Generative Probabilistic Programming in Games: Creating Character Backgrounds Using a Bayesian Network

Ruiyuan Li
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
r.li-3@student.utwente.nl

ABSTRACT

Procedural content generation is a common facilitator in game design. Successful non-player characters (NPCs) backgrounds design can help with storytelling so that the players feel more involved and immersed in the game. This paper firstly discusses the definition of character background and how to select the related attributes based on archetypes in the template called monomyth (i.e., the hero's journey), then shows the way to formalize the characteristic design with the help of probabilistic grammars. Next, we realise the background generation via the Bayesian network and JS-divergence which serves as a metric for the performance evaluation. Finally, by updating conflicting rules and corresponding conditional probability tables in the Bayesian network, KL-divergence between the criteria and new observed distributions is minimized, which results in a local optimal realizing the improvement of the generator.

Keywords

Procedural content generation, probabilistic grammar, non-player characters, Bayesian network, KL-divergence

1. INTRODUCTION

Designing a game is always an exciting topic attracting people of all ages. We know that there are many aspects of a game [22]: levels, maps, game rules, characters, background stories, quests, music, etc. It is such a heavy workload for independent game studios or individual designers. They may hold wonderful ideas but have to struggle with the complex preparation to realize them. So the importance of Procedural Content Generation is reflected here. Julian Togelius et al [28] stated, that although the core game can exist without PCG, it can be added to facilitate the establishment or accelerate the runtime. The PCG methods embedded in the AI model can also inspire the creativity of the designer. Additionally, developing PCG methods aids the designers to understand the process and in turn, betters the game.

Probabilistic grammars are capable of expressing the notion of structural correctness in a fine-grained way. While "normal" (non-probabilistic) grammars in a broad sense are used to express programming languages, metamodels, document structures [34], XML and JSON formats [16] and communication protocols, probabilistic ones are used to predict DNA sequences [26], handle uncertain and unreliable data, automate reasoning in healthcare [17]. It stands to reason that they can also handle game content well [30].

This project focuses on using probabilistic rules to generate characters with their backgrounds through a Bayesian network [25]. The first task is to design the game world and determine the usage of the characters. In this research, the characters have a background of the monomyth and serve as Non-player Characters (NPCs) which are used to enrich the storytelling of the game to attract players. The narrative generation is not considered here, and only attributes set implying certain elements of plots in monomyth will be designed. The characteristics or attributes are neither complicated in type nor numerous, but they are all based on a certain archetype and make sure the character act like a human, which is regarded important by Warpefelt [32].

According to Vogler [31], archetypes are "flexible character functions" instead of "rigid character roles", which are carried out for certain effect in a story. This means the same character can conform to various archetypes along with a story. Archetypes in monomyth are suitable for this research because the pattern complies with thousands of *Hero stories* across different ages and cultures. Besides applying it in the literature domain, it also plays important role in plot design for video games like Mass Effect and Dragon Age [15]. This pattern provides players with immersion by relatively convincing storytelling at a low price, because similar stories have been told in various genres like literature, music, theatre, and art thousands of times before. Five main archetypes in monomyth are taken here.

Character design is as not complicated as plot design, but the network of characteristics still implies the causal chain and timeline. For example, the highest education level takes 15 or more years to achieve, and the same character cannot be very young (12 to 18 years old). Pure randomness is infeasible here as "individuals are poor at behaving randomly"[33], because the actions are taken based on specific mechanisms to make the game world reasonable. The importance of using "limited randomness" pushes the designers to consider carefully the attributes selection and the postcondition of certain values.

After the attributes design is over, the rules have to be represented formally using probabilistic grammar introduced in Section 3.2, which contains the preconditions, commands to be executed, and the probabilistic parameters. The ruleset will be used as the reference for Bayesian network generation afterward.

The Bayesian network facilitates easier understanding for people, demonstrating the structure of designed characteristics and organizing the effects with conditional probability tables [25]. The performance can be evaluated by Jensen-Shannon (JS) divergence and Kullback–Leibler (KL) divergence after selecting criteria. JS-divergence serves as the metric measuring the distance between the chosen criteria and the observed distribution. Once it is too large to accept, the conflicting rules will be found and their conditional probability tables in the Bayesian network will be updated to minimize KL-divergence instead of JS-divergence because it is more sensitive to minor changes. Rules could be updated, deleted, or added, which realizes the improvement of the ruleset and satisfies the requirements better. The process of evaluation restarts again from the criteria selection until the divergence is acceptable for the designers.

2. RELATED WORK

There is much research done on this Procedural Content Generation (PCG) in games [22], and [23] designs an analytical framework and associated common vocabulary which betters understandings of PCG's role in games, concerning from both technical and design side. This paper also discusses available PCG tools and worked examples. There are various kinds of contents included in PCG, like vegetation[29],levels[7, 24, 2], landscape[21, 6, 8], quests[1] and rules[18]. Besides designing a specific type of contents, [12] focuses on integrating mission and space into level generation to deal with the problems of synergy between objects and levels.

Work contributing to non-player character generation is also available. Narratives that exist in the interaction with NPCs can enhance the storytelling of a game, and [5] proves that by giving an overview of the underlying structures to promote the narratives and analyzing how narratives can support problem-solving in the game. In 2019, Erica Jurado et al. [11] proposes a system named Cast Affinity Satisfiability Toolkit (CAST) using answer set programming (ASP) to unburden the cost in time when the authors increase the interactivity of non-player characters(NPCs) in story worlds. They focus on designing personality and social connections between characters.[20] presents a coevolutionary framework for NPCs generation in massively multiplayer online real-time strategy (MMORTS) games. The research contributes a new model to generate thousands of solutions developed by players and balance the characteristics of NPCs.

