ISLA:
An Algorithmic Approach to Assisted Narrative Planning and Assembly

Djyron Sarroza
Institute of Computer Science
University of the Philippines Los Baños, Philippines
dfsarroza@up.edu.ph

Abstract

Intelligent Story Layout Assistant (ISLA) is a forward-chaining narrative planner based on Stephen Ware's GLAIVE. It constructs story layouts that achieve the author's goals while making sure that *most* steps in the plan have clear motivations. These layouts, or solution plans, are based on a handcrafted knowledge-base of story universe elements. The current output is capable of displaying the underlying motivations of each action, and partial/complete state transition information. The output is also presented in two ways: 1) a relatively human-readable series of paragraphs and 2) a detailed state transition sequence. The objective qualities of the narrative outlines being generated are measured, quantified, and analyzed. Some limited machine learning techniques are also employed in an effort to refine the elements of the knowledge-base. As a whole, ISLA has the underlying data structures needed to potentially further assist the author in fleshing out the produced story layout.

Keywords

Artificial Intelligence, Narrative Planner, Plot Generation

1. INTRODUCTION

There has always been abundant interest in automatic story generation in the field of artificial intelligence. The ability to generate stories on demand has great possibilities in other fields, such as entertainment and education. This thesis is motivated by the need of novice writers to have the tools to assist them in creating complex and convincing story-lines. Managing multiple complex characters, all capable of an arbitrary number of actions, and all interacting in a non-static story telling universe can be a daunting task. Human error is not just a possibility in this scenario, it is an inevitability. Having a software tool which utilizes planning paradigms and working under additional creative constraints[1] aims to alleviate such workload from a human writer.

It is necessary to note early on that this rich narrative planner does not intend to replace human writers, but is designed as a tool that will help authors focus on other creative aspects of creating stories. For example, in lieu of having to keep track of dozens of characters, their actions, and their respective effects – an author using a narrative planner can focus on "fleshing out" the scaffold provided by the planner. The rich narrative planner is merely a guide. Dialogue, personality quirks, vibrant world portrayals, these are only some story elements that require an undeniably human touch.

This paper presents ISLA, Intelligent Story Layout Assistance, a variant of forward-chaining planner which aims to produce a cohesive narrative plan. ISLA models intentionality, conflict, and representation of alternate world-states very similar to GLAIVE [26]. This paper details ISLA's algorithmic approach, data structures used, knowledge-base encoding, and performance measurements.

2. REVIEW OF LITERATURE

The following sections discuss the numerous inspirations for the inner workings of ISLA. With both literary theory and narrative planning having an extensive tradition in the field of artificial intelligence, there is no shortage in raw materials. Computational narratology as described by Mani [15], is a relatively new field of study directly influenced by multiple aspects of technology, linguistics, and even mathematics, among others. This presents an opportunity to discover relationships and connections that will contribute to the construction and improvement of ISLA.

2.1 Transcription

In order to make the computational generation of an expression or artistic endeavor such as a *story* – which is uniquely human – possible, the space must first be transcribed into a form that can be understood by a machine and that allows computational operations and manipulation to be performed on it (i.e. represent it with what is basically recognized as a data structure.)

[1] Not merely "finding a goal state", as is the case with classical planning algorithms

Second, a schema must be established to support efficient manipulation so that the story elements / existents can be instanced into a new literary piece. Ample work has been done on this matter in both literary theory and artificial intelligence that when combined form a solid foundation for a launchpad in building an artificial intelligence capable of computational generation of a narrative.

Parallels between the causal and temporal data structures of some planning algorithms and certain representation of stories used by narratologists were identified by Young [28] . The concepts discussed by Young will resonate through many of his succeeding works, most notable and relevant for ISLA is the notion of searching through a search space of all possible plans. This is opposed to the incremental construction of a single plan.

2.1.1 Computing Narrative

The paper by Miller explores how subjective cognitive frameworks such as narratives can and are represented as procedures and data in creative contexts [18]. Narratives are treated as decomposable data, where readers act as evaluators who operationalize aspects of stories, whereby metrics such as "surprisingness" can be applied to the results.

The aforementioned study further delves into concepts of narrative empiricism, narrative inference, and narrative emergence, which although impressively succinct and curated, offered limited relevance to what will emerge a very specific approach for ISLA. However, insights on how the complex and difficult task of transducing narrative features into software architecture remain valuable assets to further the work on this field.

2.2 Conceptualization

2.2.1 Literary Theory

As evidenced by works such as Propp's formalization of Russian folk tales, there are certain redundancies and patterns present in stories. Although each and every story are not exactly the same, many patterns such as the presence of heroes and villains are quite universal.

Many of the efforts in developing narratology as a discipline are inspired by Vladimir Propp's *Morphology of the Folktale*, and from this seminal work sprung many publications. Too many, in fact, carry inaccurate vulgarizations of Propp's system [1]. Aguirre aimed to place a tool at the disposal of researchers in the fields of folklore and popular culture, hence is a logical place to start to get introduced to Propp's way of breaking down stories into formulaic forms.

Alexander Afanasyev's classic collection Russian Folktales (1855-64) was Propp's basis for his study. From the over six hundred folktales present in the collection, Propp derived his structural model based on the following criteria:

1. All fairytales are constructed on the basis of one single string of actions or events called "functions".
2. Function is significant action or event defined according to its place in the plot.
3. Function, and not theme, motif, character, plot or motivation, is the fundamental unit of analysis.
4. Functions are independent of how and by whom they are fulfilled; from the standpoint of structural analysis, not doers, their method, their motivations or their psychology but the deed itself alone matters.
5. The number of functions available to fairytale-tellers is thirty-one.
6. With (codifiable) exceptions, functions always follow a strict order.
7. Tales are organized into sequences; each sequence is composed of a selection of functions in the appropriate temporal order, and constitutes a narrative episode.
8. Each function is susceptible of realization by different means ("forms of function"): Propp offers lists of the "function forms" that appear in his corpus (but warns that others are possible).
9. Only seven characters are available to fairytale-tellers: hero, false hero, villain, donor, helper, dispatcher, princess (sought-for person) and/or her father.
10. All fairytales are composed of the same functions, though not every function appears in every tale.
11. All fairytales share the same fundamental structure.

Tobias' *Master Plots* [24] is another basis for story structure generalization and can be converted into Propp's formulaic way of describing story sequences. However, ISLA benefited more from Propp's conceptualization of narrative structures. *Master Plots* seems more of a guideline for writers, and is less useful in terms of distilling raw stories into rigorous structures such as schemata. At the very least, Propp verified the hypothesis that stories have recurring patterns that can be manipulated mathematically or algorithmically.

2.2.2 Narratology

Mieke Bal's introduction to narratology [3] provides key insights on the field of study. Specifically with the definition of *fabula* – product of imagination – and *story* – product of an ordering. This insight gives enough distinction to the internal components of a narrative in order to decompose them into elements that can be manipulated my mathematical or algorithmic processes. The text also mentions that *words* are products of the use of a medium, much like programming languages are products of the use technology – both are representations of more fundamental concepts of logic and meaning.

2.2.3 Narratives and Mathematics

One product of the exploration of transcribing stories into data structures is the intuition that the connection between narratives and mathematics runs far deeper. Investigation into theorem provers yielded that narratology does indeed play crucial roles in mathematics – ranging from approaches such as algebraic semiotics [9] are being developed as a way to improve user interface functionality and quality of mathematical proof assistance tools, to studies describing the narrative structure of mathematical text and lectures [7] [12] [27]. These studies are can be described as "narratology for math", where mathematical concepts and elements are mapped to

narratological concepts and elements in order to leverage the advantages of story telling that may be obvious to narratology, but not as much to mathematics. A great example of this are the conveyance of mathematical proofs. Proofs can be described as a linear progression from an initial state, to a final state, with annotations to other resources such as lemmas. However, finding non-trivial proof requires navigating through misconceptions and errors, and these conflicts can arguably positively contribute to the understandability of the proof – much like the journey of a hero, where important conflicts within that journey is explored in order to properly understand the hero's motivations, and add value to the odyssey's resolution.

Narrative analysis using proof assistant tools [5] describes the use of narrative's formal properties in order to leverage upon theorem provers as a way to verify plot structures. This "math for narratology" approach has given credence to the intuition that stories and mathematics are equivalent.

2.2.4 Computational Approach - Narrative Generation Schemata

Previous work by Gervás, León, and Méndez on plot schemata [8] has proved to be a valuable resource for encoding existing plot structures into ISLA's knowledge-base. This study recognized that computationally generated narrative artifacts have a tendency to rely on schema and templates, which can be instantiated into more complex structures. ISLA's vocabulary on describing narrative structures is also solidified by applying concepts detailed in the aforementioned paper. A detailed investigation on the properties of narrative schemata, Simonson's paper [23] on the matter is an excellent resource. The dissertation may prove invaluable in the future specifically on the planned improvements on domain manipulation, problem extrapolation, and machine learning.

Numerous existing attempts at providing an elementary set of patterns for plot are the focus for Gervas' paper. To a large extent, oversimplification of the plot to a very abstract outline is a factor why none of these attempts were accepted as generally valid. Focus on specific aspects of a story, may cause other aspects to suffer when distilling full stories into schemata. However, Gervas was successful in analyzing these schemata as a whole, and a basic abstract vocabulary to describe different plots was defined. This may then provide the basic scaffold of a desired story, regardless of other enrichment which may later be added to it.

Taking advantage of such generalizations, ISLA is intended to utilize the same basic idea as a scaffold, then augmented further with supporting concepts such as character tropes, interactions, goals, conflict, and character evolution. Applying these schemas also has another advantage– they provide additional junctions for ISLA to be improved upon in the future. The current state of ISLA's capabilities does not yet handle explicit representation of individual character personalities. Because of the robust definitions and quantization of story elements by Gervás, León, and Méndez, it was deemed possible that in future works that chapter pattern selection takes into account the type of personalities that current actors have[2].

For example, it may be argued that not all personality types can particularly fit the hero role in all narrative plots. Like a hero that has a lawful-good personality type will not make much sense in a Revenge plot. Properly matching plots with personalities may produce more desirable traditional output; conversely, intentionally diverging from this mechanic may produce results that do not conform to traditional patterns. Either way, it imbues ISLA more direct control to her story generation processes.

2.2.5 Computational Approach - Encoding the Domain

As a way to have a unified syntax for representing planning problems for the annual International Planning Competitions (IPL), the Planning Domain Definition Language (PDDL) was conceived. The main inspiration for the PDDL language was the Stanford Research Institute (SRI) Problem Solver (STRIPS) language from the early 1970s, developed at SRI International.

PDDL was a new language specifically crafted to encode two things: the planning domain, and the planning problem. In its basic form, the following are the components of a PDDL planning task:

Domain
- Objects: Distinct entities that populate the story-telling universe
- Actions: Ways of changing the state of the story-telling universe
- Predicates: Properties of objects

Problem
- Initial state: The state of the world that we start in.
- Goal specification: Things that we want to be true.

(1)

2.2.6 Computational Approach - Planners

Forward-chaining planning is perhaps one of the more intuitive planning approaches: beginning with the initial state, traversing the state-space by applying actions to the encountered states until a state is reached that satisfies the target condition. The strategy used to enter each state can be calculated by following the course of actions performed to travel from the initial state to the state in question: in sequence, these actions form the plan that leads to that state.

Theoretically, forward-chaining state-space traversal involves a directed representation of the graph: nodes represent states; the edges represent the applications of action between then. Nonetheless, such a structure would cause cycles to occur (corresponding to redundant parts of the plans) if a path

[2] Actor and existent definition is planned to be indirectly affected by the user's desired parameters, or is directly explicitly chosen.

were found back to itself from a given state. Practical implementation often demand that the search is constrained to ensure that a tree representation is sufficient: this is done by memoising the states encountered as the search progresses, and not allowing changes to those states encountered previously. Exhaustive and unguided forward-chaining search on all but the smallest of planning issues is quite infeasible. Two methods are widely used in practice to increase search efficiency: using a heuristic to direct search; and pruning the search tree.

There are numerous heuristics discussed in Cole's paper [6] such as memoization, loop-checking, and pruning – all of which were incorporated into ISLA. A version of the forwards reachability analysis from the initial state to reach the goal facts was later included (see sec. 3.2.4.5). This heuristic calculates which candidate action can lead to more predicates being true in the future. The assumption is: the more goal predicates which are TRUE in any given state, the closer it is to a goal state.

