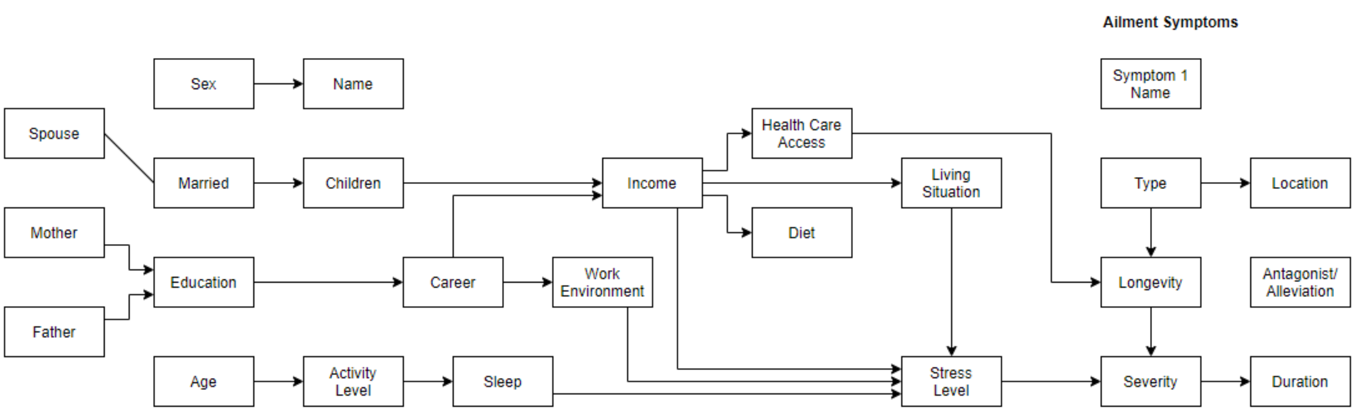
Cystic Fibrosis: ~ 0.00009287925% people above 2 get this, it is usually diagnosed at a very young age. Permanent condition. More info can be found here.
• Very salty-tasting skin.
• Persistent coughing, at times with phlegm.
• Frequent lung infections including pneumonia or bronchitis.
• Wheezing or shortness of breath.
• Poor growth or weight gain in spite of a good appetite.
• Frequent greasy, bulky stools or difficulty with bowel movements.
• Male infertility.

Pneumonia: ~0.0040247678% people any age get this. Curable. More info can be found here.
• Cough, which may produce greenish, yellow or even bloody mucus.
• Fever, sweating and shaking chills.
• Shortness of breath.
• Rapid, shallow breathing.
• Sharp or stabbing chest pain that gets worse when you breathe deeply or cough.
• Loss of appetite, low energy, and fatigue.

Pleural Effusion: ~0.00309597523% people any age get this. Common among folk with COPD. Can be caused by lung cancer. Curable. More info can be found here.
• Shortness of breath.
• Dry cough.
• Pain.
• Feeling of chest heaviness or tightness.
• Inability to lie flat.
• Inability to exercise.
• Generally feeling unwell.

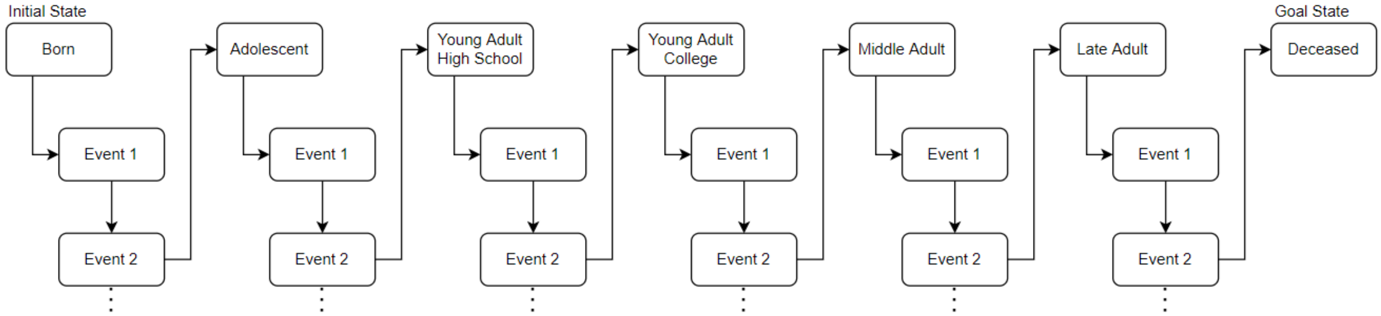
COVID-19: ~0.00104334365% people get this. Curable. More info can be found here.
• Fever or chills
• Cough
• Shortness of breath or difficulty breathing
• Fatigue
• Muscle or body aches
• Headache
• New loss of taste or smell
• Sore throat
• Congestion or runny nose
• Nausea or vomiting
• Diarrhea
• (less common and more serious) Persistent pain or pressure in the chest
• (less common and more serious) New confusion
• (less common and more serious) Inability to wake or stay awake
• (less common and more serious) Pale, gray, or blue-colored skin, lips, or nail beds, depending on skin tone

Patient Data Relational Model



Life Stages of Patient - Event Chart

Life stage progression is always the last event triggered.

