# The Game of Thrones Automyth - A Knowledge Graph Based Spin-off Story Generator

Ceren Akkalyoncu[1][2657341], Paola Feil[1][], Gabriel Hoogerwerf[1][2735225], Redouan Lamkaraf[1][2672658], and Teresa Liberatore[1][]

Vrije Universiteit Amsterdam, the Netherlands

**Abstract.** Story generation is a growing sub-field of computational creativity aimed at emulating the creative human skill of narration. Many approaches have been developed to automate story generation: in this work, the focus lies on the use of Knowledge Graphs to achieve this goal. An ontology has been modeled after the Hero's Journey story structure and a knowledge graph that stores the relations of the original story has been used: for the purpose of this study the Game of Thrones KG. The relations stored in the original knowledge graph are thus exploited through a community detection algorithm that allows generating stories that are meant to be coherent to the original story content-wise. The resulting product can thus be classified as a short summary of a spin-off based on the original story. This method is compared to a random instantiation baseline in order to test if the use of a knowledge graph for story generation can indeed enhance the quality of the generated output. Furthermore, as stories are meant to be read by humans, a human evaluation has been performed. The result is that, as expected, stories generated with community detection have been evaluated to be more enjoyable, creative, and coherent content-wise.

**Keywords:** Knowledge Graphs · Story Generation · Computational Creativity · Hero Journey · Game of Thrones · Semantic Web.

## 1 Introduction

Computational creativity is the application of AI technologies to emulate and stimulate human creativity. The field of Artificial intelligence is indeed not only aimed at emulating and enhancing quantitative intelligence, but also at developing technologies capable of emulating human creative intelligence. But what is creativity after all? Undoubtedly, creativity is intrinsically related to narratives and stories. Every piece of art, whether it be a painting, a song, a poem, or a book, has a story to be told.

This is one of the reasons why 'story generation' as a sub-field of computational creativity has been growing over the years. In this work, we are investigating a story generation method that uses Knowledge Graphs and relies on a structured narrative.

Studies on narratives and story structures have found a narrative pattern shared by most stories across time. This story structure, identified by Joseph

Campbell in 1949, is referred to as the Hero's Journey, also known as the Monomyth [5]. Simply put, the Hero's Journey is a sequence of typical story events which represent the metaphorical journey of the main character throughout the whole narration. For the scope of this work, the Hero's Journey story structure has been embedded in our story generator. In particular, the ontology about the story content has been modeled after a simplified version of the Hero's Journey structure.

Such patterns though, represent only the backbone of a story. What makes every piece of narration unique is in fact the choice of the author regarding every single character, element, surroundings, etc. This represents a second challenge for our method which can be referred to as the instantiation problem. For this purpose, the Game of Thrones (GoT) Knowledge Graph has been chosen as a use case to implement our solutions. Instances of characters for our automatically generated stories will be therefore based on the story universe of GoT.

Three story generator methods have been developed and compared: a baseline that generates stories with random instantiations, and thus without exploiting the graph representation of our data, while the second method utilizes it using a community detection approach. A third method was also implemented that takes advantage of weighted relations between characters, from a data set available online [9].

### 1.1 Motivation

The ability to craft, tell, and understand stories has long been held as a hallmark of human intelligence and a long-term goal of artificial intelligence. In particular, story generation is the problem of automatically selecting a sequence of events that meets a set of criteria and can be told as a story. Story generation is knowledge-intensive as traditional story generators rely on a priori defined domain models about fictional worlds, including characters, places, and actions that can be performed. Once the domain model has been engineered, a story generation system can tell a potentially infinite number of stories involving these characters, places, and actions known to the system. [1]

This research finds its motivations in the development of a story generator that relies on a priori structure, the Hero's Journey, but not on a defined domain model. Indeed, for the scope of the application and evaluation of the approaches a domain had to be defined, but the ontology and the generating method has been built in such a way that every domain model could fit in it.

### 1.2 Problem Definition

### 1.3 Research Question

The goal of this work is thus to create a story generator based on already existing story knowledge graphs, that through community detection would allow for a generation of stories coherent with the original one. In that sense, the scope of this work can be declared to be the development of a 'spin-off story generator'.