Concerning the usage of the monomyth in PCG, it can be found that Ruben et al. studies massive PCG generating modeling monomyth and proved evidence that their multi-agent systems realize the emergence of the monomyth-based virtual world [9]. The evaluation of the results is also researched. Lucas et al. [14] use Kullback–Leibler divergence as a way to compare and analyze levels, and they use tile patterns of mixed sizes as features, which inspires the evaluation methods in this paper.

3. METHODOLOGY

In this section, the methods used to realize procedural content generation for character background will be introduced. The workflow for generating characters through Bayesian Network is shown in Figure 1. The main methodologies are determining the definition of background, making formal ruleset, building a Bayesian network and evaluating performance by KL-divergence and JS-divergence.

3.1 The Definition of Background

There are many definitions for the background of a character. This project doesn't discuss the narratives, so it means the generator doesn't create any natural language stories but is a tool to generate a set of attributes invented for a plot, and some of the attributes have either obvious or obscure relationships with the others. While backstory, in this paper, means the narratives used to lead players into the main plot. The backstory

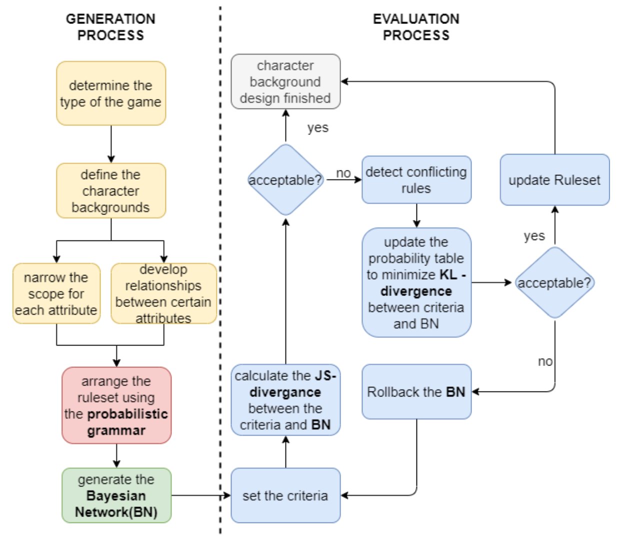

GENERATION PROCESS
EVALUATION PROCESS
determine the type of the game
define the character backgrounds
narrow the scope for each attribute
develop relationships between certain attributes
arrange the ruleset using the probabilistic grammar
generate the Bayesian Network(BN)
set the criteria
calculate the JS-divergence between the criteria and BN
acceptable?
yes
no
character background design finished
detect conflicting rules
update the probability table to minimize KL - divergence between criteria and BN
acceptable?
yes
no
update Ruleset
Rollback the BN

Figure 1. Workflow of Creating Character Backgrounds. There are four stages in the character backgrounds design: the definition of background(yellow), the formulation of ruleset(red), the initialization of Bayesian network(green) and the evaluation and improvement(blue).

can be generated based on elements like characteristics of the role, related objects, place, event in the background. That can be discussed in the future.

Since there are a lot of types of attributes to chose for a character (age, profession, education, etc.), the selection is extremely crucial and regarded as the basement of rigorous and effective procedural content generation afterward. The preparation for formal rules design at the next step includes game theme design, character usage design, main attributes selection, bridging attributes supplement.

Initially, the game theme and the usage of the generated characters should be decided. The theme of a game can be the core support point for all other components in the design of the game, and without a specific background setting, the rules, and elements in the game may contradict one another or seem inappropriate.

Moreover, the usage of character will at least decide how far the designer should go in certain directions. For instance, the characters for players may have more ways to take action and their professions, skills and education level are not fixed because these features have to be developed by the players themselves. However, the NPC helping with storytelling in a game should have a brief but relatively complete background design.

For example, the game could be an adventure videogame, while the roles inside are characters who are likely to exist in a monomyth which is "the minimum set of archetypes" fulfilling the requirement interesting storytelling mentioned by García-Ortega et al [9]. The characters will serve as the non-player characters (NPCs) in the project.

As the next step, appropriate attributes should be selected. Making designed/generated NPCs behave as similarly as possible to humans is a part of our mission. The selected attributes are expected to be the basis of backstories that are internally consistent and engaging the players. Thus, they should involve elements among all three dimensions: physiology, sociology, and psychology in the bone structure defined by Lankoski et al [19]. Physiological attribute "age", social attributes "skill", "profession", "education", while "archetype" implies psychological information. To add in some events enriching the background, "adventure", "place", "tool event" and "profession" are included.

Last but not least, to facilitate the formal rule design, some nodes realizing the bridging work between two characteristics are requisites. So "tool power" and "profession level" are attached.

The bridging nodes are invisible in the data structure of the generated characters as the output, so all the visible characteristics are shown by an array-like plot similar to this one:

1. archetype	2. age	3. adventure	4. education
5. skill	6. place	7. tool class	8. profession

3.2 Probabilistic Grammars

We designed a probabilistic grammar here to clarify the distribution of the attributes and formalize relationships between them. This notation is inspired by Hoare Triple [10], which provides a rigorous way to formalize the logic system. Unlike the Hoare triple, this grammar focuses on the possible commands and their preconditions.

The probabilistic grammar is of the form

$$Q \rightarrow [p]C$$

To be specific, $\rightarrow$ indicates that the left part is the precondition and the right is commands. Q and C denote the precondition and command. $[p]$ denotes the probability of certain commands, and there can be two or more $[p]C$ in one formula. Because even under the same preconditions, there could be more than one possible command with their probabilities. For example,

$archetype = "Hero" \rightarrow$
$profession = [0.6]"fisherman"[0.4]"hunter"$

is one rule in this project. $archetype = "Hero"$ is the precondition, and [0.6], [0.4] respectively showing that the probability of being a "fisherman" for a "Hero" is 0.6 and "hunter" 0.4.