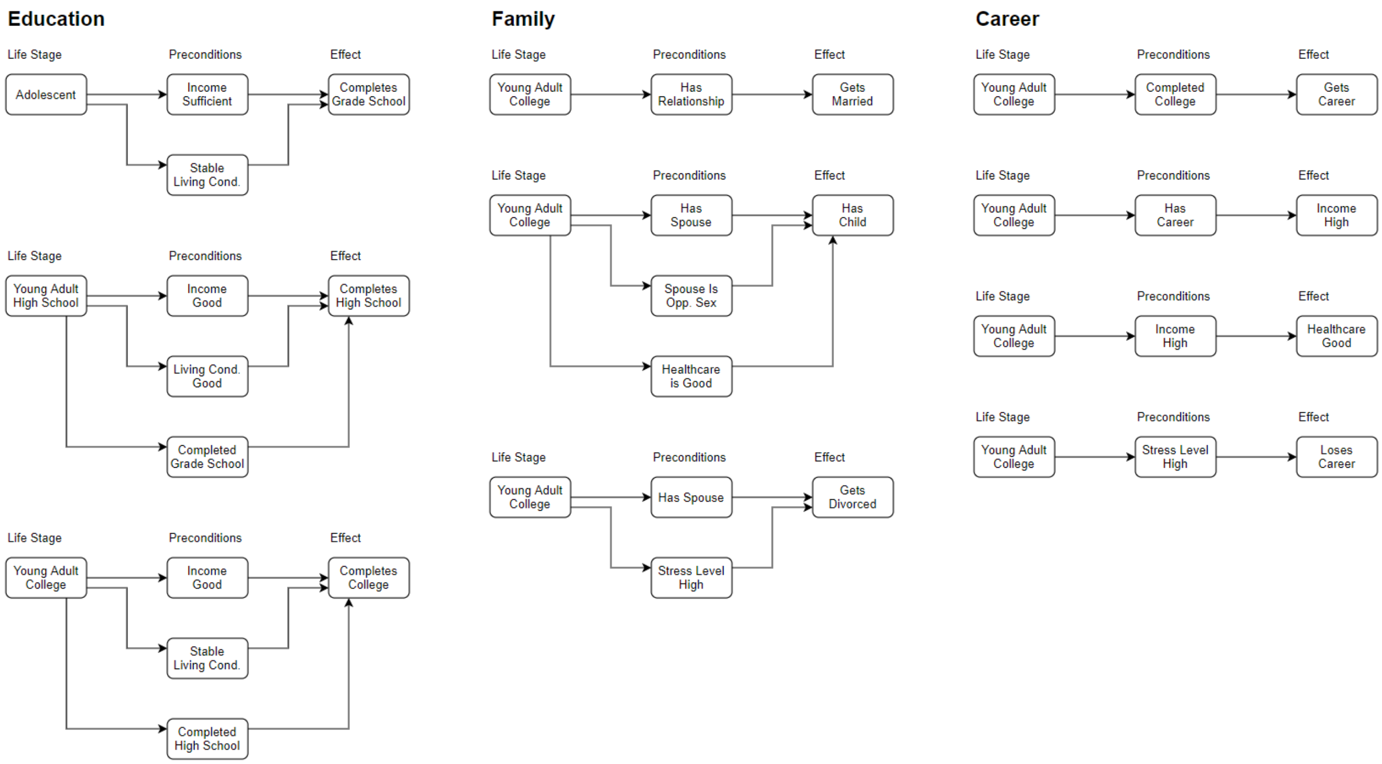


Events come in two forms: Consequential and Random.

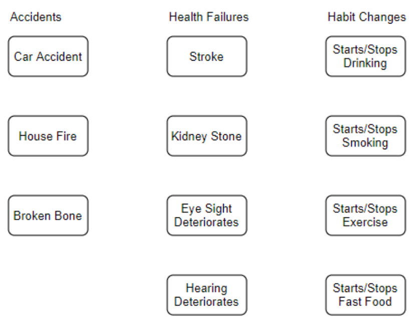
The number of consequential events is determined by the state of the patient at each iteration (during cyclical checks). The number of Random events is assigned at the beginning of each life stage. A life stage can transition to the next level only when no events remain to be processed.