### 1.4 Challenge

Story generation starting from graphs is a challenging and complex problem, and we are fully aware of the limitations of this study. The focus of this work is on the algorithm to generate the bone structure of the story. In that sense, no attention has been put in the use of natural language processing techniques to either extract triples and thus build knowledge graphs from text or to translate the graph of the generated story back into text. It could thus be claimed that this research mainly focused on the filling of what can be considered the sandwich of story generation, where the bread is usually very much related to the use of NLP techniques. For those reasons, one of the biggest challenges of this paper can be found in the evaluation of the two methods. Indeed, story generation methods always rely on a qualitative human evaluation to assess the performance of the approach as stories are meant to be read by humans. As the stories generated with our method are only made human readable through the use of a visualization, the evaluation is less intuitive. Hence, the evaluation of the generated stories needs to take into account the content of the instantiations rather than the flow of the sentences. This kind of evaluation thus requires a higher effort, and it is debatable if it can be considered scientifically acceptable or not.

## 2 Related work

Most of the work related to story generation relies on sequence-to-sequence models that generally fail to model the semantic meaning and causality of story events. One approach to solve this problem relies on generating event sequences by walking on an event graph, such as implemented by Chen and al. [1] In particular, as shown by Yao et al. [3], if the events are well planned, then the correctness of generated stories is almost guaranteed, and furthermore, the stories can be easily controlled by modifying the events. Many works in the field discussed the advantages of the use of knowledge graphs to model story generation approaches. Especially the work of Lei et al. [4] is noteworthy as it implements a combination of a plot graph and a sequence to sequence approach in order to generate articles and thus exploiting the positive sides of each approach.

In general, literature in the field is growing, as the use of knowledge graphs has been shown to significantly improve the quality of the generated stories as it allows for coherence in terms of structure and content. [7]

## 3 Approach

It is safe to say that the core of a story can be found in its succession of events. We refer to the concept of event when a certain thing takes place. In other words, an action. With this in mind, a template for the succession of events was chosen as a starting point, namely The Hero's Journey, which helps shaping every event by defining the happenings. This however is not enough, as every different story that undergoes such a template is still different from each other as it has completely different elements.This is the so-called instantiation issue and for our case of study we rely on existing instances to be picked. A proper choice of these is not a trivial task but is important as it aims at creating a coherent and meaningful story. For this purpose, three different functions were implemented and their results were compared. Strictly speaking, our approach consists in combining a first KG that represents the modeled structure of events with a second KG containing the instances necessary to fill the narration. The output is another graph that can be considered the newly generated story, consisting of triples grouped by every step of the journey. As previously stated, event structure is based on the Hero Journey. Data, on the other hand, are mostly imported from a pre-existing KG about the famous TV Show Game of Thrones, GoT.

### 3.1 The Hero's Journey

Joseph Campbell defined the Hero's Journey structure in 1949 as the universal pattern of all mythical stories that follow the archetype of a hero [5]. As it was identified as an ageless recurring pattern in successful stories across cultures, the monomyth was consequently more consciously applied in creative storytelling, the most famous example being George Lucas' Star Wars stories. However, since its discovery, this story structure was also identified in a wider range of genres, such as romantic comedies, crime, or horror [6]. This is due to the fact that the monomyth revolves around archetypical characters in human psychology, that is characters of which people across time and place dream of, and which therefore often appear in our stories. Due to this wide applicability, the monomyth was chosen as the guiding principle of the ontology design, as it might inspire more researchers to draw from it. Furthermore, the original formulation of the events matches a mythical story setting and is therefore well compatible with the GoT story universe. The following provides a short description of each story stage and its purpose within the creative process.

In line with its name, the story describes the journey of a hero. Whether this is an outward physical or inward mental journey, it generally tells about how the hero leaves their ordinary and comfortable environment and ventures into an unknown situation, where the hero challenges the antagonist, and finally returns changed from the experience [6]. For simplicity, the designed ontology reflects a physical journey. The journey pattern is commonly disassembled into twelve stages. The first stage, the 'ordinary world', introduces the hero's everyday life and character. This presentation of the protagonist within their comfort zone before the actual beginning of the story makes the audience empathize with his struggle later on. In the second stage, the hero receives a 'call to adventure', that is a quest or challenge such as a threat to safety, which summons the hero to leave the comfort of the ordinary world and marks the beginning of the plot. In the next step, the 'refusal of the call', the hero shows themselves reluctant to accept their fate, doubting their abilities and holding on to the reassurance of their own known life so far. This helps the audience to identify with the hero, as this is most people's likely first response. In the fourth stage, there is a ‘meeting with the mentor’, a wise character symbolizing the bond between child and parent or man and God. The mentor prepares the hero to face the journey ahead by passing on advice, confidence, or a lesson. Next, finally ready to accept the call, the ‘crossing of the threshold’ happens. This is the moment where the hero sets onto his adventure to leave the ordinary world and enter the unknown special world, the place of the quest. However, before reaching the final goal, the hero must prove themselves by failing, learning, and succeeding in various tests against enemies with the help of new allies. This sixth stage shows the character’s development to grow from the ordinary person to a true hero and therefore helps to identify further with the protagonist. In the seventh step, the ‘approach to the inmost cave’ happens, where the hero comes close to the often-dangerous place of the quest and therefore finds themselves faced with previous fears from the call. This moment of pause builds anticipation for the ordeal to come and highlights its magnitude. In the eight stage the actual ‘ordeal’ takes place, a struggle against the enemy or greatest fear which tests the hero’s skills acquired during the journey thus far. Suspense is created by bringing the hero to the brink of (metaphorical) death. Next, the hero receives the ‘reward’ of the ordeal. They emerge stronger from the experience and seize their reward such as knowledge, reconciliation, or an important artifact. On ‘the road back’, the hero starts his return to the ordinary world but realizes that the consequences of the confrontation in the ordeal must be faced. Vengeful dark forces might hunt the hero, disturbed by the seizing of the reward. In the eleventh stage, the ‘resurrection’ marks the last death-or-life moment, where the hero must prove to have learned their lesson from the ordeal. This crisis has far-reaching consequences for the ordinary world, and the hero is rewarded by being metaphorically reborn as a stronger person. Finally, the hero makes his ‘return with the elixir’ which ultimately helps those left behind in some way and gives cause to a celebration. While the hero has returned to the ordinary world, they are a changed person now.