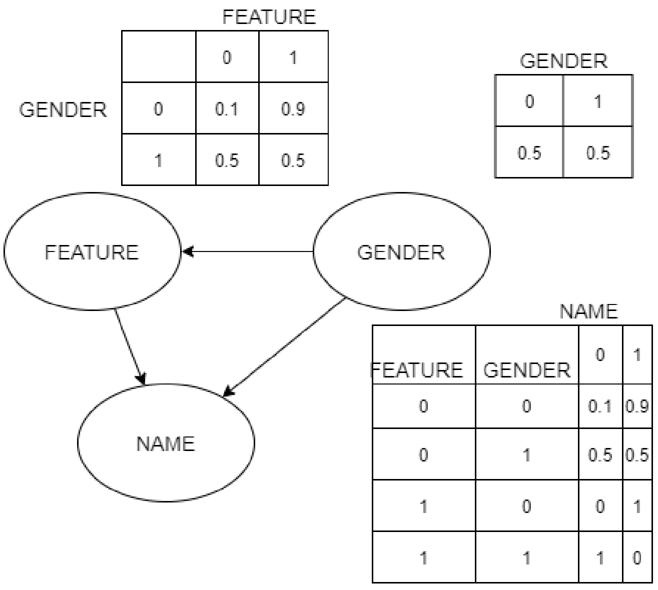
3.3 Bayesian Network

Bayesian Network is a directed acyclic graph representing a series of variables and their conditional dependencies. It is very useful to predict the outcome given existing knowledge of some observable quantities realizing by conditional probability tables. The observation of the probability of a certain situation combined with the value of the node and its parents' nodes serve as the input of a probabilistic distribution function which finally can calculate the unknown possibility of node A given B. Bayesian network depicts direct dependencies instead of complete joint distribution remaining an advantage because it is intuitively less challenging for a human to understand the whole things.

This feature is also known as the chain rule of probability. For example, the designer wants to generate some names for the characters. It is well-known that there are female names, male names, and unisex names. Given a name, the probabilities of its owner's gender are different. At the same time, the feature is sometimes shown in the name, like the name "Scarhead". What's more, gender also has an impact on one's features. So this situation can be modeled using a Bayesian network shown in Figure 2. For the sake of simplicity of this example, there are only two possible values for each variable which are 0 and 1.

$$P(N,F,G) = P(N \mid F,G)P(F \mid G)P(G)$$

is the joint probability function applying the chain rule where G, F, N are the values of the variables GENDER, FEATURE and NAME. The Bayesian network can answer questions about the probability given the inverse probability which is, for example, "If the name of the character is x, what is the probability of the gender is male?". This means, the designers can choose freely about the span of attributes having fixed values, and it is also possible to give certain effects and check if the probability distribution coincides with the design.



	FEATURE 0	FEATURE 1
GENDER 0	0.1	0.9
GENDER 1	0.5	0.5

GENDER 0	GENDER 1
0.5	0.5

FEATURE	GENDER	NAME 0	NAME 1
0	0	0.1	0.9
0	1	0.5	0.5
1	0	0	1
1	1	1	0

Figure 2. A simple Bayesian network with conditional probability tables

Bayesian Networks have two components: structure and probability, generated from the ruleset. In this case, the structure is clear from the relationship between the attributes defined in previous steps. But part of the probabilities is unavailable due to the size of the ruleset. Besides, some rules conflict with each other and the mistake is hard to be discovered and corrected until the Bayesian Network has been established. So some assumption to the probabilities is required in this stage, which may cause the divergence between the requirement and practice.

3.4 Kullback–Leibler divergence and Jensen-Shannon divergence

Kullback-Leibler divergence (KL-divergence) is a way to measure the differences between two distributions. It achieves the measurement by characterizing relative Shannon entropy. If two distribution are the same if KL-divergence is 0, while they are more dissimilar with each other when KL-divergence increases. The Shannon entropy is a concept in information theory domain which can quantify inherent information. It can be calculated as

$$\mathrm{H}(X) = -\sum_{i=1}^{n} \mathrm{P}(x_i)\log \mathrm{P}(x_i)$$

while x_i denotes the possible outcome occurring with $P(x_i)$, the probability. n is the size of sample space. KL-divergence has a similar shape of the entropy but it aims to accomplish the comparison between an observation and the model, the theory, which is the requirement implied by the ruleset. It can be written as:

$$\mathrm{D_{kl}}(\mathrm{P}||\mathrm{Q}) = -\sum_{i=1}^{n} \mathrm{P}(x_i)\log\frac{\mathrm{P}(x_i)}{\mathrm{Q}(x_i)},$$

P is usually used to represent practical observation, and Q is used for the model, sometimes is deemed as the criteria. In statistics, P is the posterior probability distribution and Q is the prior probability distribution.

It is worth mentioning that KL-divergence is asymmetric, so

$D_{\mathrm{KL}}(P \parallel Q) \neq D_{\mathrm{KL}}(Q \parallel P)$. It has drawbacks especially in certain situation:

If one probability P(x_i) is very close to 0 and Q(x_i) > 0, then $D_{\mathrm{KL}}(P \parallel Q)$ shrinks to 0 which results in a misleading about identity. But we are going to change the distribution of P to minimize $D_{kl}(P||Q)$, so if x_i is very improbable to be draw from $q(x_i)$ then the same from $p(x_i)$. When $q(x_i)$ is near zero, $D_{kl}(P||Q)$ grows quickly while $p(x_i)$ increasing. There is a symmetric divergence here: Jensen-Shannon divergence. It can be calculated like:

$$D_{\mathrm{JS}}(P \parallel Q) = \frac{1}{2}D_{kl}(P \parallel M) + \frac{1}{2}D_{kl}(Q \parallel M)$$

where $M_i = \frac{1}{2}(P_i + Q_i)$. The value of JS-divergence is in the range $[0,1]$. When the divergence is large, it will collapse to 1.