2.2.7 Computational Approach - Narrative Planners

There are many aspects which determine if the audience a story as good. Many of these aspects, such as the degree to which the audience empathizes with the protagonist, are subjective in nature. Other aspects over a wide variety of genres seem to be more universal.

The comprehension of stories requires the audience to perceive the causal connection of story events and to infer character intentionality. Accordingly, the two narrative qualities that we concentrate on in this narrative generation research are rational causal progression and believability of characters.

2.2.7.1 POCL - Partial Order Causal Link

Partial Order Causal Link (POCL) plans have proved to be powerful data structures for representing stories as they form the events of a story directly along with the casual and temporal connections between them. Such program data structures can also act as metaphors for mental models formed by people who read or watch a narrative.

POCL preparation can be interpreted as a refinement search, in which a partial plan is slowly repaired or improved until either the plan is complete (and executable) or inconsistent (and unrepairable). A partial plan is annotated with flaws in this process; each flaw indicates a specific problem with the partial plan which needs to be repaired.

2.2.7.2 IPOCL - Intent-based Partial Order Causal Link

Reidl and Young [19] defines a method of narrative generation that models the creation process of fictional narratives as a search-based planning process. The resulting item – the plan – is a summary of the series of acts temporarily ordered to be performed by story world characters. This design tells a story when it's implemented or made into natural language. Plans have been found to be good computational representations for narratives, since plans encode core narrative attributes: behavior, temporality, and causality [28].

Unfortunately, solving the planning problem also does not solve the problem of narrative creation, since (non-narrative) planners do not understand many of the narrative's logical and artistic properties. In particular, planners do not consider believability of characters (or character actions). Reidl and Young defined a novel refinement narrative planner – the Intent-based Partial Order Causal Link (IPOCL) planner – that, in addition to creating causally sound plot progression, provides explanations for intentionality of character by (a) defining possible character objectives that explain their behavior and (b) creating plan frameworks that explain why those characters commit to their objectives.

The IPOCL's primary concern is the generation of a fabula. It is assumed that the sjuzet can be created from this in another process. Two attributes of narrative that IPOCL focuses on the narrative generation are 1.) logical causal progression and 2.) character believability, hence the emphasis on the goal-oriented nature of the actors present in the planning domain.

Based on a class of planning algorithms called Partial Order Causal Link (POCL) planners, the IPOCL creates narratives that, from a reader's point of view, resemble more closely the results of a simulation narrative generation system in terms of character intentionality. In particular, IPOCL is a modification of the current search-based planning algorithms to promote the intentionality of characters regardless of the intent of the author. The goal is to create narratives through a deliberative process so that characters appear to the audience in order to shape expectations and behave as if they were replicated to achieve such intentions. In this way, IPOCL may generate narratives that both have rational causal progression, meaning that they achieve states of outcomes suggested by the author, and have credible characters.

A good heuristic that ranks believability positively impacts the probability that a plan can be found. However, it is still possible for a planner to commit to a shorter, albeit less believable plan, as opposed to a longer/more expensive, but believable plan. Their conclusion is that the fabula generation problem is sufficiently different from a classical planning problem that it [narative planning] can benefit from new definitions and metrics for plan completeness.

Since actor believability is in part due to intentionality, a story is more likely to be considered believable if actors appear to be motivated by individual goals. Reidl et. al. mentions that multiple actors with different (conflicting) goals can be a complication, not to mention the difference in goals between the story characters and the human author. Fortunately, the Conflict Partial Order Causal Link directly addresses this.

2.2.7.3 CPOCL - Conflict Partial Order Causal Link

Conflict Partial Order Causal Link (CPOCL) [25] is an IPOCL* extension that specifically illustrates how characters can contradict each other in order to achieve their goals, which are the core of narrative conflict. CPOCL enables the failed or partially failed thwart subplans, addressing a key IPOCL limitation that Riedl and Young [19] identified.

A good way to understand the difference between these algorithms is by comparing their solutions to the same prob-

lem. Consider this scenario: two guys, Abe (A) and Bob (B), are interested in marrying the same girl – Cat (C). While Bob had previously purchased an engagement ring (R), Abe and Cat had agreed to be married, as per the author's intention.

POCL's shortest solution would start with Bob giving Abe the ring. Though this scheme is causal, Bob's supporting his competitor doesn't make sense. This problem would not be solved by IPOCL*, because Abe and Bob can't both meet their goals. CPOCL is able to find a solution that guarantees causal well-being and good motivation for character.

2.2.7.4 GLAIVE

GLAIVE is a state-space heuristic search planning algorithm that explicitly explains intentionality of character and describes alternative worlds to promote thinking regarding phenomena such as conflict. It is based on Hoffman and Nebel's Fast-Forward planner [10], but similar to both IPOCL and CPOCL algorithms, GLAIVE also keeps track of the causal history of each proposition.

GLAIVE solves the problem of intentional planning described by Riedl and Young [19]: A valid plan is one that achieves the goals of the author but is only composed of steps that are explained in terms of the individual goals of the characters who take them. Ware and Young [26] have expanded this issue to include approaches in which some characters' plans fail. It does this considering that there are multiple agents which sometimes collaborate and sometimes clash as they are directed towards the objective of the author by an unseen puppet master.

GLAIVE is a state-space planner, that is, it begins at the initial state of the problem and takes steps that alter the state until it finds a state in which the expectations of the author are valid. The search space can be described as a directional tree. A node in the tree represents a state; an edge $n_1 \rightarrow n_2$ represents applying the effects of step s to the node $n1$ state to create the new node $n2$ with a different state. In practice, a node also represents a plan composed of steps taken from the root to that node on the path. The tree's root is the initial problem state, and contains an empty plan. GLAIVE explicitly tracks how the preconditions of later steps are met by earlier steps, a set of goals or intentions for the actors, and a set of unexplained steps.

2.2.8 Other Related Systems

One of earliest software that generates stories is Meehan's TALE-SPIN [17]. His approach is very similar to GLAIVE – story characters are given goal, and searches are made, with the help of planning rules, in order to fulfill these goals. The traces of these searches are used as the basis for the output narrative.

A goal-based model of character personality is explored by Bahamón et. al. [2]; character belief as modeled by Shrivani et. al. [22] also plays a crucial role in describing how characters act based on their personal knowledge of the world – which may or may not be true. Both papers influenced how ISLA's planning mechanisms are designed, although goal-based character models have a more direct impact. The tree-like data structures required to describe alternate world-states are already present, and only the addition of *epistemic edges* are required to fully represent character/actor beliefs.

Assistive tools such as LISA: Lexically Intelligent Story Assistant [20] also exists in a different branch of the narrative software taxonomy. LISA focuses on real-time feedback specifically for lexical inconsistencies in the story.

Narrative planners are also used as tools for natural language story scripts [21]. Sanghrajka et. al. describes a tool that assists writers with the aspects of "story world bookkeeping", which is especially critical when creating new stories within already-established story-telling universes.

2.3 Plot Evaluation

2.3.1 Challenges with Objective Evaluation

Narrative, like most objects studied by humanists, is a subjective, culturally-dependent phenomenon. As such, attempts to model it fall victim to the problem of validation in the humanities; each reading of a narrative serves as one of many possible readings by one of many possible readers [18].

Although the question of computational creativity evaluation is non-trivial, Jordanous presented an impressive attempt to create a standardized procedure to evaluate creative systems [11]. Researchers who tackle relatively ill-defined and definitely highly subjective topic of creativity research, tools such as the Standardised Procedure for Evaluating Creative Systems makes creativity more tangible to work with.

2.3.2 Tellability

Capturing the quality measure of plot, independent of discourse is a necessity for systems such as ISLA. Tellability, a concept introduced by narratologists to describe the potential suitability of a configuration of events to be rendered into discourse, presents itself as candidate for such a measure of plot quality [4]. If a narrative planning system is to implement measuring tellability, it must possess 1) a character architecture that represents beliefs and intentions as a propositional knowledge base, 2) capture of represent well-formed plots and3) implicit or explicit representation of character conflict.

The presence of complex plot units are offered as the key component of the measure of tellability. In Lehnert's work [14], she introduced several intra-character and cross-character complex plot units. Those units are by no means complete and comprehensive, since there there is no theoretic limit to the size of these units and more can be derived.

In fig. 2, the vertices are denoted as:

- +: an internal or external event that is appraised with a positive emotion by the character,
- –: an internal or external event that is appraised with a

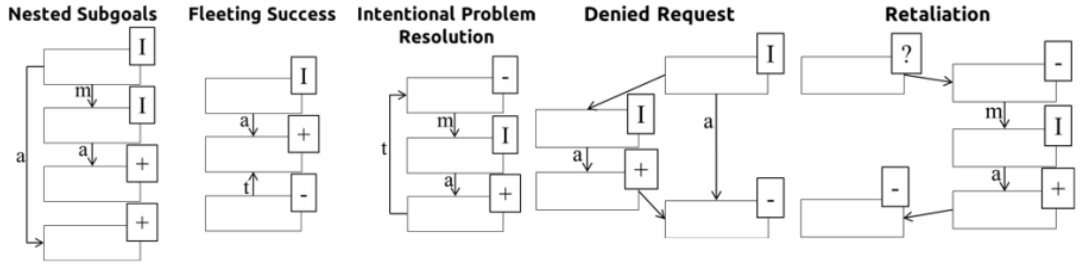

Figure 2: Examples of complex plot units; '?' is a wildcard for arbitrary vertex type.

negative emotion by the character,

- **I**: a neutral internal event that denotes the setting of an intention by the character

And these vertices can be connected by five types of edges:

- **m**: a motivation edge can lead from any type of vertex to an I vertex and denotes that the former event causes the intention,
- **a**: an actualization edge can lead from an I vertex to a + or - vertex and denotes that the intention causes the latter event,
- **t**: a termination edge can connect either two non-neutral, or two I, vertices and leads from the event that terminates an affective/intensional state to the one that is terminated (respectively supplanted in the I case)
- **e**: an equivalence edge can connect two non-neutral, or two I vertices, and denotes that an event has multiple affective evaluations, or two intentions are congruent; the edge direction is anti-temporal,
- **a cross-character edge** is temporally directed and can connect all types of vertices, so long as they belong to different characters' subgraphs; it denotes that an event is affecting several characters, prominently also including speech acts.

2.4 Theoretical Framework

The equivalence of narratives and mathematics, specifically in logical structures, establishes the groundwork for the ideas that will eventually become ISLA. Tools to succeed in this endeavor are already present – from mapping narrative to mathematical elements (and vice versa!), to condensing corpora to schemata, to the numerous algorithms designed to tackle narrative planning.

Investigating a broad range of narrative-related systems have also help provide a good understanding on how to proceed with developing ISLA. Knowing where the most influential systems, such as GLAIVE, place in the taxonomy of narrative have given critical insight on what can be done, both within the current bounds of this thesis' objectives and future work in the field. In order narrow down the candidate algorithmic approaches, an impressive survey on existing story generation techniques by Kybartas and Bidarra [13] is examined. Notably, they have described distinct classification range for different story automation approaches.

Perspectives provided by existing works have provided the tools to make ISLA possible. Riedl, Young, and Ware's work on narrative planner algorithms provided the backbone for ISLA's own implementation of representing logical causality, with intentionality and conflict. Shrivani et. al. highlighted specific concepts such as belief modeling, emphasizing that there is still ways to grow systems like ISLA. Berov and Lehnert's *tellability* concept has provided a concrete way to measure the quality of the narrative outline output.

3. ISLA: INTELLIGENT STORY LAYOUT ASSISTANT

3.1 Knowledge-base Encoding

3.1.1 Domain-Problem

In order to properly represent the narrative universe, the Planning Domain Definition Language (PDDL) was chosen to encode information about world-states, allowable actions, actor intentionality operators, object/existent descriptors, author and character goals, and other metadata. ISLA utilizes a knowledge-base encoding language based on PDDL, but with a number of customizations, such as explicit intentionality operators, conditional statements, and author-related statements.

Allowable actions are controlled mostly by action parameters and precondition input. All *precondition* predicates must be satisfied before the action *can* be applied to a world-state. The *effect* section denotes which state predicates are applied (or removed in case of a *not* operator). *Parameters* and *agents* facilitate ease of parsing and action validity evaluation.