### 3.2 Ontology Modeling

For the translation of the chosen template into computer-readable knowledge, the well-defined twelve event division was kept. For this purpose we partially took advantage of a preexisting ontology aimed at representing an event along with its details: the Simple Event Model (SEM) [**?**]. Some of the classes from SEM were thus adopted to define a basic main event class and, based on that, event subclasses were created for every step of the journey. We will refer to the latter as subevents. They are involved in all the properties of the main event class as well step specific declared properties that along with the common ones will be later instantiated. Designing such ontology after the template of a story is not an easy task as usually, because of its universal nature, such structures are very abstract and conceptual since they must be applied to various practical examples. Since we want our story to be not as generic, we modeled the ontology in a way that is logically compatible with the adopted data (from the GoT domain), and refers exclusively to physical (and not metaphorical) actions performed by the hero.

Under these premises, strong embedding choices were done during this phase and the genre matches that of myths and adventure stories. For instance the fourth event of the story, usually referred to as 'Meeting the Mentor', was modeled by defining two properties that involve the instantiation of the 'Mentor name' together with the 'Power' that the Mentor teaches to the Hero.

```
HERO:MeetingTheMentor rdfs:subClassOf sem:Event .

HERO:meetsMentor a rdfs:Property;
rdfs:range HERO:Mentor;
rdfs:domain HERO:MeetingTheMentor.

HERO:powerLearned a rdfs:Property;
rdfs:range HERO:HeroPower;
rdfs:domain HERO:MeetingTheMentor.

HERO:Event_04 a HERO:MeetingTheMentor; #instanciated
HERO:journeyStage '4';
rdfs:label "Meeting the Mentor".
```

Implementation wise, the choice to model every event as N-ary relation was made. N-ary relation is a popular technique to represent complex knowledge pieces. In our output graph, every event is represented by an instance of the class of that specific step in the journey and entertains $n$ relations with $n$ instances, with $n$ changing every time based on the event. Notable is that an instance of every event is actually defined already in the initial ontology. That is because some core features are known prior to the story generation phase (i.e. the stage in the journey and a label) and an instance of every step is necessary as a subject for all the triples that are going to be created and constitute the story.

### 3.3 Dataset

In order to apply the event structure while creating a new narration, a set of elements to characterize every new story is necessary. Such elements, which can be thought of as all the entities that appear in a story are thus instantiated in a turtle file as being of rdf:type of their assigned class. For our application, we rely on existing data that can be found on the world wide web for the value of most of the instances. Specifically, the considerable amount of both loose and structured data that can be found about the famous TV Show *Game Of Thrones* accounts for the choice of such domain for our case study.

It is to be remarked, though, that given the lack of data about some classes of interest from the GoT databases, some classes have been manually instantiated.

We use Wikidata as a source for information on Game of Thrones characters, as it contains well-structured data that can easily be integrated. We define a set of main characters and properties of interest, then query Wikidata for this information. The available properties are mainly static knowledge about the characters' family and love interests, and do not provide information on the enemies or events the character was involved with. The results are stored as an RDF file used for character instantiations. The data extraction process is elaborated under Annex section.