It is very useful to have a metric if there is only one observation because a score from $[0,1]$ judges the result well. However, when it is used to minimize the divergence, then KL-divergence is better because the minor changes when JS-divergence close to 1 can be obvious using KL-divergence. That is the reason why JS-divergence aims to give a preliminary judgement of the Bayesian network and KL-divergence is used to measure the improvement of the model.

4. RESULT

4.1 Selection of Main Attributes

To begin with, the problem that should be dealt with here is the selection of archetypes. There are thousands of archetypes, “as many as there are human qualities to dramatize in stories” [31], so it’s not wise to include as many archetypes as possible in every game. Inspired by García-Ortega [9] and Tillman [27], five character types are chosen here: the Hero, the Shadow, the Trickster, the Fool, and the Prophet. Relying on Campbell [3], Vogler also summarized that a quest of a hero compromise to 12 stages in a monomyth. Inspired by his work, 5 stages are generated:

1. Be born.
2. Go to school.
3. Learn some skills.
4. Travel and get special tools.
5. Determine the profession.
6. Experience an adventure.

The hero and the shadow are representations of good and evil, the trickster can switch between them, and the fool always larks or ruins things to move the plot. And the prophet is the wise and mysterious role who knows secrets the others don’t know. They can be the mentor of the teacher sometimes.

The archetypes set the preference of choosing skills, education, places to travel and adventure events. There are five types of skills designed for each archetype but the character still has the chance to do things not match with their archetypes but of a lower probability. There are four types of education levels correspond to "primary", "moderate", "advanced" and "palace level/highest" education. Different places have different tools. For example, let’s assume a "Hero" is more likely to go to the "battlefield", because the character can get his/her own "weapon" there, and at the same time, "distant and peaceful island" is the right place seeking for mysterious book especially for a "Prophet". As for the adventure, it is designed as a bonus point and most of the characters only "lived in peace for many years" with no special adventure event happen. The content of adventure is also related to the archetypes, while each role has its inner motivation to participate in the story which can be called "the goal". In the work by Canheti et al [4], the Hero’s goal is "Make the world better" and the way to achieve that is killing its enemy, "the Shadow" so the adventure is "Was predicted by a witch to kill the shadow". By the way, if the tool matches with the skill, for instance, "weapon and hunt", then the value of toolpower is "useful", otherwise like "hammer and potion craft", the value will be "useless" with no doubt.

If all the values are available, it is time to consider the part of the profession assignment. The skills have their corresponding types of professions, since each archetype matches along with the main skill, and there are three levels of professions. So in the end character can have one of fifteen professions.

The values of each attributes are shown in Table 1 while they are represented by an index from 0.

attribute	index	value
age	0-3	"12-18", "19-25","25-60",">60"
education level	0-3	"primary", "moderate", "advanced", "palace level/highest"
skill	0-4	"potion craft", "hunt", "tool maker", "steal", "lurk"
tool event	0-2	"visit the battlefield", "visit the town", "visit the island"
tool power	0-1	"useless","useful"
tool power	0-2	"common","rare","priceless"
profession level	0-2	"low","medium","high"
profession	0-14	"Herb Collector", "Fisherman", "Repairer", "Thief(1)", "Spy", "Therapist", "deer Hunter", "Taylor", "Thief", "Assassin", "Alchemist", "Warrior", "weapon producer", "Scammer", "Dark Warrior"
adventure	0-4	"Was predicted to kill the shadow", "Was predicted to kill the Hero", "Fought with a Troll and got a scar on back", "Lived in peace for many years", "Dreamed of a distant place full of exotic flowers inlaid with gems."

Table 1. The possible values of the attributes

4.2 Formal Ruleset Design

In this section, the previous vague designs and concepts have to be transformed into formal expressions. The archetype and age, are the starts of the generator, and they are independent. As mentioned in subsection 4.1, the archetypes only represent a "preference", which means a character of a specific archetype has a most possible choice of skill but there are still chances of other results. To illustrate, a "Hero" is supposed to developing the skill of hunting, but he/she might "want" to learn how to "craft potions" or "make tools" instead. This ends up with the Appendix A *Rule 4.a*:

$$\begin{aligned} \textit{archetype} = \text{"Hero"} \rightarrow & [0.6]\,\textit{skill} = \text{"hunter"} \\ & [0.2]\,\textit{skill} = \text{"tool maker"} \\ & [0.2]\,\textit{skill} = \text{"potion craft"} \end{aligned}$$

In the myth, the shadow always has similar experiences as the hero, but one stands for evil and one for good, so the rules of "the Shadow" and "the Hero" are symmetrical. The other features like "Prophet" tend to obtain higher education and more likely to adhere to "potion craft" than other archetypes to their most probable skill(Appendix A *Rule 4*).

The skills have their corresponding types of professions, while each archetype matches along with the main skill. The education level and the worth of the tool determine one of three profession levels. The higher education the character had received, the more likely he/she does a difficult and rare job. A useful tool, of course, contributes to getting a better job.

Besides rules of each characteristic, Appendix A *Rule 10* is the requirement that doesn't influence the initialize of the Bayesian network but serves as criteria in the future. This is the general but core aim which can't be realized directly but only achieved through a finer level of steps.

The complete ruleset can be found in Appendix A.

4.3 Using Bayesian network for Character Generation

After the ruleset has been designed, the relationship between attributes and the probabilities will be input into the conditional probability tables. The structure of the network is shown in Figure 3. The nodes archetype and age are independent, showing the discrete distribution.