As for intentionality restrictions, only the *actor* supertype is allowed to have intentions. If, for example, we wish to force certain event where there are no valid actors that can be involved (e.g. forces of nature, accidents, etc.), intentions are attached to the mandatory *author* type.

$$
\begin{aligned}
&\textit{Domain} && \langle T, P, A \rangle \\
&\textit{where} && T \quad \text{is a set of object/existent type definitions} \\
& && P \quad \text{is a set of state predicate definitions} \\
& && A \quad \text{is a set of allowed actions}
\end{aligned}
\tag{2}
$$

$$
\begin{aligned}
&\textit{Problem} && \langle E, S_0, S_f \rangle \\
&\textit{where} && E \quad \text{is a set of object/existent instances} \\
& && S_0 \quad \text{is the initial state} \\
& && S_\mathrm{f} \quad \text{is the goal state}
\end{aligned}
\tag{3}
$$

3.1.2 Narrative Structures

Plot structures used by ISLA are based on work by Gervás et. al. [8]. This provided a convenient source of standardized plot structures distilled from multiple plot descriptions. In order to properly encode these plot structures, custom narrative structure components were crafted which facilitate key plot elements and domain state predicate behavior. This encoding approach enables ISLA to have a non-deterministic output when tasked to produce a chapter pattern instance.

In the current state of ISLA, she is using a simple hierarchy to describe the current storytelling universe:

Narrative Chapter Structure Hierarchy
Domain
+– Chapter Patterns
| +– Sequence Terms
+– Predicate Descriptor Definitions
+– Object Name Lookup
(4)

The domain defines the base rules of the storytelling universe: what kind of objects can exists, what are their relationships between each other, and how they interact. Story patterns are frameworks that define how the story outline should progress. This is achieved by using sequence terms (which is functionally a "story chapter") and chapter patterns.

3.1.2.1 Sequence Terms and Chapter Patterns

Sequence terms is a fundamental knowledge-base component that defines specific intentions, or sub-goals, that are required to be fulfilled per story chapter. In other words, a sequence term represents a story chapter which the desired result or goals are known. The specific steps on how the story outline instance will take, however, is determined during *planning phase*. This means that ISLA is not guaranteed to produce the exact same sequence of actions when creating multiple plans for the same sequence term.

To have further control over the output narratives, ISLA also employs chapter patterns. The chapter pattern knowledge-base component represents the general structure of the whole narrative, which is composed of an ordered sequence of sequence terms. This allows the definition of common and special component sequence terms which can be arranged in any arbitrary way to conform to any story or narrative pattern paradigm.

3.1.2.2 Predicate Descriptor Definitions

Another supporting knowledge-base component is the predicate descriptor definitions. This augments the domain state predicate definitions by adding more metadata such as likelihood to spawn at the initial state and cardinality. For example, the predicate *at* two entries in the predicate descriptor definitions: one for each of its parameters, the *object* and the *place*. The *object* parameter has a cardinality of one-to-many relative to *place*, which means that an object can only be at one place at a time, but places can have multiple (or no) object located on it.

State Predicate - "At":
(at (?object - object) (?place - place))
(5)

3.1.2.3 Object Name Lookup

ISLA employs a simple object naming lookup. Database tables store arbitrary object names, and ISLA chooses object instance names randomly. Internal mechanisms in ISLA prevent names from being chosen multiple times, as this can cause confusion during the planning phase.

3.1.3 Others

As of time of writing, individual character personalities have not yet been implemented. Numerous code junctions have been identified for future work on this aspect. For example, individual action pools can be assigned to each actor, which can be filtered/augmented based on that actor's personality. Run-time initial state refinement and state consistency mechanics may also benefit from the additional information maintained in the predicate descriptors.

3.2 Core Planner

Perhaps the single most important piece of code in ISLA is the core planner module (CPM). The CPM takes a fully defined domain-problem object instance, and produces these key data structures as output:

- Goal Graph
- Plan Graph

The solution paths, which represents the actual relevant per-chapter solutions, are built while growing the plan graph. The succeeding sections provide more details on how solutions are discovered using the two aforementioned data structures.

3.2.1 Based on GLAIVE

Heavily influenced by the work of Ware et. al. [26], the core planner module is very similar to GLAIVE. ISLA's CPM is a forward-chaining planner and functions like so: beginning with the initial state, the search progresses from the initial state towards *a* goal state, applying appropriate actions along the way. Formally, it is a search through a landscape where each node is defined by a tuple < *S*,Π >. S is a world state comprised of predicate facts and Π is the plan (a series of ordered actions) used to reach S from the initial state. The initial state is denoted as: $< S_0, \{\} >$

Directed edges between pairs of nodes in the search landscape correspond to applying actions to lead from one state to another. When an action A_n is applied to a search space node $< S, \Pi >$ the node $< S', \Pi' >$ is reached, where S' is the result of applying the action A_n in the state S, and Π' is determined by appending the action A_n to Π. As such, $\Pi = A_0...A_n$. Forward-chaining search through this landscape is restricted to only considering moves in a forward direction: transitions are only ever made from a node with plan Π to nodes with a plan Π' where Π' can be determined by adding (or 'chaining') actions to the end of Π.

3.2.2 Goal Graph

To facilitate the search for solutions given a set of known intentions, a structure known as a *goal graph* is required. The goal graph will take part in constructing the actual plan graph by providing a heuristic when choosing the next viable steps from the current *plan graph* node state.

The goal graph is a layered tree-like directed graph with state predicates as nodes and action instances as edges. Construction begins at the "goal" layer, which is comprised of state predicates from all known *intentions*. Intentions are special state predicates denoting that an actor *A* intends state predicate *P* to be true at some point in the future. As such, it represents one of *A*'s possibly many goals in the narrative.

$$\begin{aligned} &intends(A, P) \\ &intends(Author, has(A1, Item1)) \end{aligned} \tag{6}$$

From the goal layer, the goal graph is actually grown backwards – with the parent's edges denoting an action that will resolve into one of (in this case, an intention) the state predicates as its child. In ISLA's implementation, progression from one goal graph node to another is somewhat loose.

Let P_{effect} be the set of effects for some action A_n, $P_{precondition}$ be the set of preconditions for some action A_{n+1}. The positive goal graph chaining rule is as follows:

Goal Graph Chaining Rule

$$\begin{aligned} if \quad & |(P_{\text{effect}} \cap P_{\text{precondition}})| >= |P_{\text{precondition}} - \text{t}| \\ then \quad & A_{\text{n}} \rightarrow A_{\text{n+1}} \end{aligned} \tag{7}$$

The *goal graph* chaining rule only checks how compatible A_n's effects are with A_{n+1}'s precondition by counting their intersecting terms. The rule does not take into account any configuration of the *plan graph*'s current state. Therefore, any path[3] in the goal graph leading to some goal node G_{goal_a} is a *potential* series of plan actions that can fulfill the goal predicate represented by G_{goal_a}

ISLA maintains one massive goal graph, as opposed to GLAIVE, which maintains several smaller goal graphs – one for each actor. The hypothesis is that having one goal graph for all active actors will inherently encode possible interactions with multiple actors and goals.

[3] As illustrated in figures 9-15

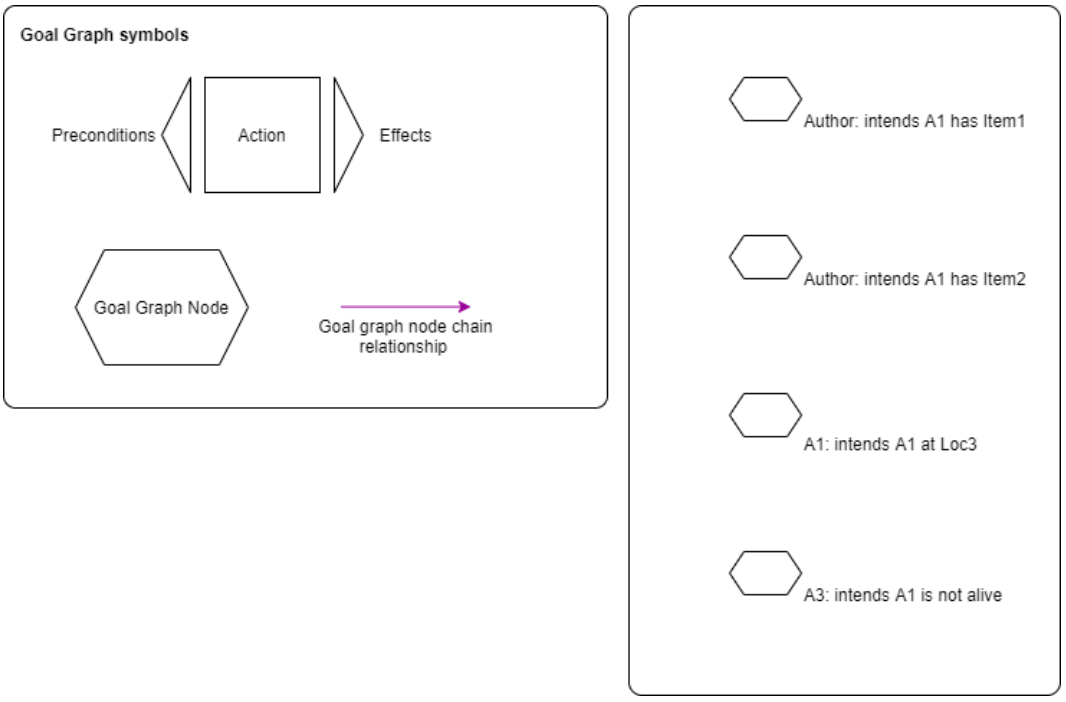

Figure 3: Goal Graph initiation – Setting of all known goals/intentions as nodes in goal graph goal layer.

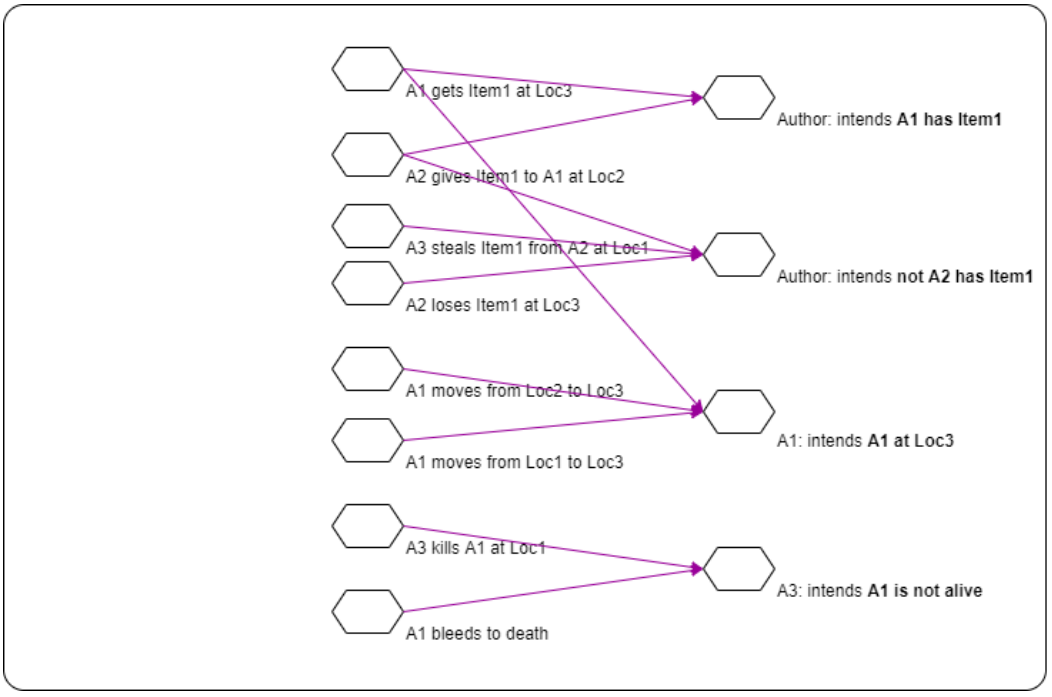

Figure 4: Goal Graph progression – Growing the goal graph one layer at a time. The nodes in preceding layers are possible actions that can be taken in order to potentially fulfill the intentions in the goal layer. These nodes only take into account how compatible the current action is (e.g. A1 gets Item1 at Loc3) with another goal node (Author intends A1 has Item1) based on the goal graph chaining rule (eq. 7)