Another dataset that provides information about GoT is the Network of Thrones data [9]. Instead of defining relations between characters, this dataset considers the interaction dynamics between characters. Using the series' script, each pair of characters are connected by an edge weighted by the number of interactions. Interaction types can be categorized as follows:

1. Character A speaks directly after Character B
2. Character A speaks about Character B
3. Character C speaks about Character A and Character B
4. Character A and Character B are mentioned in the same stage direction
5. Character A and Character B appear in a scene together.

Unlike Wikidata, this dataset shows the dynamics of the series in more detail, as it reflects on the script directly. However, it still fails to inform us about the nature of the interaction. We use a subset of this data, that only includes the selected main characters and side characters who relate to them.

## 4 Story Generation, the Algorithm

The process of generating a new graph containing the story can be conceptually divided into two tasks. The first one consists in looking upon every subevent, and for each one considering all the properties in which they are involved (recall that the model is structured after N-ary relations). Such properties are accurately defined in the ontology graph by their domain (`rdfs:domain`), which is the event itself, and range (`rdfs:range`). Once the range is acknowledged, it is necessary to find an according instance for that class. This is where the second task starts, namely the instantiation problem. To tackle this issue we propose two different methods and we compare them with a random base line. It is worth mentioning how the algorithm designed to generate the story was planned to be flexible and independent from the KGs given as input.

### 4.1 Random baseline

The first method, later to be considered as the baseline, picks randomly one possible instantiation from all the existing instances of a specific class. This is therefore just a query to the GoT graph asking for all the nodes entities in a relation `rdf:type` with the required range. Noteworthy is the fact that having a structured KG of elements for this task is not required as only a list of feasible instances could be enough.

### 4.2 Smart Selection

We attempt to make smarter instantiations that reflect on the universe of choice, by integrating contextual information into the story generation step. The GoT universe is chosen as a basis for this project, but this approach is applicable to any knowledge graph. We only apply smart instantiation to the characters, and follow the random initialization method for other instances. This is because information corresponding to all instances of the Hero's Journey is unavailable on the GoT universe.

**Community-based Instantiation** The first method makes use of the knowledge graph created from Wikidata. We extract the number of relations between each character using SPARQL queries and use this information as an indication of how related the characters are in the context of GoT. This information is represented in a weighted undirected graph. We then identify communities within this graph to find closer relation groups, where characters are more likely to interact with each other. The resulting communities are shown in Figure 1 below.

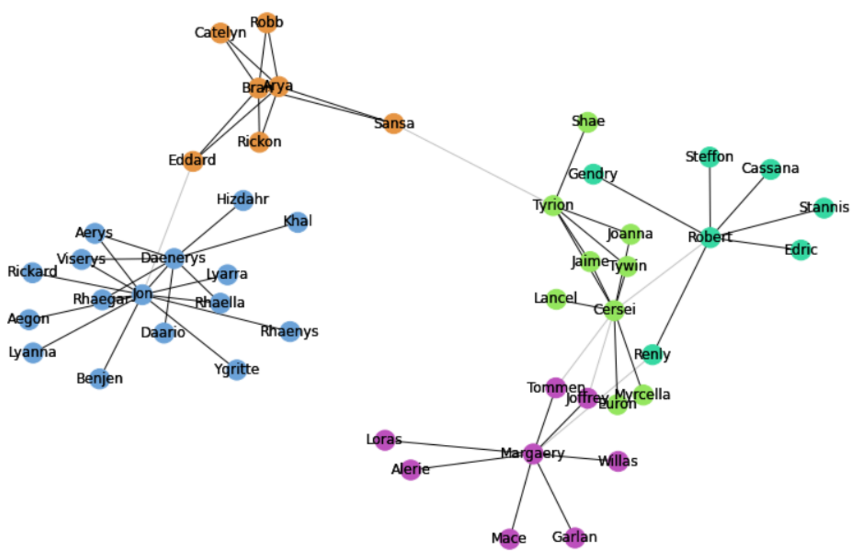


**Fig. 1.** Community detection on Game of Thrones characters data. Surnames omitted for readability.

When instantiating characters for the story, we refer to this graph to make reasonable choices. The hero is picked randomly from the main character set as defined before. A character's ally is chosen randomly from within that character's community. The main villain is randomly picked from the set of main characters that belong to any other community than the hero's. The disadvantage of this method is that it depends on the information provided on Wikidata. The relations represented in Wikidata are rather superficial, and only include family or marital relations. Therefore, the communities often reflect the Houses of the characters. Furthermore, family relations are not enough to indicate that two characters are likely to be allies within the context of GoT.

A detailed description of the data extraction and the implementation of community detection is given in Annex section.

**Interaction-based Instantiation** The second method uses the Network of Thrones data when instantiating characters related to the hero. The model randomly returns a character based on the interaction weights the hero has between the characters.