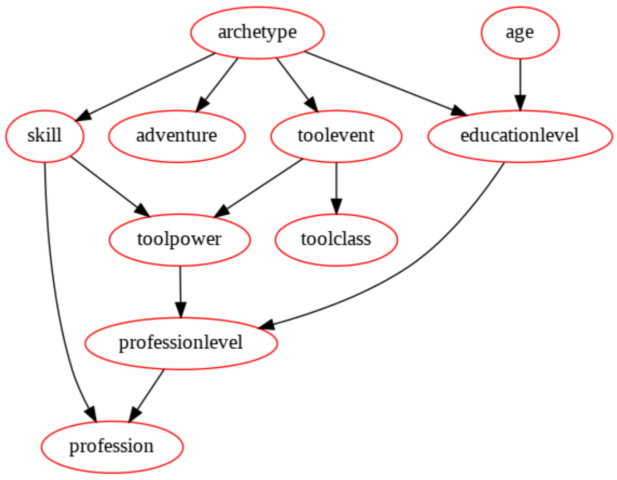


Figure 3. Bayesian Network of the character's background

In this project, Python is used to realise the generation. Give certain values of attributes, or leave everything empty, then the generator will return with the most likely result with the probability. The outcome of the character generation function is a JSON class object which facilitates alternating and reusing. The output JSON file looks like:

```
{
  'archetype': 'Prophet',
  'age': '>60',
  'adventure': 'lived in peace for many years',
  'education level': 'palace level/highest',
  'skill': 'potion craft',
  'traveled place': 'island',
  'tool class': 0,
  'profession': 'Alchemist'
}
```

Only the essential characteristics will be shown to the players, the values of bridging nodes like "profession level" is hidden from the outside.

On the other hand, given some values of the characters, the generator will return the probability of it. For example, given

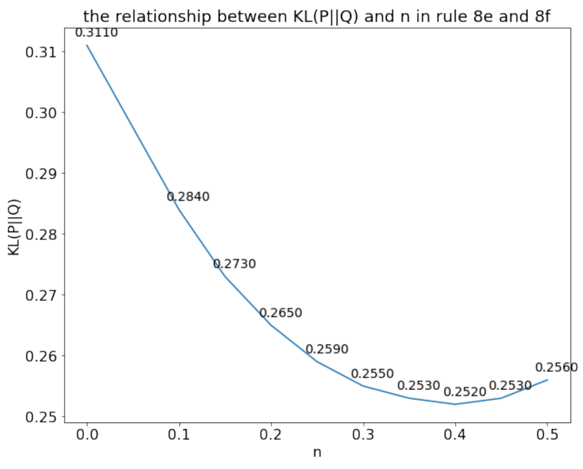


Figure 4. P denotes the observed distribution, and Q denotes the theoretical distribution according to Appendix A **Rule 2a-2e, 10a** here. KL(P||Q) is a function of n, as n changed(in Equation 4.4, etc.), part of the conditional probability table which should be updated, which results in different observation of the profession distribution in this case. As n increases from 0 to 0.5, KL(P||Q) decreases rapidly at first and slowly approaches the minimum, then increases.

the archetypes, the distribution of the possible professions is shown in Figure 5. Each subplot displays the probability of the archetypes ending up entering certain types of professions. The red dotted line represents the professions of the lowest level, the green line for medium level and blue for the highest. The distribution manifests the result of specific rules designed in section 4.2. Part of the third subplot, e.g., is the consequence of the rules having reference to the archetype "fool", which includes

$$\begin{aligned} archetype = \text{"Fool"} \rightarrow & [0.8]\, educationLevel = 0 \\ & [0.15]\, educationLevel = 1 \\ & [0.05]\, educationLevel = 2 \\ & [0]\, educationLevel = 3 \end{aligned}$$

Most "fools" only obtain the lowest level of education, so the majority of them are "repairers" instead of the other two professions and due to the low percentage of existence among all the archetypes, it's not hard to explain why "tailor and "weapon maker" are the professions to the skill "toolmaker", which is the most likely choice for archetype "the fool", but in the end, the green and blue line have the minimum at archetype="Fool".

4.4 Evaluation

As mentioned in the previous sections, although most of the relationships and probabilities are predesigned, a minority of structure and probability table in the Bayesian network is still empty. Therefore, when some new parameters in Bayesian network generation might be inaccurate resulting in the conflict between the outcome and requirements. In addition, it is tough to guarantee the accuracy of all rules at the stage of rule designing. So evaluation and improvement are indispensable processes for PCG.

In order to evaluate the outcome of the generator, a standard should be set. The distribution of professions will be measured here. As there is no direct rule about the type of professions, the Appendix A *Rule 10.a* is chosen as part of the criteria because the profession level is nearly the end of the Bayesian net-

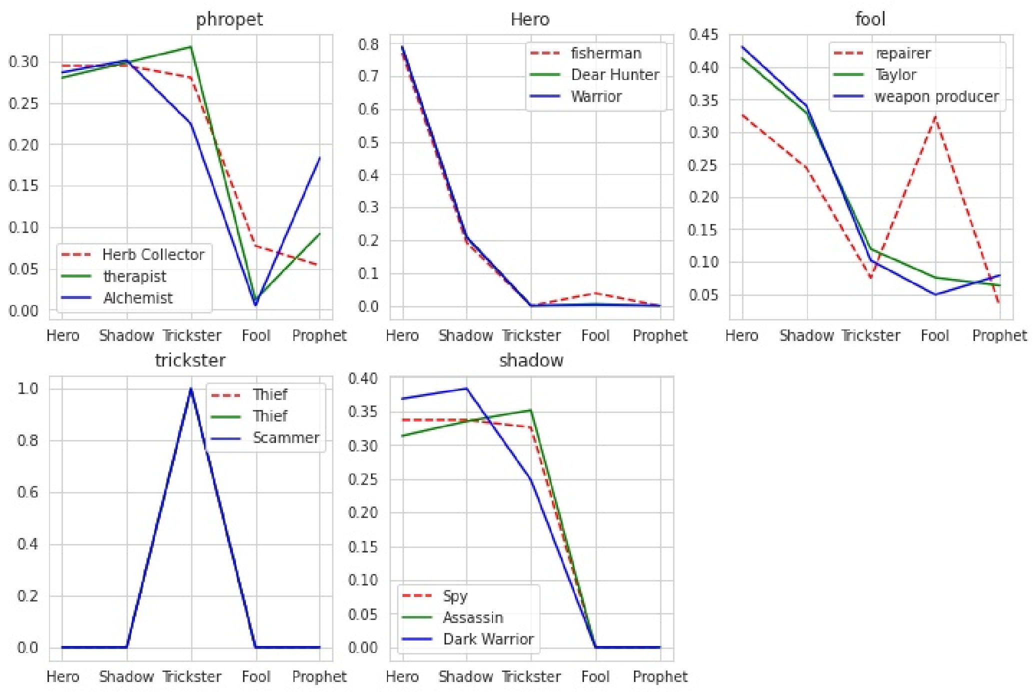


Figure 5. The probabilities of different professions given the type of character Each subplot represents the main professions of a certain archetype. The red dotted line is for the low level of profession, green line for medium, and blue line for the high level.