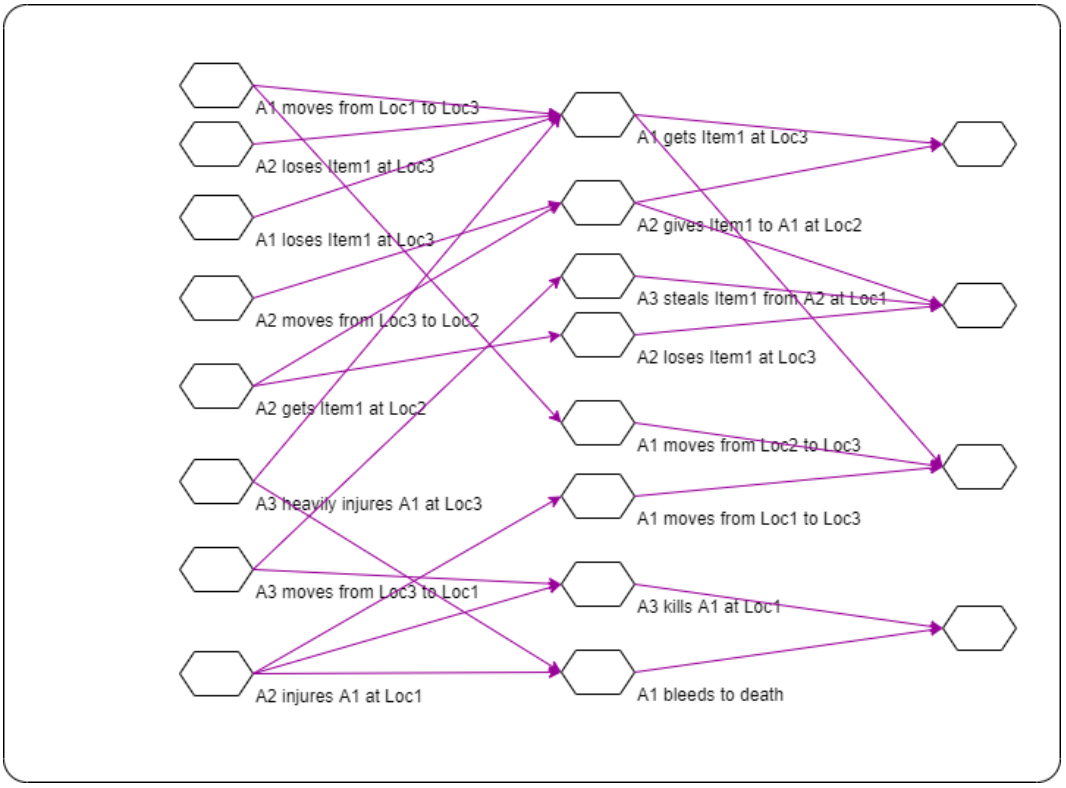

Figure 5: Goal Graph progression – The goal graph is grown until it reached an arbitrary depth.

3.2.3 Plan Graph

The actual data structure that contain possible solutions is the plan graph. The plan graph is also a tree-like directed graph with full states – as opposed to mere predicates – as nodes, and actions as directed edges. It has a root node denoting the current state of the narrative S_0. From this starting point, candidate actions are chosen using the goal graph as a heuristic guide. The plan graph grows in a breadth-first manner. If certain conditions are met (e.g. solutions have been found and unexplained steps/actions are below threshold), plan graph construction is halted. Valid plans are then extracted as paths from the current state to solution state/s.

It is noteworthy that a breadth-first-search algorithm is used to explore the plan graph since it is often beneficial to find the shortest solution path, specifically when solving for chapter goals quickly

3.2.3.1 Backward Reasoning

The goal graph assists in growing the plan graph by way of an estimated sequence of actions. This sequence is derived by creating a backward reasoning path, with an arbitrary length *l*, from some goal graph node G_n towards some goal graph node in the goal layer G_{goal_a}. This marks a key difference between GLAIVE and ISLA; while GLAIVE considers which character's goal is closest to being achieved – as it maintains a separate goal graph for each active actor, ISLA chooses at random. This junction in ISLA's algorithm can improved in future works to fine-tune the range of the final solutions. For example, certain characters may get special priority because the current chapter is focused on them, or all characters are alternated equally so all their goals progress at roughly the same rate.

3.2.3.2 Intentional Paths

In order to represent character intentionality, ISLA employs relaxed intentional paths for step explanation based on GLAIVE's more rigid implementation [26].

Definition 1. A *causal link* $A_s \xrightarrow{p} A_t$ exists from step *s* to step *t* for some predicate p, if and only if step *s* has effect *p*, and step *t* has precondition *p*.

Definition 2. An *intentional path* for some character *c* and some goal *g* is a sequence of *n* causally linked actions $\langle A_1, A_2, ...A_n \rangle$ such that:

1. Character c consents to all actions
2. *c* intends *g* is true before action A_1 and true until action A_n
3. Action A_n has effect *g*
4. For *i* from 1 to $n - 1$, there exists a causal link $A_i \rightarrow A_{i+1}$

Definition 3. A step *s* is *explained* if and only if:

1. $\forall$ consenting character *c*, *s* in on an intentional path for *c*.
2. All other steps on that intentional path are explained.

Definition 4. A valid intentional plan is a sequence of *n* actions such that:

1. Each action A_i is causally link to action A_{i+1}
2. After action A_n is applied, the author's goals are satisfied
3. The number of unexplained steps does not reach some unexplained step threshold

Any candidate solution plan found during a plan graph search would still need to be a valid *intentional plan*. This means that the algorithm will not terminate upon locating a node containing a state that satisfy the author's goal – it will continue to locate partial solution nodes that will explain a sufficient number of steps already present in the candidate solution plan. For the sake of performance and flexibility, ISLA is equipped to control the unexplained step threshold. If the author desires that each and every step be explained (i.e. all character actions are properly *motivated*), then the threshold can be set to 0%; otherwise, the author can set this threshold closer to 100%, which may be described as complete chaos – with characters apparently in the mercy of ISLA choosing at random what they are to do next.

To elaborate, consider the following example. Given the follow author goals [eq.8] and actor intentions [eq.9], ISLA produced a solution plan [eq.10].

Author's goal:

$$\begin{aligned} &and(&& (I_{a1}) \\ &\quad hasbeenproposedto(talia) \\ &\quad hasproposed(rory) \\ &) \end{aligned} \tag{8}$$

Actor intentions:

Talia - person

(intends talia (not (single talia))) (I_{t1})

(intends talia (not (at villagesquare talia))) (I_{t2})

Rory - person

(intends rory (happy rory)) (I_{r1})

(intends rory (not (single rory))) (I_{r2})

(intends rory (hasproposed rory)) (I_{r3})

Sja-anat - monster

(intends sjaanat (not (single sjaanat))) (I_{s1})

(intends sjaanat (marriedto heather kyle)) (I_{s2})

(9)

There are 3 active actors in this scenario, with distinct intentions of their own. Each action is attempted to be explained by exploring the plan graph space for states which satisfy any of the actor's intentions. In [eq.11], step/action π_1 can be explained if ISLA encounters an action *marry(sjaanat, rory, <anyplace>)* which results in some state $S_{f'}$. In practice, however, steps are not always easy to explain – the suc-

ceeding section will discuss how the chapter chainer module in conjunction with knowledge-base elements work together to solve this subproblem.

Found solution:

$$
\begin{aligned}
&fallsinlovefromafar(sjaanat, rory) && (\pi_1)\\
&travel(rory, grassfields, villageproper) && (\pi_2)\\
&travel(talia, market, villageproper) && (\pi_3) \quad (10)\\
&fallsinlovefromafar(rory, vince) && (\pi_4)\\
&proposeforlove(rory, talia, villageproper) && (\pi_5)
\end{aligned}
$$

Steps π_2, π_3, and π_4 are easily explained. [Fig.8] shows how step π_2 was explained when the state $S_{f''}$ is encountered. Take note that only preceding actions performed by *consenting active actors* whose intention/s are fulfilled by state $S_{f''}$ are explained. Hence for the aforementioned example, only actions proposeforlove(rory, talia, villageproper) and travel(rory, grassfields, villageproper) are explained, *not* fallsinlovefromafar(sjaanat, rory), even though it is part of the causal link that spawned state $S_{f''}$ on this particular "alternate reality".

Step explanation:

(π_1)

fallsinlovefromafar(sjaanat, rory)

travel(sjaanat, underriver, grassfields)

proposeforlove(sjaanat, rory, grassfields)

...

marry(sjaanat, rory, <anyplace>)[Satisfies I_{s1}]

(π_2)

travel(rory, grassfields, villageproper)

proposeforlove(rory, talia, villageproper)[Satisfies I_{r3}]

(π_3)

travel(talia, market, villageproper)[Satisfies I_{t2}]

(π_4)

fallsinlovefromafar(rory, vince)

...

marry(vince, rory, <anyplace>)[Satisfies I_{r1}, I_{r2}]

(π_5)

proposeforlove(rory, talia, villageproper)[Satisfies I_{r3}, I_{a1}] (11)

3.2.4 Plan Graph Sub-algorithm Comparison

During the course of ISLA's development, different approaches are implemented and tested in an attempt to improve performance. In this section, we describe the difference sub-algorithms used in the coreplanner; specifically on how the coreplanner chooses the next action to take from any given plan node.

3.2.4.1 Full Random

On full random setting, ISLA just chooses randomly from *all* legal actions. Only the action's ability to be executed on the current state is tested by ISLA (legality). Results using the full random setting is included as the control set in all sub-algorithm comparison experiments.

3.2.4.2 Goal Graph Paths

Based on an interpretation of the GLAIVE algorithm, this is the initial sub-algorithm that was equipped to ISLA's coreplanner. ISLA utilizes the paths present in the goal graph[4]– a pre-generated data structure that represents the potential paths towards story goals. ISLA does this by searching the goal graph for a node with highest potential match to the current plan node, then tracing a path to a known goal node.

3.2.4.3 Directed Random

It has been observed that when using goal graph paths (section 3.2.4.2) alone, ISLA often gets lost by "travelling too often". Since travelling from one location to another is one of the easiest[5]actions to enact, ISLA's choices tend to get flooded by different combination of travel actions. This results in a too-large plan graph with little state changes that progresses (or potentially progresses) the story. To alleviate this issue, the directed random approach is formulated.

This 'directed random' approach is an attempt to improve the performance by guiding ISLA through manually defined buckets and using two different selection criteria.

- Action pool buckets
  - Actions present in the goal graph (GG-Actions)
  - Actions not present in the goal graph (GG-Complement-Actions)
  - Travel actions (Travel-Actions)
- Selection criteria
  - Highest Δ(Delta)
  - Random

The highest-delta selection criteria evaluates the score of each action in the bucket according to how much change the action will have an effect on the current state, and then selecting the action with the highest score. This approach effectively reduces the chance of ISLA stalling during the cultivation of the plan graph.

[5]Just needs the creature to be alive, and that the source is not the same as the destination