Even though this approach better embeds the interactions between characters, it does not provide information on the nature of the interaction. Therefore, it is not possible to understand the manner of their relationships, such as if they were likely to be allies or enemies. For example, Ned Stark is the top candidate to be Cersei Lannister's "mentor", even though they are enemies in the original story.

## 5 Visualization of the story

The story generation methods output an instantiated knowledge graph that contains the generated story. In order to visualize our Story in a human-readable way, we integrated a dashboard visualization tool based on Google Data Studio. An example is provided in the Annex section. 4 The full interactive dashboard can be accessed under this link: `https://datastudio.google.com/reporting/3e1887bb-c02d-4c74-8d74-a14b7399c524` In practice this was done by querying our story graph with SPARQL. A query for each event of the story has been designed to report a sentence that represent the given event: this is done by binding the label of all the elements of the event and concatenating them with string of hard coded text. The queries for each event where then collected together in a unique query that was used to generate all the stories used in the experiment. The resulting sentences are then imported in DataStudio in a csv file in order to visualize the generated story. The visualization is thus the final output of our work as it is the human readable version of our generated stories and for each method a generated visualization output is reported in the annex.

## 6 Experiments and Evaluation

### 6.1 Setup

In order to compare the output of the different developed methods an experiment involving human evaluation was setup. We introduce three evaluation metrics, ranging from 0 to 3, to qualitatively asses the performance of the three different character picking approaches. Such metrics are:

*Coherence*: Coherence aims at establishing how much does the character choice fits into the existing Game of Thrones plot. In other words, consistency, and plausibility with GoT world.

*Creativity*: Creativity represents the level of novelty that the story carries within itself. Highest level of creativity means that the generated story could introduce interesting narration.

*Enjoyment*: Enjoyment is purely how much the reader has enjoyed the story. Participants are asked to read the story proposed to them and assign a numerical value in the range [1-3] for every of the evaluation parameters. The nature of the story was not revealed to the reader beforehand.

The visualization of five generated stories for each method were given to a sample of 10 people who watched all the TV Show Game of Thrones and the results, based on the metrics described above, are reported in the following section.

### 6.2 Results

The ten readers assigned points ranging from 1 to 3 on each metrics for each of story. In the following table the mean of the the assigned point for each metric for all the stories generated by a given method has been reported. From this table it can be easily seen that the metrics that got the highest results is Enjoyment: this can be explained by the fact that all the readers had previously seen the TV show GoT: this feature of the participant selection was necessary for the coherence evaluation but undoubtedly biased the values assigned to the Enjoyment metrics. For what concerns creativity, the stories that received the highest score are the randomly generated ones: this can be explained by the fact that humans usually associates to the terms creativity what is considered to be unexpected: in that sense a random instantiation assured the choice of the relations between character to be unexpected. The results for the coherence metrics, which is after all the metrics we are the most interested about, doesn't show significant differences between the stories generated with the different methods. In general random generated stories got a lower coherence score, but none of the method reached an overall score higher than 1.5.

| | Coherence | Creativity | Enjoyment |
|---|---|---|---|
| Random | 0.6 | 2 | 2.8 |
| Community | 1.1 | 1.3 | 2.6 |
| Relation | 0.8 | 1.1 | 2.2 |

**Table 1.** Results of the evaluation: the mean of the metrics for each category of the generated stories is reported

## 7 Limitations

Encountering boundaries in the space of exploration and development of stories has limited the differentiation of generating stories. The Hero Journey is a very broad description and modeling into one ontology narrows down all the possible instantiation of events. Missing features describing the characters and other entities by data that are more oriented to a specific topic; this limitation causes boundaries to the creativity of generating stories. The diversity is therefore more dependent on the data of the story provided.

Another limitation is regarding smart initiation, where we want to integrate a given universe in our story. For smart initiation of characters, both methods fail to identify interesting communities of characters. This is mainly due to the lack of data on more complex relations, which also results in similar stories generated.

Moreover, a story is usually composed of actions as well as description narrative breaks. Our approach is entirely based on an events structure and does not provide a proper description of the elements or the settings.

## 8 Conclusions

This research investigated the possibility to generate stories from and to Knowledge Graphs by exploiting their relational structure via a community detection algorithm that would enhance the quality of generated stories. For this study, the GoT Knowledge Graph was used as a starting point and the ontology used to generate the stories was modeled after the Hero's Journey story structure. A community and detection method were developed to instantiate the generated knowledge graph and a random instantiation method was developed as a baseline for the evaluation of the generated stories. A human evaluation was then performed to assess the quality of the generated stories. It was found that the community detection instantiation enhanced the coherence of the generated story with respect to the baseline, but the no community generated story got a score higher than 1.6 out of 3 in terms of coherence. It is to be remarked, though, that a high level of enjoyment was reported on average for all the generated stories.