work, while the node profession is fixed given the values of skill and profession level. The expected distribution will be the expected professional level multiplies the archetype distribution (defined by Appendix A *Rule 2a-e*), which is the start of the Bayesian network.

In Figure 6, the scatters are the probabilities of different professions observed and the green broken line is the expected ones. Every five professions is a group on the same profession level, and the corresponding archetypes of these elements in each group are in the sequence of "prophet", "hero", "fool", "trickster" and "shadow". From this picture, the tendency of the values of each group is similar and there are peaks at the data points of "hunting" professions, which are the most likely skill the character may learn given the archetype "Hero".

Firstly JS-divergence is used to measure how the observed distribution is different from the theoretical one as a metric. It comes out that $JS(P||Q) = 0.075 bits$, as the base of log is 2. P denotes the observed distribution and Q for the expected distribution. It is a small number close to 0, so the conclusion can be addressed that the generator satisfies the general requirements.

But as JS-divergence is neither equal nor very to 0 so minor progress is still possible. The conditional probability table of profession level is changed, because the criteria Appendix A *Rule 10.a* has conflicts with Appendix A *Rule 8*. In Figure 6, it could be observed that the low and medium level of professions has data points above the expected distribution line, so the action to increase the possibility of high level education and decrease the others is taken. Appendix A *Rule 8.d* and *Rule 8.e* has changed: the probabilities of professionLevel=1 decrease n or and probability of professionLevel=2 increase n. To illustrate, Appendix A *Rule 8.d* is changed into:

$$\begin{aligned} &educationLevel = 2 \wedge toolpower = 1 \rightarrow \\ &\qquad [0.7 - n] professionLevel = 1 \\ &\qquad [0.3 + n] professionLevel = 2 \end{aligned}$$

and *Rule 8.e* in the same way.

As n changes from 0 to 0.5 (scale 0.05), the value of $KL(P||Q)$ firstly decreases rapidly, then slowly reaches a minimum, and finally shows an increasing trend. These means the local optimal minimum point in this case is between $n = 0.35$ and $n = 0.45$.

5. CONCLUSION

The present paper uses probabilistic grammar to formalize the character's background design in a game within the concepts of monomyth after defined the contents in the background. The five archetypes, the Hero, the Shadow, the Trickster, the Fool, and the Prophet here, correspond to their most possible characteristics set, including skill, education level, tools, and profession, together with probabilistic grammar, however, it still provides other possibilities enriching the characters design.

The rules are realized by a Bayesian network, and the result consists of character information in JSON format and probabilities of the specific outcomes. After the criteria have been chosen, The observed probability distribution is measured with $JS(P||Q)$, which is 0.075 bits in this research, to give a preliminary judgment. Then rules(Appendix A *Rule 8*) conflicting with the criteria(Appendix A *Rule 10.a*) will be updated and find a local optimal which minimizes $KL(P||Q)$. $KL(P||Q)$ is a function of the unknown value n in new rule, i.e. Equation **??** . The minimum point lies between n=0.35 to n=0.45 as shown in Figure 4. When the divergence is small enough for designers to accept, the process of character design is finished. This proves that the Bayesian network and the ruleset can be im-

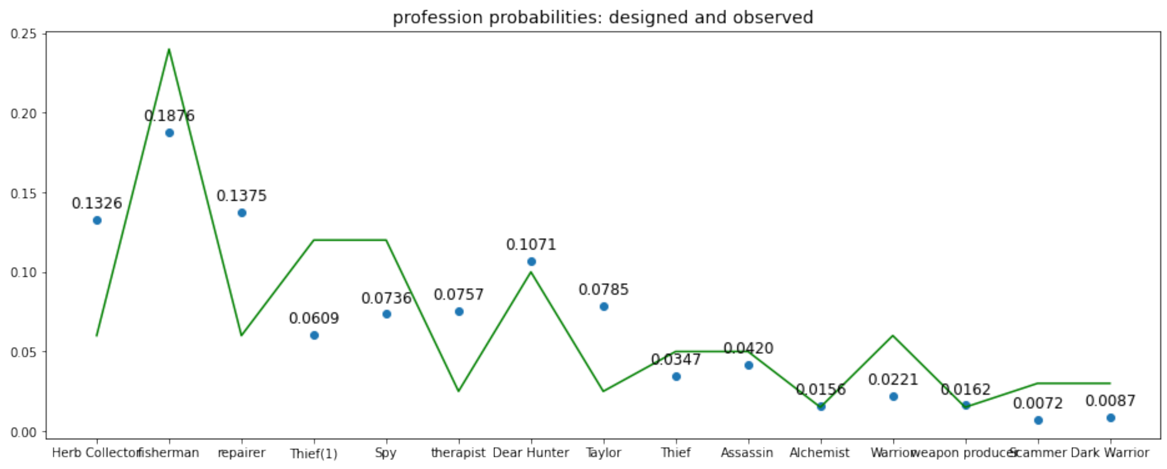


Figure 6. The distribution of professions among all the characters

proved and measured quantitatively to fulfill the requirements for PCG-character design.