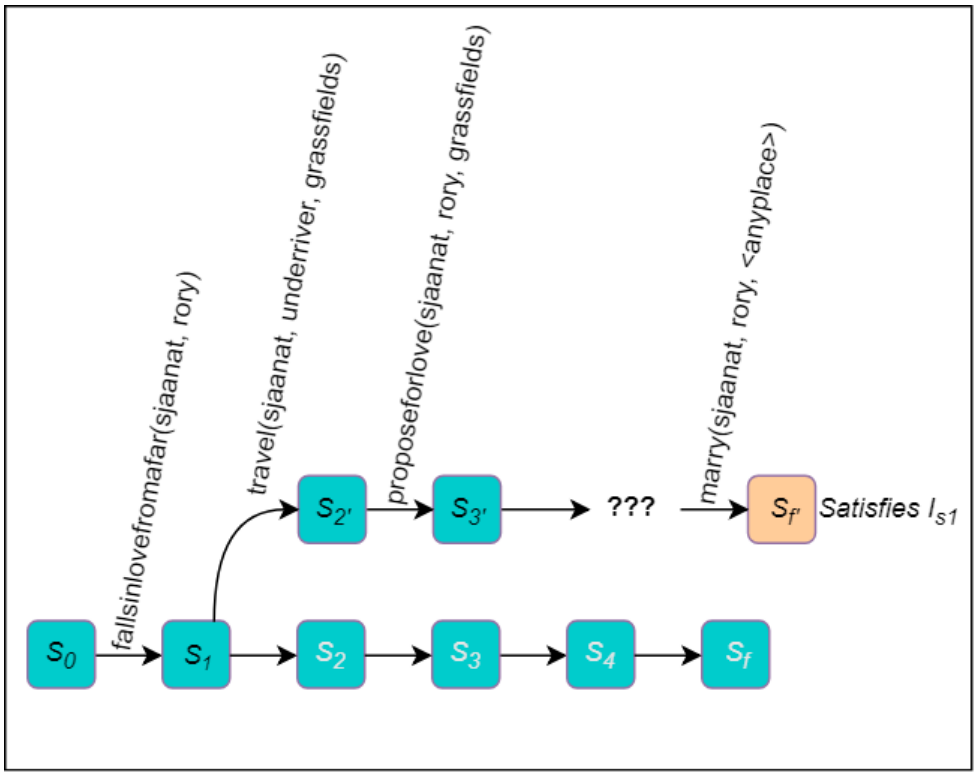

Figure 7: Plan Graph - Attempting to explain step π_1

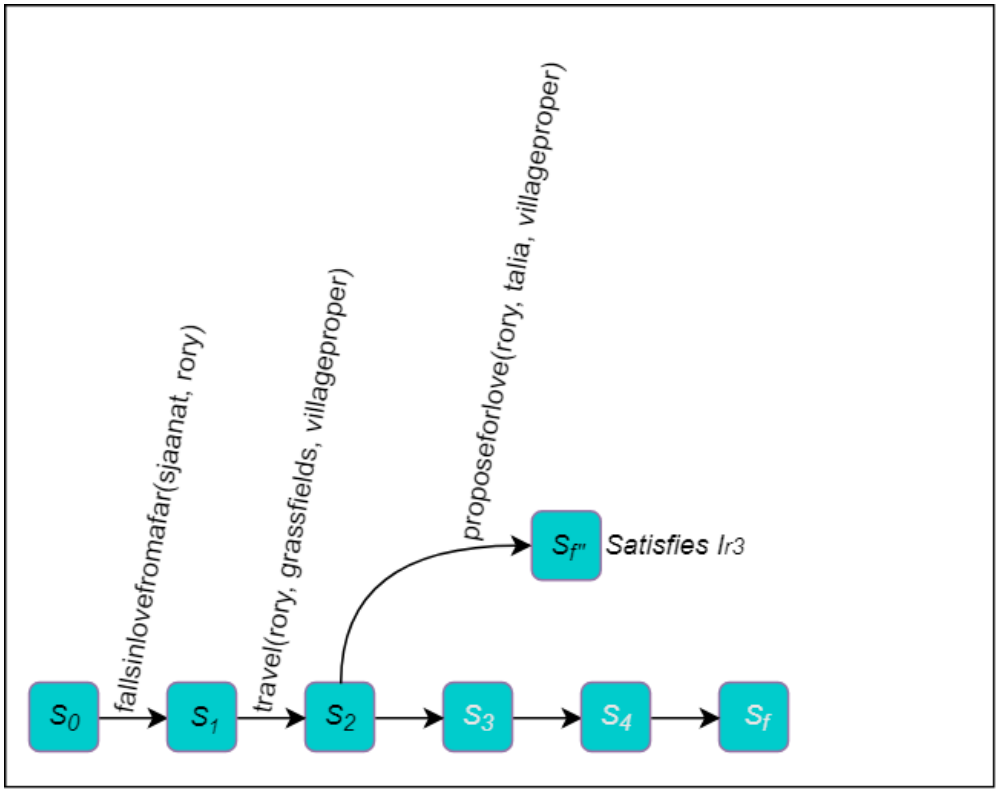

Figure 8: Plan Graph - Attempting to explain step π_2

To further increase ISLA's chances of finding the most effective action, ISLA also has several passes or attempts at this selection process. Each pass has a different action pool, and is basically a way to encode the "trying different things when one thing did not work" heuristic into ISLA.

1. Highest delta among GG-Actions
2. Highest delta among Travel-Actions
3. Highest delta among GG-Complement-Actions
4. Random Travel-Action
5. Others. Randomly chosen from an arbitrary mix of the four previous

3.2.4.4 DR-GGP Hybrid

The directed random - goal graph paths (DR-GGP) hybrid seeks to improve the base goal graph path sub-algorithm by adding the directed random approach as the catching mechanism when GGP fails to find a suitable action.

3.2.4.5 Goal Graph Multi-Paths

The goal graph multi-path algorithm (GGMP) is an improved version of GGP. Instead of selecting a single path from the goal graph, GGMP looks for multiple paths in order to have more suggested actions for the plan graph to try to apply to the current state. Figures 9 to 16 illustrates how this works.

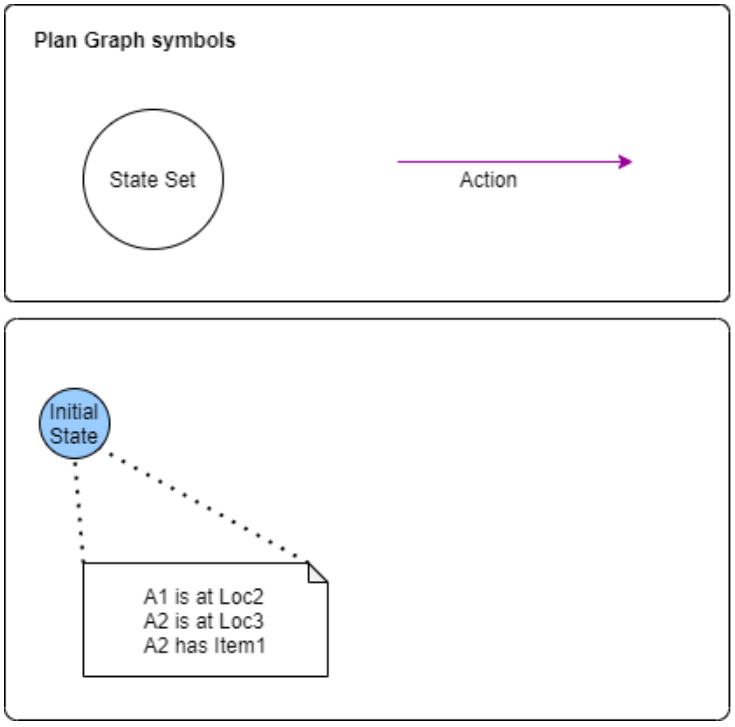

Figure 9: Plan Graph initiation – After the goal graph is generated, the plan graph is started up as a single node which contains the current state, which in the case of our example is the set *{A1 is at Loc2, A2 is at Loc3, A2 has Item1}*

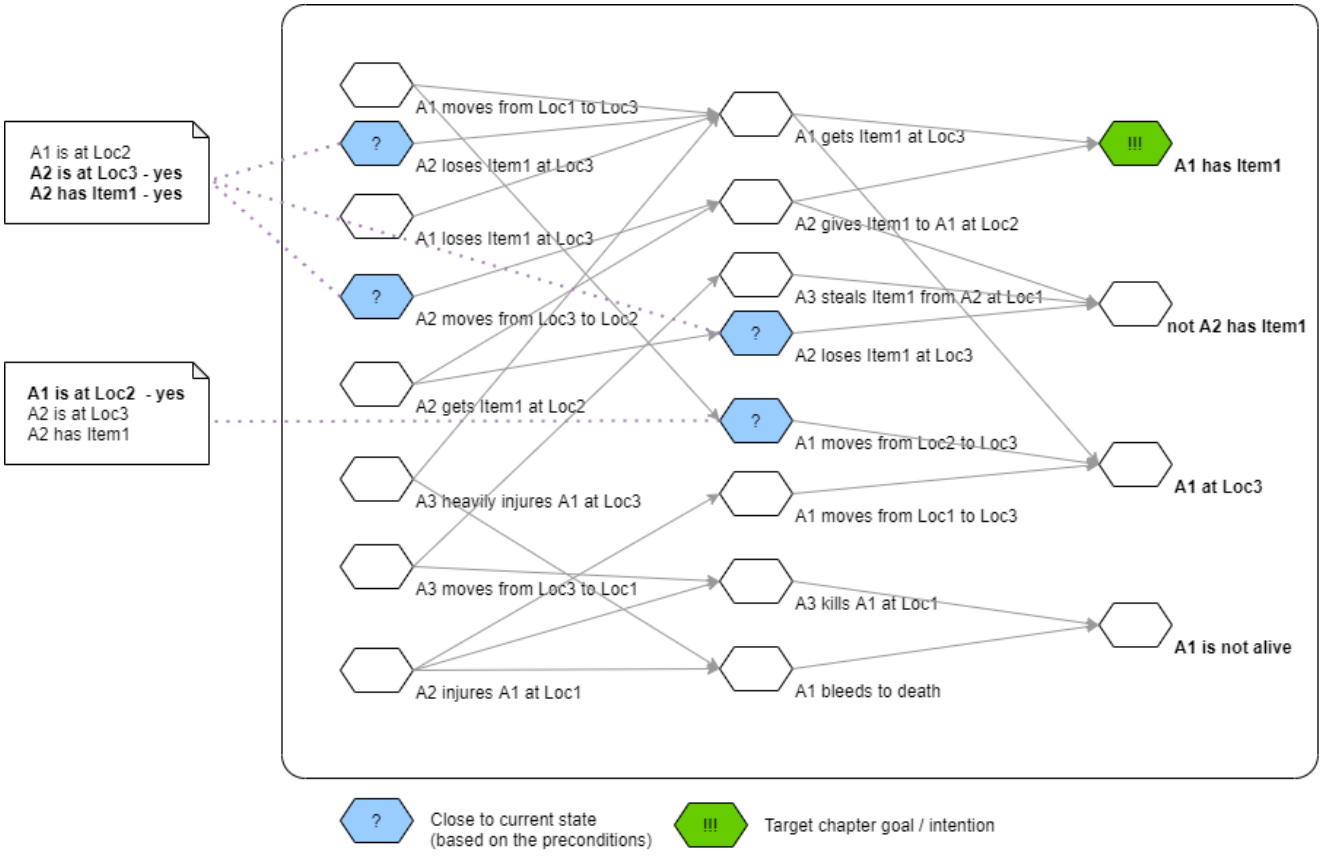

Figure 10: Using the goal graph from figure 5, ISLA identifies which goal graph nodes have an acceptable intersection with the current plan graph node's *state*. These candidate starting nodes (marked as '?') are potential starting points for paths to the relevant goal node (marked as '!!!') which represents the specific goal which ISLA trying to solve for.

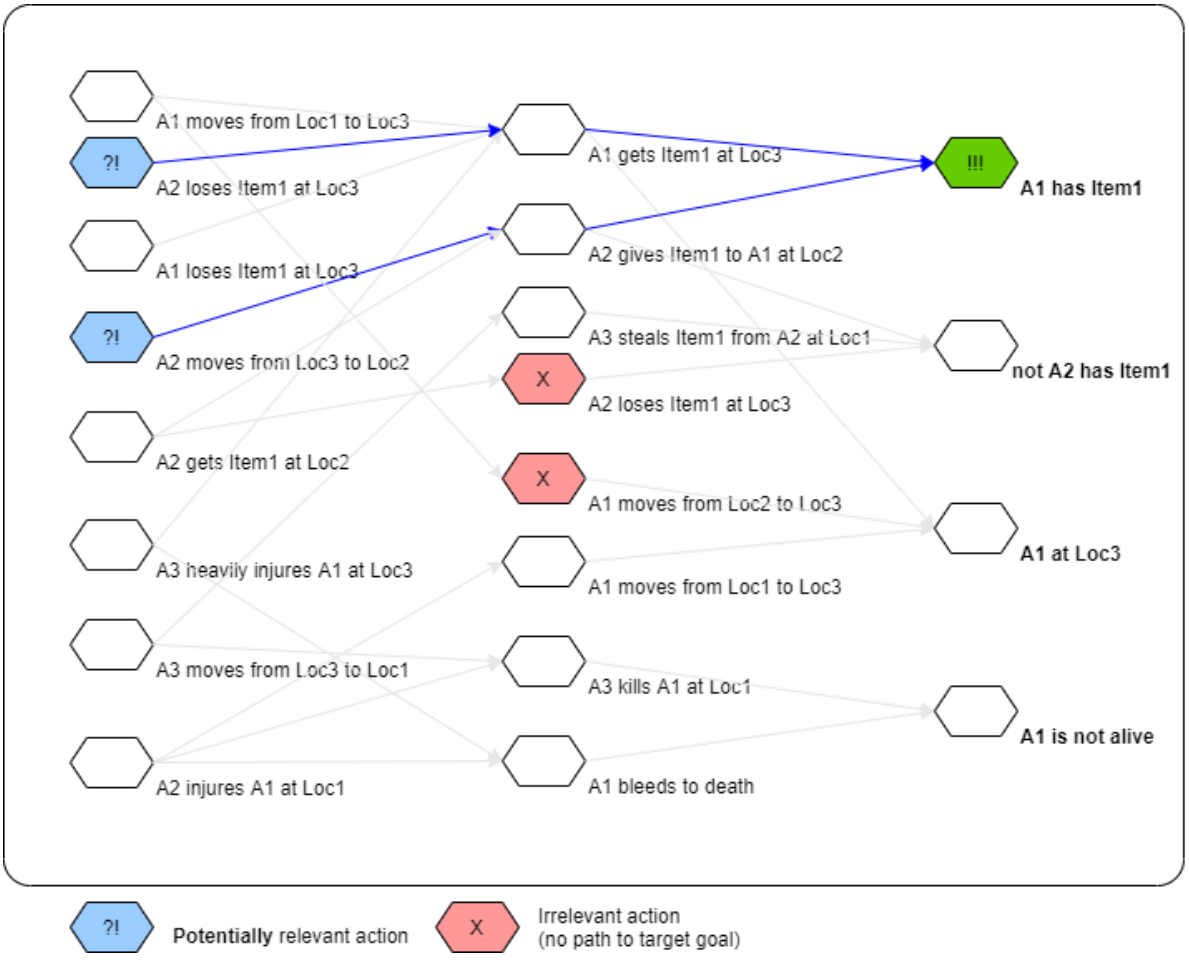

Figure 11: Goal graph nodes with paths to the relevant goal nodes are chosen as the actions to be applied to the current plan graph node (see fig. 12)