Events will have an associated category. After a patient has Been generated from birth to death, a list of all life events can be accessed. From that list, an ailment can be selected. In this fashion, an ailment will be presented in the context of a patient's life at the time the event occurred.

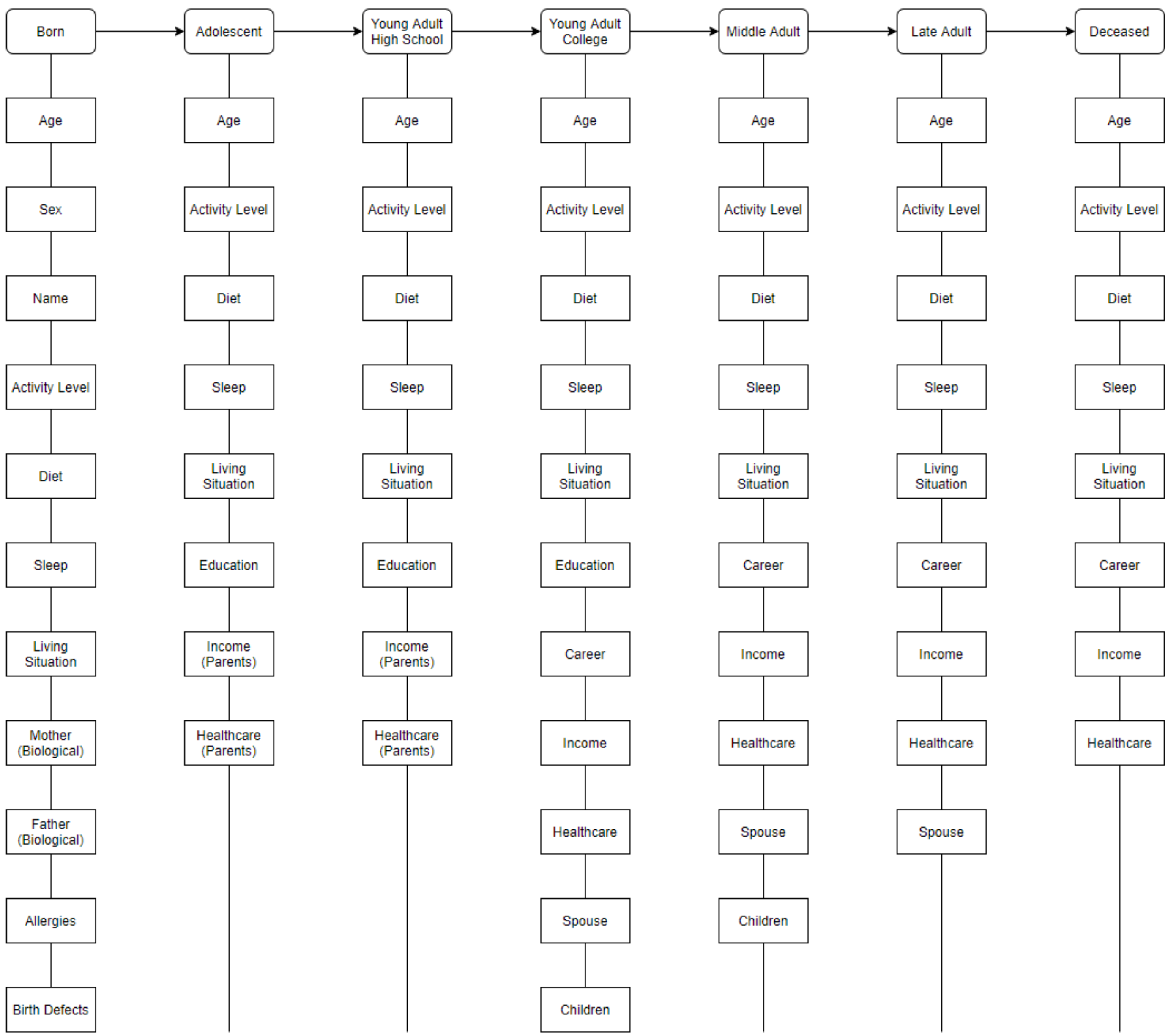
Consequential Event Examples



Random Event Examples



Life Stages of Patient: Listing data that may change in each stage



Appendix B

Common Cold
- Cough
- Mucus

Ear Infection
- Ear Pain
- Fever

Flu
- Difficulty breathing
- Body Aches / Chest Pain
- Fever
- Cough
- Chills
- Sore Throat
- Runny/Stuffy Nose