6. FUTURE WORK

Initially, this project is a showcase to present the possibility of using probabilistic grammar and Bayesian networks to generate the background of NPCs. The intention is to prove the feasibility of the entire process. For practical game design, more elements can be added, e.g. skin, the color of hair, nationality, family life, frustration, and obsession mentioned in [13].

What's more, in the process of evaluation and improvement, the rules and involved conditional probability tables are currently updated by hands. In This case, the local minimum KL-divergence is also an approximate value whose error is within 0.5, which implies the relatively minor divergence between the Bayesian network and the criteria is under-discovered. At the same time, no measurement scores the criteria. In another word, there is no guarantee the selection of criteria is the most reasonable. Taking advantage of automatic methods to choose or edit the rules is interesting to be considered in the future to promote efficiency and accuracy.

Last, the Bayesian network is now initialized by manual work which is weak in scalability. To construct a more complicated network, new methods are supposed to be utilized. For example, the initialization can be divided into two parts – inputting compiled conditional probability tables or learning from existed data through machine learning algorithms (e.g. K-means clustering). Other topics like runtime and storage of the system also remain to be researched.

Acknowledgement

Thanks to my supervisors Marcus Gerhold and Vadim Zaytsev for their unfailing patience, professional support and valuable feedback every time I lost my directions. Without their help, I would not be able to understand this topic and find practical methods in 10 weeks.

7. REFERENCES

[1] C. Ashmore and M. Nitsche. The quest in a generated world. In *DiGRA conference*. Citeseer, 2007.

[2] G. Brewka, T. Eiter, and M. Truszczyński. Answer set programming at a glance. *Communications of the ACM*, 54(12):92–103, 2011.

[3] J. Campbell. *The hero with a thousand faces*, volume 17. New World Library, 2008.

[4] C. Canheti, F. Andalo, and M. L. H. Vieira. Case study: Game character creation process. In *International Conference on Applied Human Factors and Ergonomics*, pages 343–354. Springer, 2018.

[5] M. D. Dickey. Game design narrative for learning: Appropriating adventure game design narrative devices and techniques for the design of interactive learning environments. *Educational Technology Research and Development*, 54(3):245–263, 2006.

[6] J. Doran and I. Parberry. Controlled procedural terrain generation using software agents. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(2):111–119, 2010.

[7] J. Dormans. Level design as model transformation: a strategy for automated content generation. In *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games*, pages 1–8, 2011.

[8] M. Frade, F. F. De Vega, and C. Cotta. Modelling video games' landscapes by means of genetic terrain programming-a new approach for improving users' experience. In *Workshops on Applications of Evolutionary Computation*, pages 485–490. Springer, 2008.

[9] R. H. García-Ortega, P. García-Sánchez, J. J. Merelo, A. San-Ginés, and A. Fernández-Cabezas. The story of their lives: Massive procedural generation of heroes' journeys using evolved agent-based models and logical reasoning. In *European Conference on the Applications of Evolutionary Computation*, pages 604–619. Springer, 2016.

[10] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, 1969.

[11] E. Jurado, K. E. Gillam, and J. McCoy. Non-player character personality and social connection generation. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, FDG '19, New York, NY, USA, 2019. Association for Computing Machinery.

[12] D. Karavolos, A. Bouwer, and R. Bidarra. Mixed-initiative design of game levels: Integrating mission and space into level generation. In *FDG*, 2015.

[13] P. Lankoski. Character design fundamentals for role-playing games. *Beyond Role and Play*, pages 139–148, 2004.

[14] S. M. Lucas and V. Volz. Tile pattern kl-divergence for analysing and evolving game levels. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 170–178, 2019.

[15] A. Medić. *Literature and Contemporary Media: Campbell's Monomyth in video game trilogies" Dragon Age" and" Mass Effect"*. PhD thesis, Josip Juraj Strossmayer University of Osijek. Faculty of Humanities and . . . , 2017.

[16] N. Nurseitov, M. Paulson, R. Reynolds, and C. Izurieta. Comparison of json and xml data interchange formats: a case study. *Caine*, 9:157–162, 2009.

[17] F. Ongenae, M. Claeys, T. Dupont, W. Kerckhove, P. Verhoeve, T. Dhaene, and F. De Turck. A probabilistic ontology-based platform for self-learning context-aware healthcare applications. *Expert Systems with Applications*, 40(18):7629–7646, 2013.

[18] B. Pell. Metagame in symmetric chess-like games. 1992.

[19] L. Petri, H. Satu, and E. Inger. Characters in computer games: Toward understanding interpretation and design. 2003.

[20] A. S. Ruela and F. G. Guimarães. Procedural generation of non-player characters in massively multiplayer online strategy games. *Soft Computing*, 21(23):7005–7020, 2017.

[21] B. Schuettengruber, M. Ganapathi, B. Leblanc, M. Portoso, R. Jaschek, B. Tolhuis, M. van Lohuizen, A. Tanay, and G. Cavalli. Functional anatomy of polycomb and trithorax chromatin landscapes in drosophila embryos. *PLoS Biol*, 7(1):e1000013, 2009.

[22] N. Shaker, J. Togelius, and M. J. Nelson. *Procedural Content Generation in Games*. Springer Publishing Company, Incorporated, 1st edition, 2016.

[23] G. Smith. Understanding procedural content generation: a design-centric analysis of the role of pcg in games. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 917–926, 2014.

[24] G. Smith, J. Whitehead, and M. Mateas. Tanagra: A mixed-initiative level design tool. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, pages 209–216, 2010.

[25] J. V. Stone. *Bayes' Rule: A Tutorial Introduction to Bayesian Analysis*. Sebtel Press, 2013.

[26] A. THOMAS and M. H. SKOLNICK. A probabilistic model for detecting coding regions in dna sequences. *Mathematical Medicine and Biology: A Journal of the IMA*, 11(3):149–160, 1994.