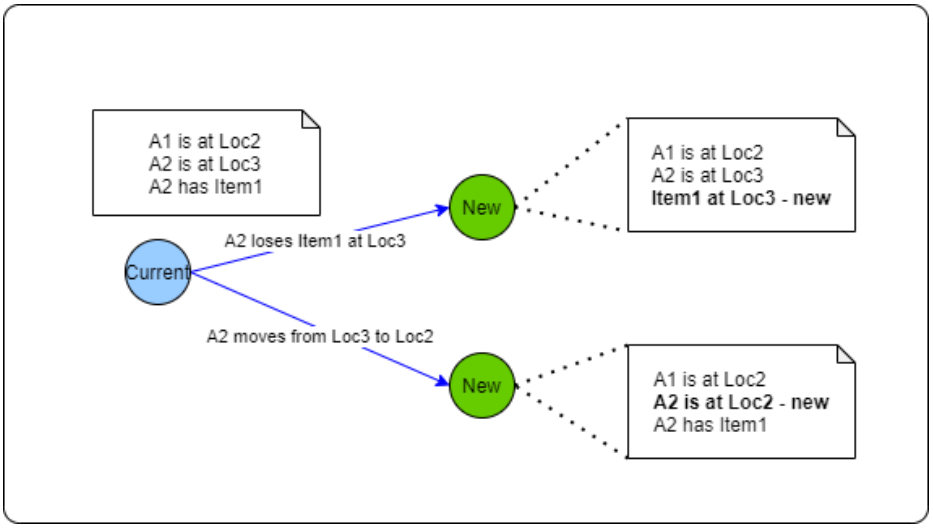

Figure 12: The plan graph grows by applying the actions identified by the goal graph as likely to move the current state closer to one of the goals. In this case, both actions *can* be applied to the current state – their complete preconditions are satisfied – however, this is not always the case (see fig. 14)

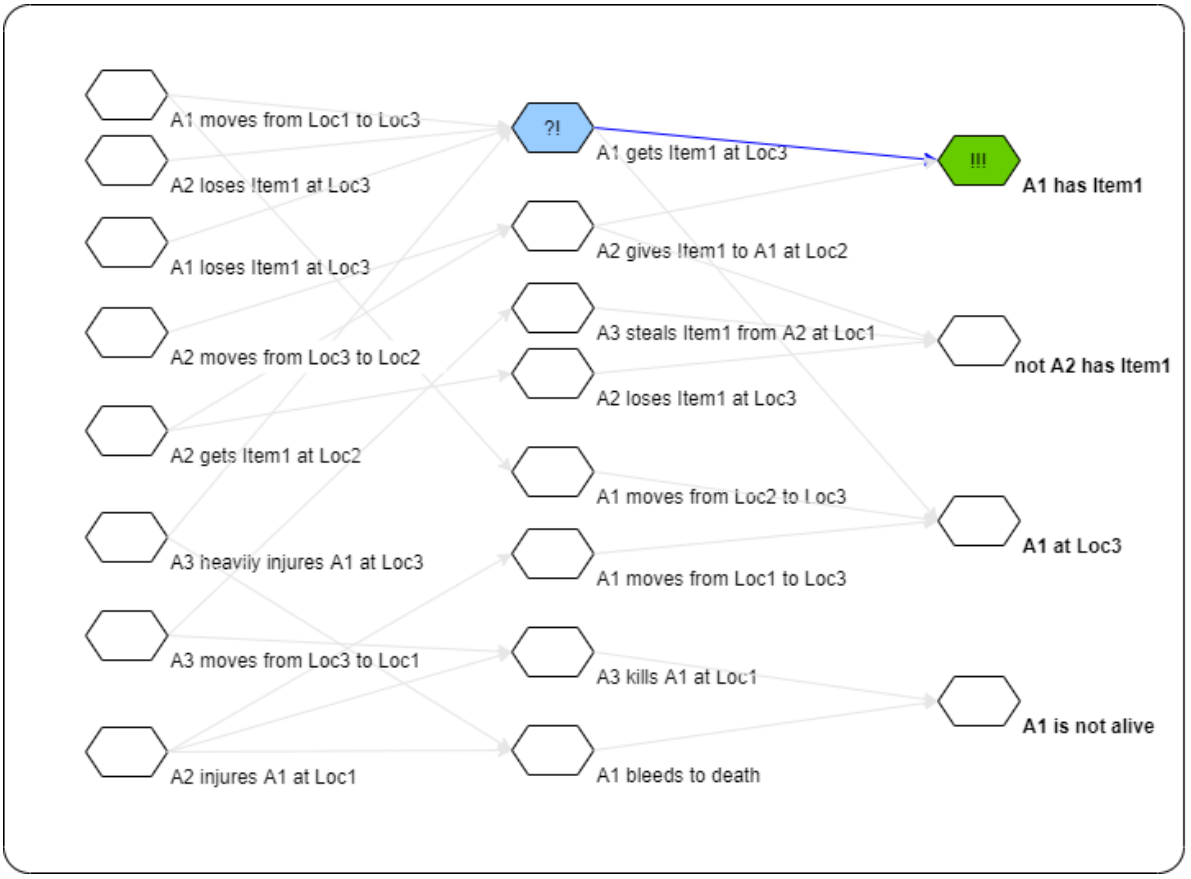

Figure 13: The *goal graph* continues to identify the next action to be tried by the *plan graph*.

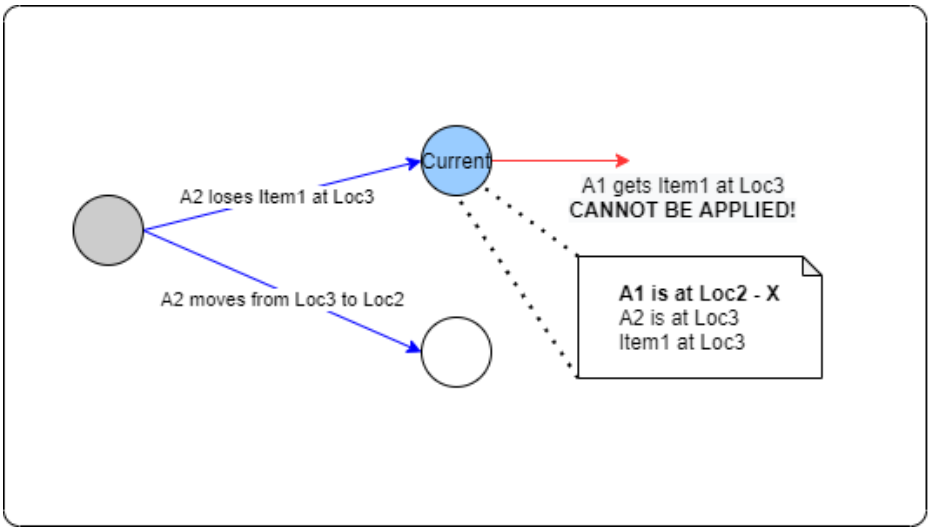

Figure 14: The process of *goal graph*-assisted *plan graph* growth continues, but the action identified step (fig. 13) is not compatible with the current state, and the now-current plan graph node is a dead end for the goal intention ***Author*** *intends A1 has Item1*

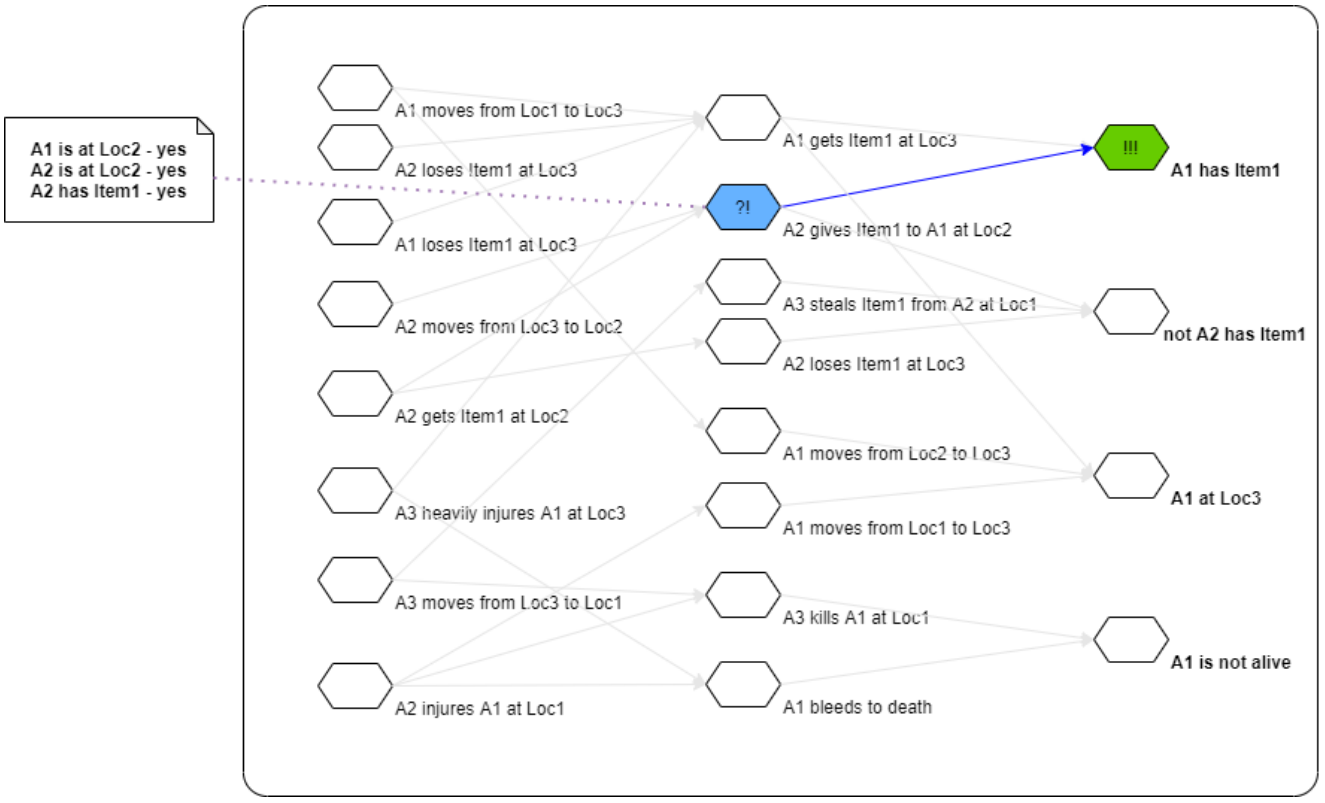

Figure 15: The *goal graph* continues to identify the next action to be tried by the *plan graph*.