Sinus Infection
- Headache
- Runny/Stuffy Nose
- Face Pain

Skin Infection?
- Skin Redness
- Skin swelling around infected area

Sore Throat
- Throat Pain
- Hoarse Cough

Complex ones
- Asthma
    - Having a parent with Asthma
    - Sensitivity to irritants and allergens
    - Past respiratory infections
    - Overweight
    - Symptoms:
        - Coughing
        - Wheezing
        - Shortness of Breath
        - Rapid Breathing
        - Chest Tightness
- COPD (Chronic Obstructive Pulmonary Disease)
    - Symptoms:
        - Breathless-ness / Inability to exhale normally
        - Cough up mucus from lungs
    - Exposure to cigarette smoke
    - "Other Environmental Exposures"
- Chronic Bronchitis
    - Form of COPD w/ a very hoarse cough
    - Symptoms:
        - Hoarse Cough
        - Cough up mucus
- Acute Bronchitis
    - Form of COPD w/ a very "wet" cough
    - Infectious -> comes from a viral / bacterial infection -> **Out of Scope?**
- Emphysema
    - Most common cause is smoking
    - Problems exhaling air from lungs
- Lung Cancer
    - Too Difficult to detect without a scan / test -> **Out of Scope?**
- Cystic Fibrosis
    - Genetic caused by a defective gene that creates thick/sticky mucus that clogs up tubes and airways.
    - Symptoms:
        - Salty skin
        - Cough
        - Frequent Lung Infections
- Pneumonia
    - Common lung disease
    - Can be viral, bacterial, or fungal - **Out of Scope?**
    - Symptoms:
        - Cough
        - Fever
        - Chills
        - Shortness of breath

JEREMY SECTION A:

The user will first look at what information is currently known about the patient such as medical history and why they are seeking medical attention. Interact with the generated patient via a dialogue system that will allow the user to interview the patient and ask questions. The user will use all this info to try and properly diagnose a patient.

SCOPE:

Primarily Respiratory (minimal Cardiac)
- Basic first aid: bleeding, burns, sprains, breaks, shock
- Basic CPR: cardiac arrest, respiratory arrest

- Other areas: Viral, Neuro, ...
- Minimal scene focus
- Minutiea of treatment is left for VR

We plan on using a model and ontology based off these two diagrams:

A model in Computer Science is a way to describe something in a virtual environment. For example, ailment has certain properties such as symptoms. An ontology more of how things are related to one another for example both a cold and the flu can have the symptom cough and sore throat. We could also make an ontology like above for a patient's history, showing how certain things relate to and affect one another in a patient's medical history. The reason we need these models and ontologies is because we need to understand how things relate and how to classify things in regards to ailments and a patients properties like medical history so a patient can be useful in training.

A: Diagnostic Patient Model (Codify the data we'll need to generate a patient)

- Background / Theory
  - What is a model?
    - A model is a way to describe something in a virtual environment. For example a cat has certain properties such as a cat likes to meow, it has four legs, it likes to catch mice.
  - Ontology - concepts and relationships
    - How we classify things, and how things are related for example a dog is a mammal and a dog interacts with a cat by chasing the cat.
- How does it relate to our work?
  - We need an understanding of what comes together to make an interesting patient that makes for a good training scenario. We need to know how things relate and how to classify things so a patient can be useful in training.

- Classifications
  - We are focusing on respiratory ailments. Is this what we are talking about in regards to classification?
- Optional / Required
  - Dunno what to touch in regards to this?
  - Optional and required patient properties.
- Domain of values
  - Ians chart is the idea we are going for, we want this fairly small initially to be able to test, then expand upon this later.
- Probability variances
  - How probable an ailment symptom is to show, how likely an ailment itself is to show such as a person having cancer is less likely than a person having a cold.

B: Procedural Content Generation Algorithm (What is it, how does it relate, what will we use?)

- Background / Theory
  - What is PCG?
    - Procedural Content Generation is the generation of content through algorithmic means through the lens of a specific model. One of the biggest advantages of PCG is that the developers are in control of how content is generated, but not necessarily what is generated.
  - How is it different from Random?
    - Random generation involves creating content by randomly assigning properties based off of some probability that is independent of any other property. This type of content generation puts the control in the hands of the algorithm to create an item with randomly assigned properties.
  - Why do we need it?
    - Uniqueness
    - Believability
    - Predictability
    - Need to simulate a real world scenario of a patient entering a hospital with a set of symptoms and an ailment the user needs to diagnose.
    -

- Conway's Game of Life
- Planning
  - Best defined in the statement from the PCG book: "A planning algorithm finds a path from an initial state to a goal state; the sequence of actions that constitute this path can then be interpreted as a story"
  - CONAN
- L-systems

- Typed up a goog doc explaining L-Systems : https://docs.google.com/document/d/1ULJEocdmBp78HrwqgcB2y9WnrXzi6DgTr4donP0ISoA/edit?usp=sharing
- Caves of Qud
- Others?