### 8.1 Future Work

A simple step to make the instance selection more reflective could be to merge the two distinct methods we applied. As such, information about the characters as well as the interaction dynamics could be integrated, to give a more representative overview of the relations between characters. However, this would still not address the other challenges mentioned before.

A further improvement in conducting a search for data on knowledge completion, research will be in the future aimed at the use of link prediction on triples. During the research, the search for data to create a story out of triples noticed difficulties due to the lack of essential information of entities that make up a story to a certain extent. To prevent such difficulties in the future, the use of link prediction is a solution for the missing entities. Link prediction provides the solution for missing triples in h,t in triples (h,r,t) Link-based prediction could be set based on a rank of candidate entities from a knowledge graph. During a test phase, each test triple (h,r,t) could replace the head/tail entity by all entities in the knowledge graph and rank these entities in descending order of similarity calculated by a score function, Fr. For measuring the evaluation a metric can be: (1) mean rank of correct entities (2) properties of correct entities in a top-ranked entities As for the result, the assumption would be that a good link predictor should achieve a lower mean rank or higher hits.

## References

1. Story Generation with Crowdsourced Plot Graphs Boyang Li, Stephen Lee-Urban, George Johnston, and Mark O. Riedl, 2013, School of Interactive Computing, Georgia Institute of Technology
2. GraphPlan: Story Generation by Planning with Event Graph Hong Chen1,Raphael Shu1, Hiroya Takamura, Hideki Nakayama, 2021, The University of Tokyo, Tokyo Institute of Technology https://arxiv.org/pdf/2102.02977.pdf
3. Yao, L.; Peng, N.; Weischedel, R.; Knight, K.; Zhao, D.; and Yan, R. 2019. Plan-and-write: Towards better automatic sto- rytelling. In Proceedings of the AAAI Conference on Artifi- cial Intelligence, volume 33, 7378–7385.
4. Li et al., ACL 2019, Coherent Comments Generation for Chinese Articles with a Graph-to-Sequence Model https://aclanthology.org/P19-1479
5. Campbell, J. (1973). The Hero With A Thousand Faces (21st ed.). Princeton University Press.
6. Vogler, C. (2020). The Writer's Journey: Mythic Structure for Writers (4th ed.). Michael Wiese Productions.
7. Feifei Xu, Xinpeng Wang, Shanlin Zhou, "Story Generation Using Knowledge Graph under Psychological States", Wireless Communications and Mobile Computing, vol. 2021, Article ID 5530618, 12 pages, 2021. https://doi.org/10.1155/2021/5530618
8. https://semanticweb.cs.vu.nl/2009/11/sem/
9. Beveridge, Andrew. "Character Interaction Network for HBO's 'Game of Thrones' Series." GitHub, 2019, https://github.com/mathbeveridge/gameofthrones. .

# 9 Annex

## 9.1 Extracting Data from Wikidata

Wikidata provides well-structured information about the Game of Thrones characters, therefore, it is a good starting point to identify relations between characters. We first define a set of main characters, for whom we extract more detailed information; whereas side characters that are related to the main characters do not contain any additional info. Then, we manually investigate the interesting properties to include, and note their IDs to use for querying. Using `wikidata` package for Python, defined property relations for the main characters are extracted and processed into Turtle format. All mentioned instances are created with the specified object type.

The knowledge graph can be made more representative by adding more main characters, or adding more detailed information on side characters. However, for simplicity on evaluation, we focus on a small subset of characters.

## 9.2 Community-based Instantiation

The RDF file generated through Wikidata extraction is used to identify closer relationship groups between characters. Using SPARQL, the number of relations between each character is enquired. A weighted `NetworkX` graph is created with the resulting adjacency matrix. The community detection methods available in the `NetworkX` package is applied to explore the communities. Specifically, we utilize the greedy modularity maximization approach to find the optimal community partition of the graph, which incrementally joins pairs of communities that lead to largest modularity until it is not possible to increase modularity.

The communities are stored as a dictionary where the keys are the main characters. The method returns characters within or outside of the community of the hero, based on the type of the character requested. Community detection only considers relations between characters, but a more complex model would be able to represent relations between other instances such as places.