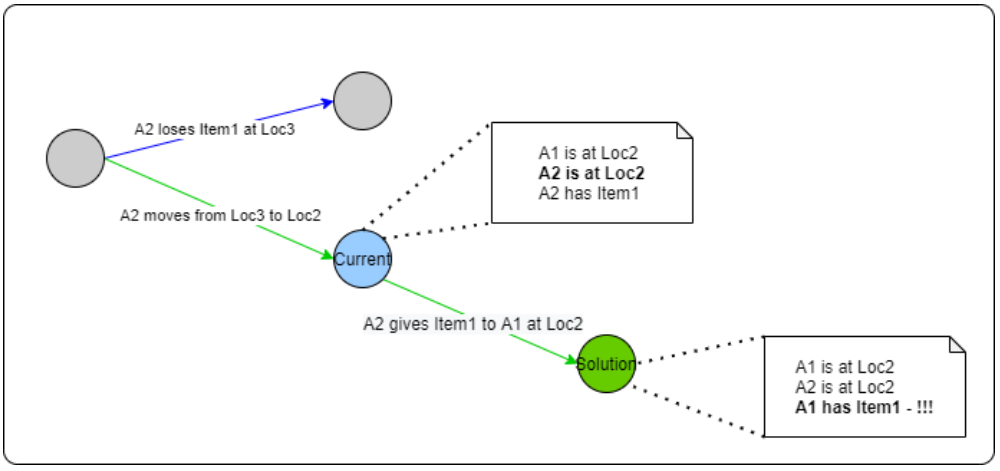

Figure 16: The latest action applied to the now-current plan graph node fulfills the intention *Author A1 has Item1* because the action *A2 gives Item1 to A1 at Loc2* has the relevant effect predicate: *A1 has Item1*.

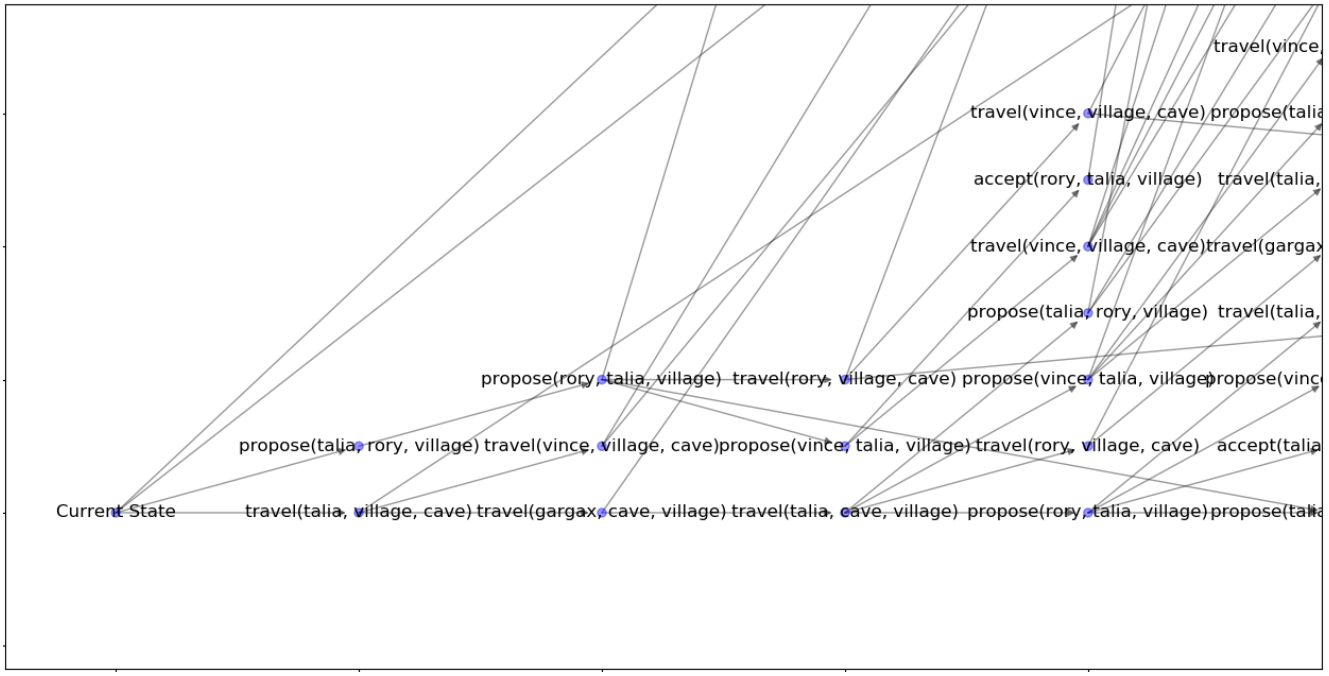

Figure 17: Plan Graph - From the initial state (taken from an earlier version of ISLA)

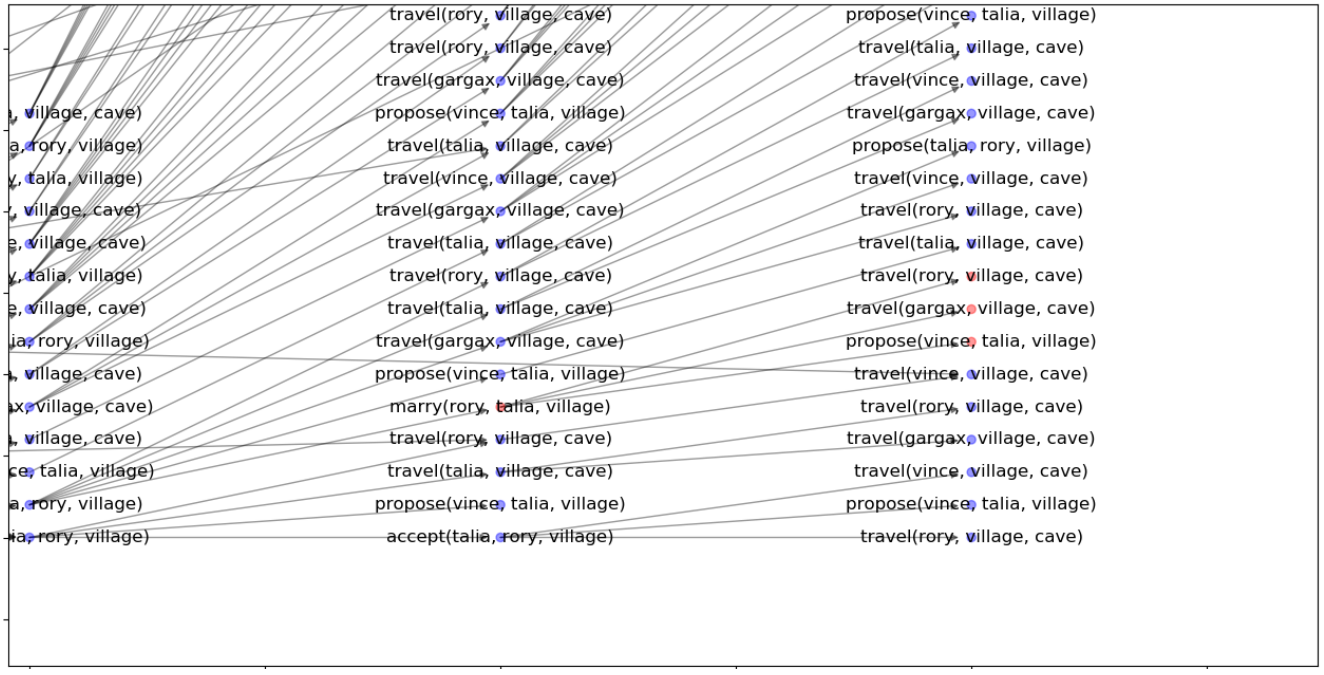

Figure 18: Plan Graph - Red nodes are solutions (taken from an earlier version of ISLA)

3.3 Chapter Chainer

Finding an intentional path for an entire narrative proved to be an intractable problem for ISLA. In order to find the long term author goals, ISLA needs a very large plan graph (with a depth of at least 8 nodes). Related to this issue was the difficulty in explaining all the steps in a candidate solution sequence. Steps such as π_1 and π_4 often have long causality link chains associated with them, which cause plan graphs to become very large. These two issues had a dramatic negative effect on ISLA's ability to find solutions for most trial problems during development.

The solution was, instead of setting lofty character goals (as is GLAIVE's approach) and letting ISLA find solutions by herself, character goals are divided into manageable sub-goals. These smaller goals are then solved and can be better understood if viewed as individual chapters of a larger narrative. Aside from solving the long causality link chain problem, having a *chapter chainer* module also facilitates more structured narratives to be produced.

3.3.1 Definitions

The narrative chapter structures introduced in [sec.3.1.2] are more properly described with the chapter chainer in the context. Each specific sequence term is linked to its successor, much like a causality link. Formally, a *chapter link* is defined as:

$$C_{\mathrm{n}}\langle S_{\mathrm{n}}, G_{\mathrm{n}}\rangle \rightarrow C_{\mathrm{n+1}}\langle S_{\mathrm{n+1}}, G_{\mathrm{n+1}}\rangle$$

where C is a chapter
S is the initial state of C
G is a set of author's goals for state C
S_{n+1} is the final state of chapter C_n (12)

A *chapter chain* is a sequence of chapter links from C_0 to C_{n}; where S_0 is the initial state of chapter 1, G_0 is the author's goal for chapter 1; S_1 is the initial state of chapter 2, G_1 is the author's goal for chapter 2; and so on ... , until C_{n}.

3.3.2 Benefits of the Chapter Chainer

Using a "quest story" as an example, the following sections discusses the benefits of implementing the chapter chainer in ISLA.

The Quest
(1) Difficult Task Arises
(2) ???
...
(n-1) ???
(n) Quest Resolution (13)

During the early stages of ISLA's development, the coreplanner was expected to produce solution plans using only two inputs:

- a set of initial predicates (the initial state)
- a set of target final predicates (author's goals)

This resulted in ISLA trying to navigate very large plan graphs in an effort to produce a solution path. The hypothesis was that if the singular author goal needs to be broken down into a series of easier-to-achieve sub-goals, ISLA can reach the author's goals by solving these sub-goals sequentially. By using known narrative structures (sec. 2.2), we are able to represent these sub-goals as story chapters – essentially a smaller narrative in itself – which ISLA can solve using the coreplanner.

The Quest
(1) Difficult Task Arises
(2) Quest Assigned
(3) Departure 1
(4) Departure 2
(5) Confront Guardian
(6) Quest Resolution (14)

Since the number of chapters is known beforehand, a depth-first-search is employed to look for a suitable sequence of chapters for the narrative. By traversing the depth of a potential chain of chapters, ISLA attempts to locate complete chapter chains first before branching out to different paths.

4. RESULTS AND DISCUSSIONS

4.1 Initial Outputs and Analysis

Given a reasonably crafted domain, an earlier version of ISLA is able to find narrative solutions on most runs. Initial tests involved a domain with around 30+ actions and a comparable amount of state predicate definitions, however, this relatively average domain complexity proved to be too much for ISLA to handle. With this in mind, a much smaller domain (10 action definitions) was crafted specifically to help conduct repeatable tests to quantify ISLA's performance. This smaller domain describes a simple quest story, where characters can delegate intentions, travel, obtain and lose items, and harm other characters. Succeeding sections will discuss in more detail several factors affecting ISLA's efficacy. Findings from these experiments have led to numerous changes to ISLA, and she can now consistently produce story outlines from larger domains, at higher success rates.

4.1.1 Core Planner

Input Domain factors

Initial experimental results are based on a relatively tiny domain – *The Quest*. This domain contains a mere 10 action definitions and 7 state predicate definitions.