Item generation [14]

For an artificial patient to be convincing, their history must be complex and subsequently generated patients must have enough variation to avoid becoming predictable. While we could turn to randomization to produce this content quickly and efficiently, this approach could result in the patient's story being difficult to rationalize in a way that would make sense to the user. As a result, procedural generation, or algorithmic generation based on established rules, is necessary. Producing a patient's complexity and variation algorithmically, it is no simple task, however. And there are many types of procedural generation to consider.

- Item Generation

Generating items involves creating a set of rules to guide the procedural generation of items.The first step in the item generation paper is determining if an item is generated, which in our case will always be yes.

L-SYSTEMS:

An L-system is a type of formal grammar that consists of an alphabet of symbols that can be used to make strings, a collection of production rules that can be used to expand each symbol into some larger string of symbols, an initial "axiom" string from which to begin construction, and a mechanism for translating the generated strings into geometric structures. L-Systems are often used to procedurally generate plants.

FUNCTIONAL L-SYSTEMS:

Functional L-Systems are very similar to a standard L-System. Where a functional L-System differs instead of having a normal alphabet and rules consisting of symbols or characters the alphabet and rules can contain functions, variables and conditions. This means that functional L-Systems are much more powerful and flexible than a standard L-System.

C: Prototype (Explain what we'll be building to prove our idea)

- Virtual Environment
  - Will be made in Unity.
  - 3D Clinic Environment:

    https://assetstore.unity.com/packages/3d/environments/doctor-s-office-165327
- Dialog System
  - Will use yarn spinner to implement.
  - Player can interview patient to get info
- Interaction System
  - Player can look at things to see if there are any obvious physical things that can be used to diagnose. The interactions will mostly be through dialogue.
- BACKLOG: Authoring Tool?
  - Might pull this out of the backlog as it might make our jobs easier to be able to quickly add things for testing. Would also give end users the ability to add specific things to test on.

JEREMY CONCEPTUAL DESIGN:

We plan on utilizing PCG to generate a believable patient. How we plan on doing this is via an algorithm that will start at a patient's parents, it will generate their living conditions such as social class, job and more. These will affect things such as a patient's mental health, their relationships, past injuries, some of the medical history and lifespan. Some factors of a patient's parents may be random such as parts of their medical history hereditary conditions. Our algorithm will also take into account that some properties may be optional, for example not everyone has access to parent medical history, that however does not mean it does not generate parents to make a patient from, it just means that the end user will not have access to the parent data. These factors from the parents will then be used to generate a patient and their ailment(s). Not everything will come from a parent such as social status and job, however that can be influenced by a patient's parents.

Ian’s proposed content for the Approach section:
Summary of Medical Education (to become doctor):

- Attain Bachelor’s Degree
- Medical School
    - Split into two halves
        1. First two years are comprised mostly of classroom work, learning the basics of anatomy, diseases, and body functions.
        2. The second half is comprised of clinical, hands-on patient work, usually in a teaching hospital or academic medical center.
- Residency
    - On-the-job training under the auspices of an attending physician. Length of time as resident depends on chosen field. Can be a little as a single year in some states to practice general medicine, to another six or seven years with some of the more complex specializations like neurology.
- Attending Physician
    - Receive your license to practice medicine unsupervised. Can seek Board certification.

Ian: Proposed persona categories:

Persona 1:

Age: 27

In her Second year of medical school. Will start working with patients directly in her next year of school. Would like a way to practice with patients before taking on the real thing.

Persona 2:

Age: 28

Has been working directly with patients since the beginning of the year, his third year in medical school. Would like a way to practice more complex and varied cases than what he’s seen so far.

Persona 3:

Age: 29

Nearing her residency. After two years of hands-on patient work at the academic medical center, she’d like a way to test her accuracy before starting her residency.

Journey (General):

Reads info on patient's reason for making appointment **- Intake Form**

- Past medical history
    - Medications
    - Surgeries
    - Family History
    - Social History
- Review of the systems of the patients history
    - Cough, Headaches, Body Aches, etc.
    - "30 to 40 questions on everything you've ever had"

Calls patient into office
Greets patient
Analyze the patients demeanor

- How do they act?
    - Walking slowly, Breathing heavily, Do they have trouble standing? …

Asks patient what brings them into the office
Patient Explains their issue

- This is the **Chief Complaint**
    - Acronym: **NLDOCAT**
    - Nature of the issue -> Burning? Throbbing? Dull? Acute? Sharp?
    - Location of the issue -> Where is it happening?
    - Duration of the issue -> How long has it been happening? How long does it occur?
    - Onset of the issue -> When did it start? How did it start?
    - Characteristics of the issue -> Similar to the nature of the issue
    - Antagonistic / Alleviations of the issue -> What makes it worse? What makes it better?
    - Treatment of the issue -> What have you (the patient) done so far?
- Patient may not directly inform the doctor, will require some Q&A between them