**Visualization** Here we report the visualization of a generated story for each of the methods. The full interactive dashboard is accessible under: `https://datastudio.google.com/reporting/3e1887bb-c02d-4c74-8d74-a14b7399c524`

## 9.3 Code

The code and all the data can be found here: https://github.com/GabHoo/The-Game-of-Thrones-Automyth

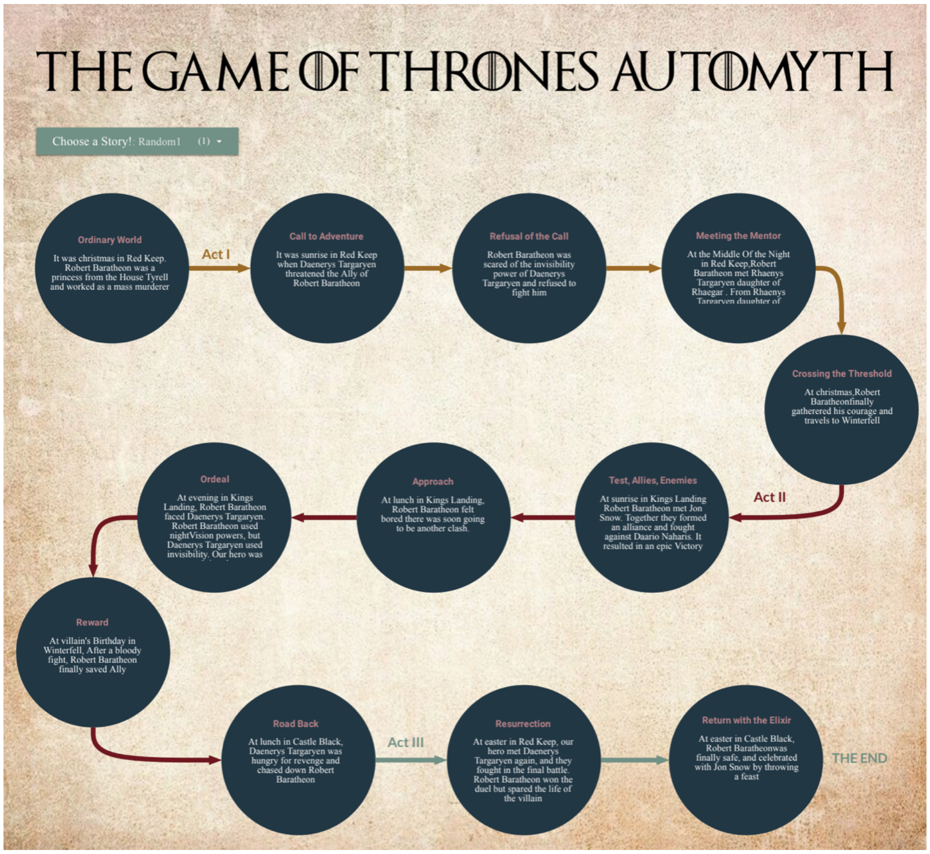


**Fig. 2.** Visualization of a story generated with the Random instantiation method

## 10 Contributions

**Ceren Akkalyoncu** In this project, I mainly worked on the smart instantiation of the character instances. I explored options of integrating Game of Thrones universe knowledge, by searching datasets available online or unstructured information stored in fandom pages. I used Wikidata as a resource to create a small dataset of characters and other instances available in the universe. I used this dataset to detect communities and suggest characters related to a hero. I also implemented the interaction based instantiation model. I mainly described these sections I worked on in the report, and the implementation details under the Annex section.

**Paola Feil** My main areas of work were the research on creative story telling, research on relevant vocabularies, the engineering of the ontology, as well as