[27] B. Tillman. *Creative character design*. CRC Press, 2012.

[28] J. Togelius, A. J. Champandard, P. L. Lanzi, M. Mateas, A. Paiva, M. Preuss, and K. O. Stanley. Procedural Content Generation: Goals, Challenges and Actionable Steps. In S. M. Lucas, M. Mateas, M. Preuss, P. Spronck, and J. Togelius, editors, *Artificial and Computational Intelligence in Games*, volume 6 of *Dagstuhl Follow-Ups*, pages 61–75. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2013.

[29] J. Togelius, N. Shaker, and J. Dormans. Grammars and l-systems with applications to vegetation and levels. In *Procedural Content Generation in Games*, pages 73–98. Springer, 2016.

[30] F. S. G. Toto and G. Vessio. A probabilistic grammar for procedural content generation. In *Sixth Workshop on Non-Classical Models of Automata and Applications (NCMA 2014)*, volume 31, 2014.

[31] C. Vogler. *The writer's journey*. Michael Wiese Productions Studio City, CA, 2007.

[32] H. Warpefelt. *The Non-Player Character: Exploring the believability of NPC presentation and behavior*. PhD thesis, Department of Computer and Systems Sciences, Stockholm University, 2016.

[33] R. L. West and C. Lebiere. Simple games as dynamic, coupled systems: Randomness and other emergent properties. *Cognitive Systems Research*, 1(4):221–239, 2001.

[34] V. Zaytsev. Grammar Zoo: A Corpus of Experimental Grammarware. *Fifth Special issue on Experimental Software and Toolkits of Science of Computer Programming (SCP EST5)*, 98:28–51, Feb. 2015.

APPENDIX

A. THE RULESET

```
1.age
 a. [0.25]ageLevel=0 (12<age<=18)
 b. [0.4]ageLevel=1 (18<age<=25)
 c. [0.25]ageLevel=2 (25<age<=60)
 d. [0.1]ageLevel=3 (60<age)


2.archetype
 a. [0.4]archetype="Hero"
 b. [0.2]archetype="Shadow"
 c. [0.2]archetype="Trickster"
 d. [0.1]archetype="Fool"
 e. [0.1]archetype="Prophet"


3.education level
 a. archetype="Hero"||archetype="Shadow"->
    [0.4]educationLevel=0[0.35]educationLevel=1
    [0.2]educationLevel=2[0.05]educationLevel=3

 b. archetype="Trickster"->[0.33]educationLevel=0
    [0.45]educationLevel=1[0.2]educationLevel=2
    [0.02]educationLevel=3

 c. archetype="Fool"->[0.8]educationLevel=0[0.15]
    educationLevel=1[0.05]educationLevel=2
    [0]educationLevel=3

 d. archetype="Prophet"->[0.25]educationLevel=0
    [0.4]educationLevel=1[0.27]educationLevel=2
    [0.08]educationLevel=3

 e. educationLevel<= ageLevel

4.skill
 a. archetype="Hero"-> [0.6]skill="hunt"
    [0.2]skill="tool maker" [0.2]skill="potion craft"

 b. archetype="Shadow"->[0.6]skill="lurk"[0.2]
    skill="tool maker"[0.2]skill="potion craft"

 c. archetype="Fool"->[0.1]skill="potion craft"
    [0.8]skill="tool maker"[0.1]skill="hunt"

 d. archetype="Trickster"->[0.6]skill="steal"
    [0.2]skill="lurk"[0.2]skill="potion craft"

 e. archetype="Prophet"-
>[0.9]skill="potion craft"
    [0.1]skill="tool maker"


5.tool event
 a. archetype="Hero"||archetype="Shadow"->
    [0.6]placeEvent="battle field"
    [0.2]placeEvent="town"
    [0.2]placeEvent="island"

 b. archetype="Fool"->[1]placeEvent="town"

 c. archetype="Prophet"->[0.1]placeEvent="town"
    [0.9]placeEvent="island"

 d. archetype="Trickster"->
    [0.2]placeEvent="battle field"
    [0.6]placeEvent="town"[0.2]placeEvent="island"


6.tool power
    if( skillType="potion craft"
    &&placeEvent="island"||
    (skillType="tool maker"||skillType="steal")
    &&placeEvent="town"||
    (skillType="hunt"||skillType="lurk")
    &&placeEvent="battlefield")
        ->[1]toolpower=1
    else
        -> [1]toolpower=0
7.tool class
 a. toolclass=
     [0.5]low
     [0.3]middle
     [0.2]high

8.profession level
 a. educationLevel=0->
    [1]professionLevel=0

 b. educationLevel=1&&toolpower=0->
    [0.6]professionLevel=0
    [0.4]professionLevel=1

 c. educationLevel=1&&toolpower=1->
    [0.4]professionLevel=0
    [0.6]professionLevel=1

 d. educationLevel=2&&toolpower=0->
    [0.9]professionLevel=1
    [0.1]professionLevel=2

 e. educationLevel=2&&toolpower=1->
    [0.7]professionLevel=1
    [0.3]professionLevel=2

 f. educationLevel=3&&toolpower=0->
    [0.1]professionLevel=1
    [0.9]professionLevel=2

 g. educationLevel=3&&toolpower=1->
    [1]professionLevel=2


9.profession
    prof_value={
        "potion craft":
        ["Herb Collector", "therapist","Alchemist"],
         "hunt":
        ["fisherman", "deer Hunter", "Warrior"],
         "tool maker":
        ["repairer", "Taylor", "weapon producer"],
         "steal":
        ["Thief(1)","Thief","Scammer"],
        "lurk":
        ["Spy","Assassin", "Dark Warrior"]
    }
 a. profession=[1]{prof_value[skill][professionLevel]}


10.other requirements
 a. [0.6]professionLevel=0 [0.25]professionLevel=1
    [0.15]professionLevel=2
```