Doctor gathers patients vitals, looking for anything out of the ordinary

- Heart Rhythm
- Respiratory Rate
- Blood Pressure
- Temperature
- Skin
    - Rash?
    - Paleness?
- BMI? -> Not an indicator for much but has recently been included
- "Constitution of the Patient" Examples:
    - Weak Elderly Male who is unaware of his surroundings and is fatigued
    - Healthy Adult Female who is completely aware of her surroundings

Doctor considers and decides on the best course of action

- Differential Diagnosis -> What tests need to be run / What more information does the doctor need to confirm a diagnosis? May need to return to the examination phase
- If the doctor can say "The Patient definitely is suffering from <X>" continue forward

Doctor explains to the patient the intended treatment plan or necessary tests needed to determine their diagnosis
Patient asks questions about treatment plan
Doctor ensures all questions are answered
Patient leave the office

Ian’s go at building a patient model 01/18/21

Patient Info (All, from user journey):

- Name
- Age
- Height
- Weight
- Vitals
    - Heart rhythm
    - Respiratory rate
    - Blood Pressure
    - Temperature
- Medical History
    - Ailments (List)
    - Surgeries (List)
- Family History
    - Mother/Father
        - Living/Deceased Age
        - Ailment History
- Social History
    - Education level
    - Vision Problems
    - Hearing Problems
    - Living Situation
    - Smoking/Tobacco Use
    - Alcohol
    - Recreational Drug Use
    - Sexually Active
    - Issues at home/work
- Symptoms
    - Type
    - Intensity
    - Location
    - Start date
    - Duration if not constant
    - Antagonist/alleviation
    - Treatments applied
- Demeanor
    - Must provide a description of fitness, alertness, and energy level
- Dialog
    - Patient must be able to convey information about symptoms through dialog
- Other hidden patient fields:
    - Career (yes/no)
    - Income level (good/bad)
    - Diet (good/bad)
    - HealthCare Access (good/bad)
    - Married (bool)
    - Children (int)

Attempt to Conform to STRIPS Format

Conditions	Initial State	Goal State
LifeStage(Stage)	LifeStage(Born) ^ Sex(null) ^ Name(null)	LifeStage(Deceased)
Name(String)		
Income(Level)		
HealthCare(Level)		

Do we need operators for initial values?

Name(x)	Sex(x)
Preconditions: LifeStage(Born) ^ Name(null)	**Preconditions:** LifeStage(Born) ^ Sex(null)
Add List: Name(x)	**Add List:** Sex(x)
Remove List: Name(null)	**Remove List:** Sex(null)

Transition to next age level. Note sure if we can use number of events as a condition.

Age(x,y)
Preconditions: Age(x) ^ Events(0)
Add List: Age(y) ^ Events(rand)
Remove List: Age(x) ^ Events(0)

Ian's Mathematical Reports:

Reinfection Dampening: https://drive.google.com/file/d/1RkK9BQ62yWlcvqdG9CvllifStHEbWJUT/view?usp=sharing
Probability Distribution Simulation:
https://drive.google.com/file/d/1cwR9HBvCHXc-ieF415bdrwNv7eDwA56V/view?usp=sharing

Patient Domains :
https://docs.google.com/document/d/1xRO7Av3A3vvHlxJMy8NbSnYFIRR-RYnnPDz7NHq9BJU/edit?usp=sharing

What could work in generating the NPC's "story":