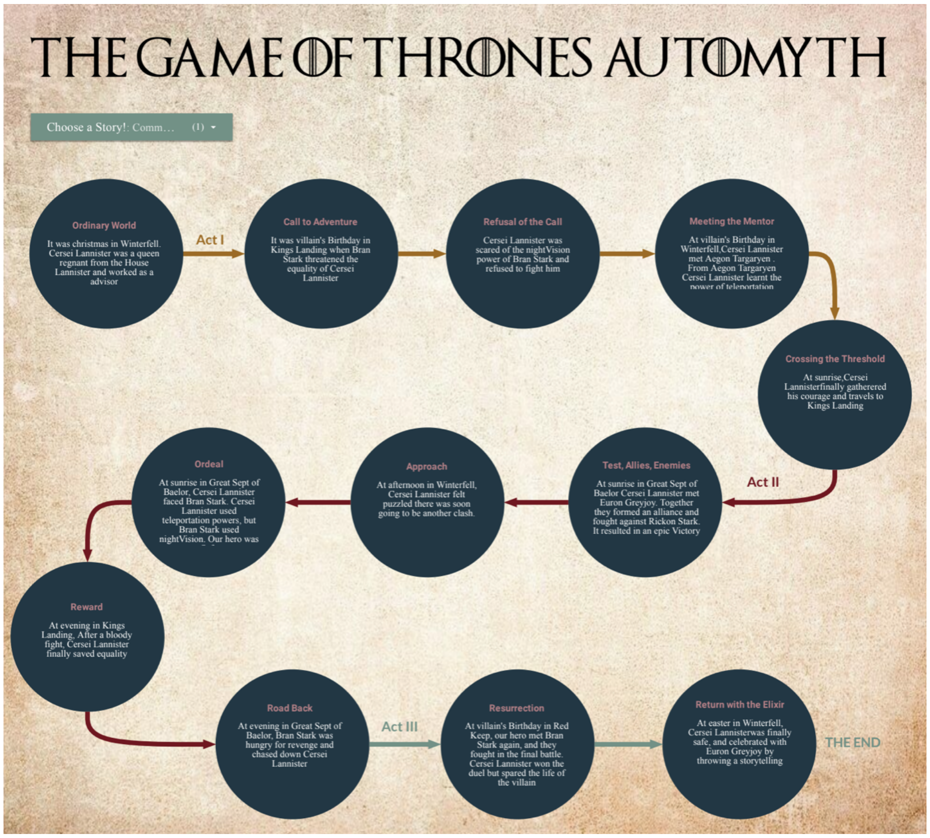


**Fig. 3.** Visualization of a story generated with the Community instantiation method

the translation of the results via Sparql into the dashboard visualisations. Also writing parts of the report.

**Gabriel Hoogerwerf** During the project I contributed initially shape our application and partially desining the event ontology. I also took care of the code and the repos during the whole project , writing the central part of the generator and the random method. I wrote various paragraphs of the report and set up the experiment.

**Redouan Lamkaraf** My main task was conceptualizing the core of project and algorithm. With creation of triples, graphs in graphDB and turtle files for use in the algorithm of instantiations Also research and related work investigation

**Teresa Liberatore** My main contributions to the project can be found in the engineering of the ontology and in the development of the SPARQL queries. I had

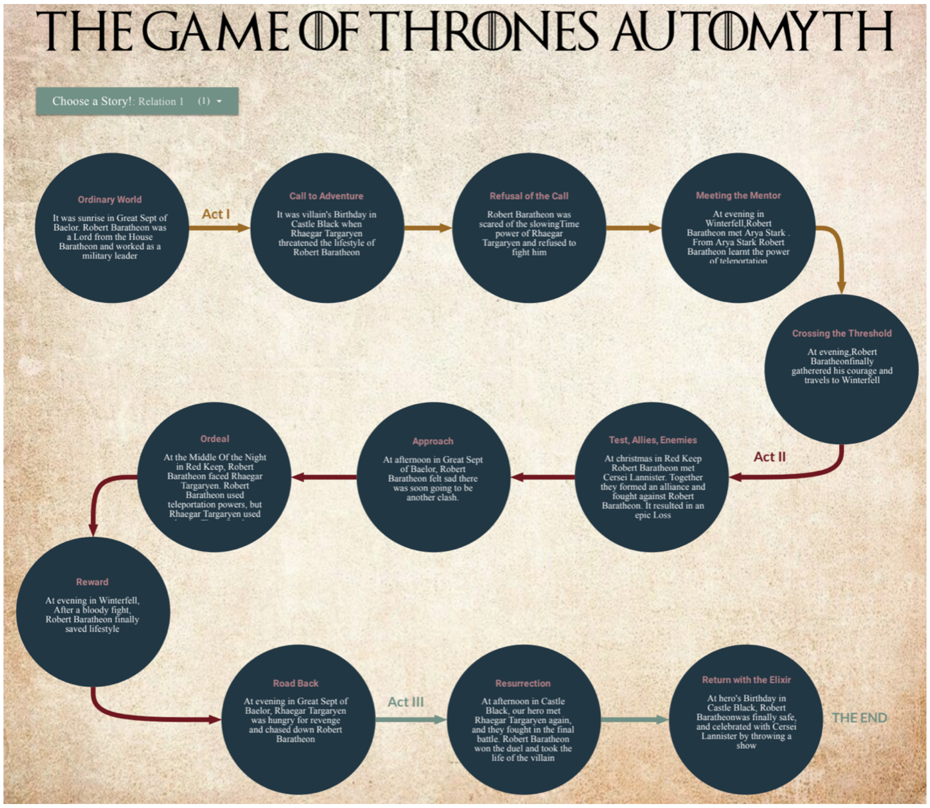


**Fig. 4.** Visualization of a story generated with the Relational instantiation method

minor contributions in the writing of the code in Python and in setup experiment, and I took care of the literature review, while writing various paragraph of the report.