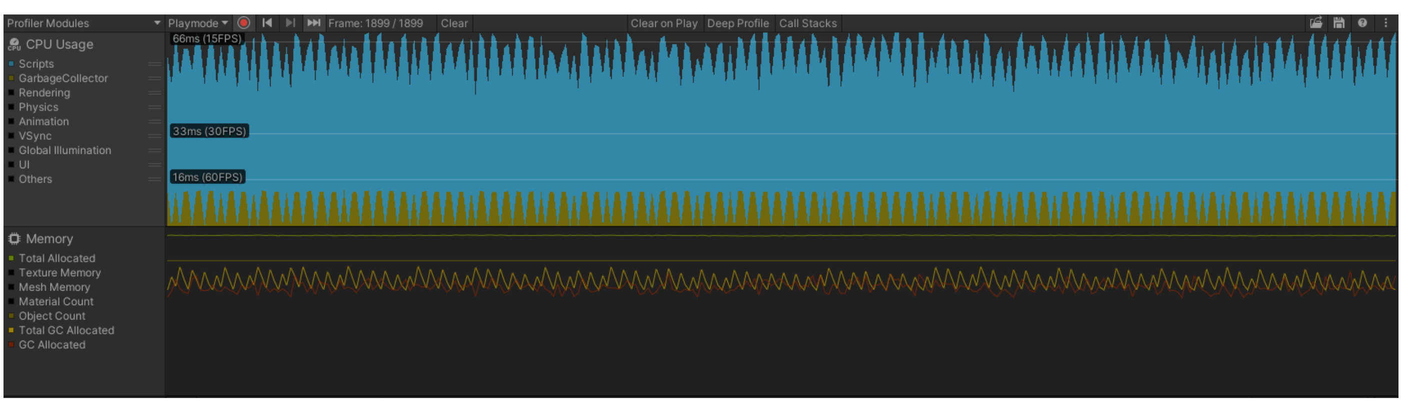
- Current Health Condition:
    - Physical health
        - Fitness
        - Weight or BMI
        - Diet
    - Mental Health
- Current symptoms
    - A list of symptoms that led the NPC to visit the doctor's office
- Symptom history
    - The lifetime of each of the patient's symptoms
    - Perceived correlation between symptoms (if more than one is present)
- Environment
    - Residence
    - Income
    - Occupation
    - Occupational environment
- Relations
    - NPC with its own attributes, some of which will be shared with the patient (residence, for example)
    - NPC's relation to patient
    - Quality of relationship with patient

A relation is the patient's family. For example, we can determine that the patient has a spouse and a child. Each NPC will share the same set of attributes as the patient. Then, for each adult, we can generate one or two NPCs to serve as their parents. From my experience with doctors, I'm assuming that parents will suffice. I don't suspect we'd need any further than grandparents at the most.

Jeremy's Performance Notes:

Hardware used:
CPU: Ryzen 5 3600
Graphics Card: Nvidia RTX 2060 Super
Ram: 16gb

With 1000 patients generated back to back:



-Took ~1 minute.

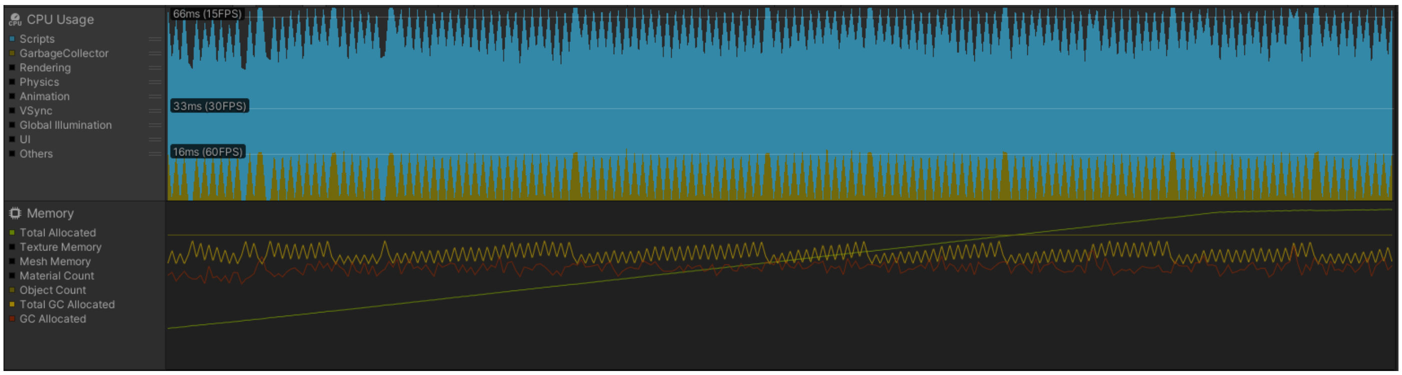
-Unity is maintaining an average of 15 fps

-There are no longer bottle necks from garbage collection.

-CPU calls take around 66ms.

-Ram usage is around 4.5 gigabytes.

With 500 patients generated back to back:



-Took about 30 seconds.

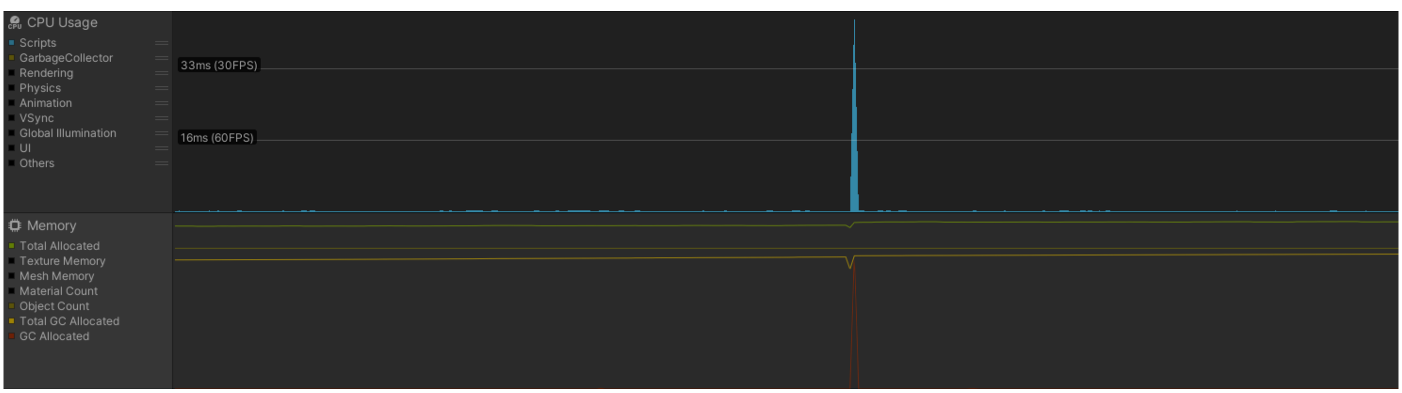
-Unity is maintaining an average of 15 fps

-CPU calls take around 66ms.

-Ram usage is around 4.5 gigabytes.

-Once so many patients are generated it hits a threshold where performance stays constant.

With 2 patients generated back to back:



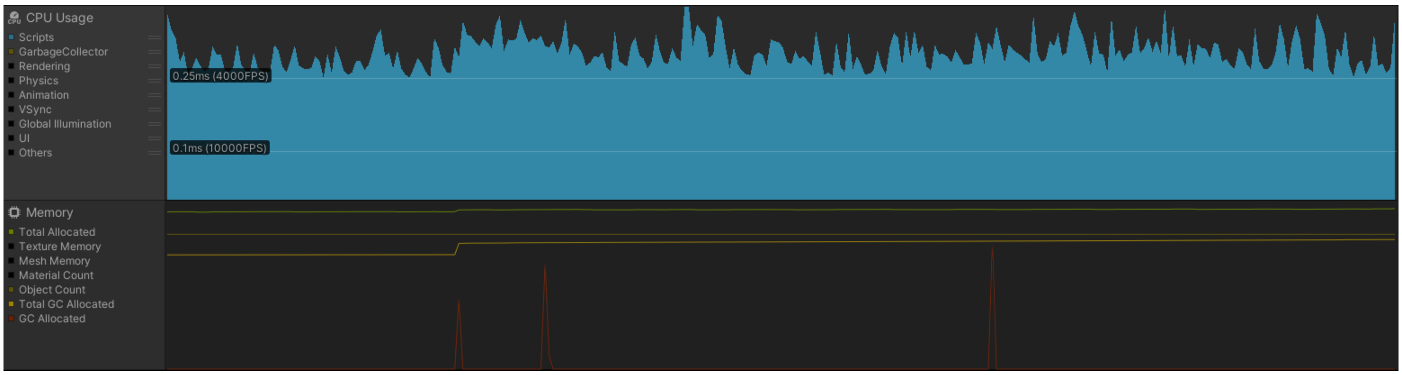
-Took about 1 and a half frames or ~45ms.

-Unity is maintaining an average of 30 fps

-CPU calls take around 47.45ms.

-Ram usage is around 0.59 gigabytes.

With 1 patient:



-Took less than 1 frame or ~0.25ms.

-Unity is maintaining an average of 4000 fps

-CPU calls take around 0.3ms.

-Ram usage is around 0.58 gigabytes.

Brief Notes since data is still be finalized:

- As a whole the data seems to be pretty diverse besides specific variables that are fixed by the PCGManager.
    - Age being the primary example is always the same value, and has a standard deviation of zero
- Some of the poor results
    - Masters & Doctorate degree's do not occur at all PERFORMED TUNING
    - Poverty income level does not occur at all SAME THING TUNING
    - In the career field, 8k out of 15k generated have the "entrepreneur" career, which seems disproportionate to the expected outcomes
- Some of the better results
    - Even spread amongst Smoking, Drinking, Exercise & Diet - ~20% split amongst the 5 possible values
- Some of the Ambiguous Results
    - Temperature has a std deviation of .4 which seems very low… however not problematic at face value HOW CAN WE TUNE THIS? CHANGE UPPER / LOWER THRESHOLDS
    - Number of medical incidents has an average of 25 and a std deviation of 2 - this std deviation seems low? Feels like it could potentially be higher…?
    - Marriages could have more variance… only ~1500 patients had more than 1 marriage