Chapter Pattern (The Quest):

HeroDispatched
Departure1 (journey of hero)
Departure2 (continuation of journey)
Guardian (confront the item guardian)
QuestResolution (hero obtains the mcGuffin)

Actions (The Quest):

delegate_getquest
creaturegetsitemfromplace
creaturelosesitematplace
creaturegiveitemtocreature
steal
creaturekillcreature_withweapon
creaturekillcreature_noweapon
creatureinjurescreature_withweapon
creatureinjurescreature_noweapon
creaturebleedsout

(15)

State Predicates (The Quest):

(adjacent (?fromplace - place) (?toplace - place))
(alive (?somecreature - creature))
(at (?object - object) (?place - place))
(has (?somecreature - creature) (?item - item))
(hasweapon (?somecreature - creature))
(isinjured (?somecreature - creature))
(isheavilyinjured (?somecreature - creature))

Output Outline

The following outline is a good example of what ISLA is capable of producing:

Run ID 20190503_114856: Outline

Chapter 1
delegate_getquest(jisala, rin, mysteriousorb, cave)
creaturegetsitemfromplace(rin, mysteriousorb, cave)
Chapter 2
creaturegiveitemtocreature(rin, sjaanat, mysteriousorb, cave)
creatureinjurescreature_noweapon(sjaanat, jisala, cave)
Chapter 3
travel(michael, forest, village)
creatureinjurescreature_noweapon(heather, gargax, village)
creatureinjurescreature_noweapon(sjaanat, jisala, cave)
Chapter 4
travel(rin, cave, forest)
creatureinjurescreature_noweapon(gargax, michael, village)
creaturebleedsout(michael)
travel(rin, forest, cave)
creatureinjurescreature_noweapon(gargax, heather, village)
Chapter 5
creaturekillcreature_noweapon(heather, gargax, village)
creaturelosesitematplace(heather, shield, village)
Chapter 6
creaturegiveitemtocreature(sjaanat, rin, mysteriousorb, cave)
creatureinjurescreature_noweapon(jisala, sjaanat, cave)

(16)

Recall that the goal of *The Quest* is for the hero to obtain a special item type – the McGuffin (15). Run ID 20190503_114856 Outline (figure 16) is particularly interesting because that goal was achieved right away on Chapter 1; the hero, *Rin*, immediately finds the *MysteriousOrb* after being given the quest by another character, *Jisala*. One might expect that ISLA will halt at this point since the author's goal has been met, however the requirement that the story outline match the story pattern **HeroDispatched-Departure1-Departure2-Guardian-QuestResolution** has not been satisfied.

The primary story-line progresses on chapters 1,2,3,5, and 6 like so:

Run ID 20190503_114856: Primary Story-line

Chapter 1: *HeroDispatched*

Rin obtains quest from Jisala

Rin obtains MysteriousOrb

Chapter 2: *pre-QuestResolution*

Rin loses MysteriousOrb to Sja'anat

Chapter 3: *Guardian (injure)*

Heather injures Gargax (the Guardian) in the Village

Chapter 5: *Guardian (kill)*

Heather kills Gargax in the Village

Chapter 6: *QuestResolution*

Sja'anat gives the MysteriousOrb to Rin

(17)

Since ISLA assembles the story using actions from different actors with potentially different intentions, it is observed that some narrative sub-goals *can* be satisfied by actions performed by a different actor from the actor with the goal *intent*. For example, the story sub-goals Guardian (injure)[6] and Guardian (kill)[7] were fulfilled by Heather (a supporting actor) and not the main hero actor, *Rin*. When the narrative reached outline Chapter 4, ISLA was looking to fulfill the intention *intends(rin, isinjured(gargax))*. However, that state predicate is already true, thanks to the action *creatureinjurescreature_noweapon(heather, gargax, village)* applied on outline Chapter 3.

4.2 Overhaul of the Coreplanner, Chapter Chainer, and GUI

4.2.1 Improvements to Coreplanner and Chapter Chainer

During the course of development, several algorithms were implemented for the coreplanner module: with goal graph paths (GP), directed random (DR), and goal graph paths - directed random hybrid (GPDR) during the initial phase, and goal graph multipaths (GMP) developed and implemented during the overhaul phase. The pre-overhaul algorithms seemed to work, but suffered from significant failure rates and high runtimes during executions that are bound to fail (see figure 20). This triggered the re-evaluation of the core codes and eventually led to the creation GMP (expounded on section 3.2.4.5). Comparative tests indicated that even though GGMP has a higher average execution time (fig. 19), the fact that it does not fail has proven that it is the most desirable algorithm that can (currently) be implemented by ISLA.

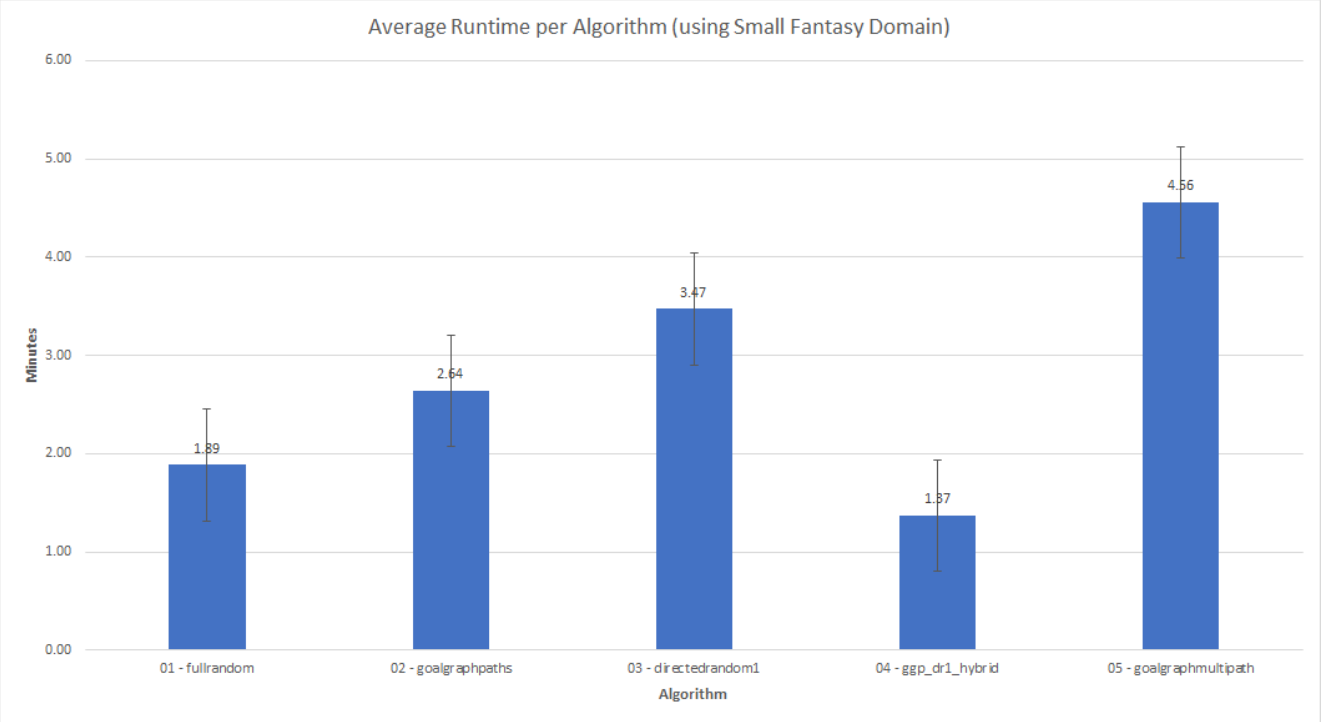

Figure 19: Average Runtime per Algorithm (using Small Fantasy Domain)

[6] Hero (Rin) intends Guardian (Gargax) to be injured

[7] Hero (Rin) intends Guardian (Gargax) to be *not* alive

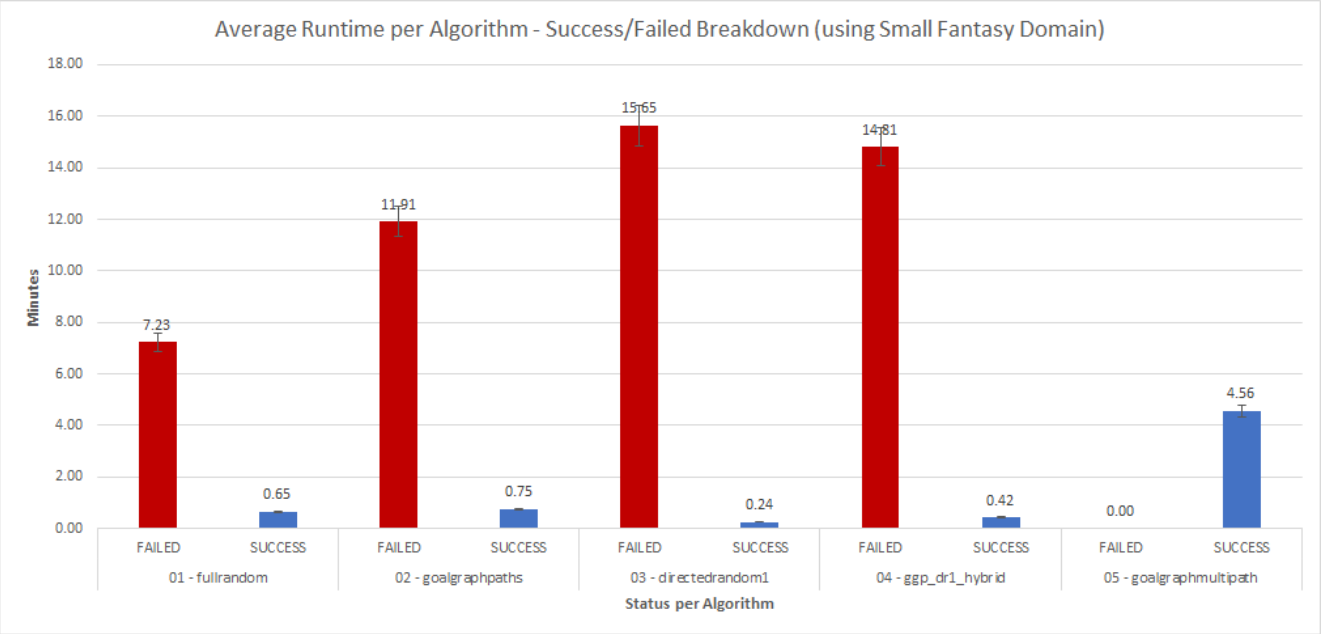

Figure 20: Average Runtime per Algorithm - Success-Failed Breakdown (using Small Fantasy Domain)

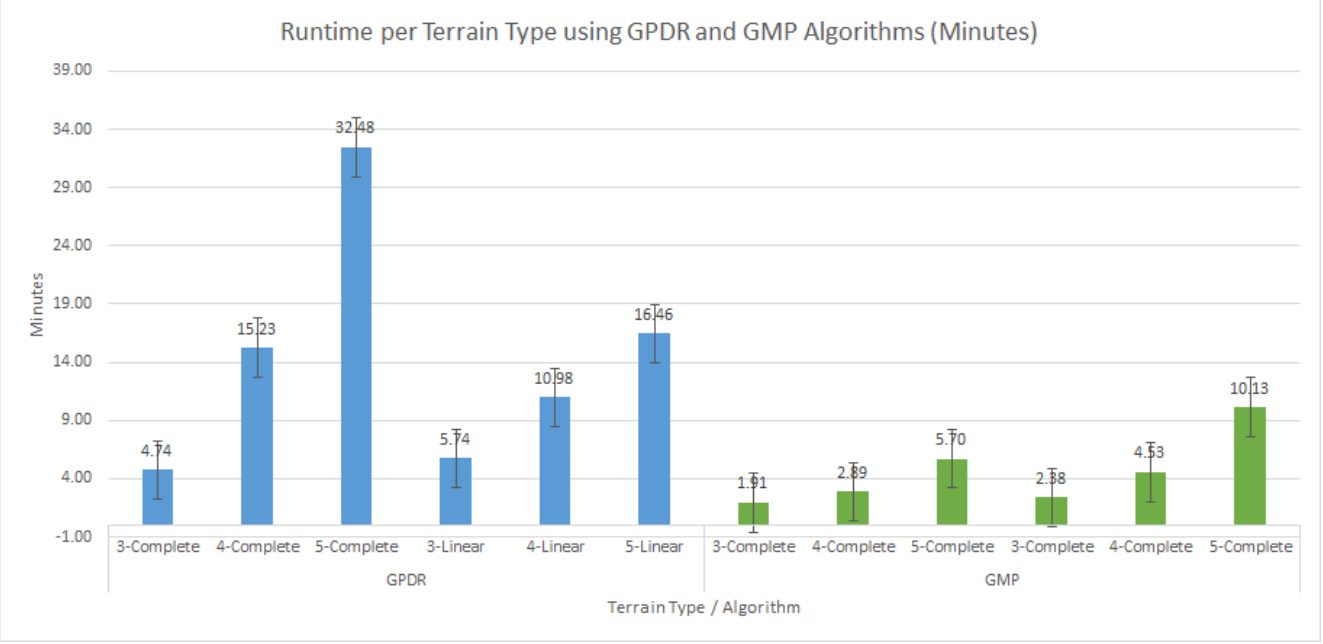

Figure 21: Runtime per Terrain Type using GPDR and GMP Algorithms (Minutes)

4.2.2 Using Tellability as a Measure

Complex plot units as explored by Berov [4] and introduced by Lehnert [14] seems to closely resemble ISLA's "chapter patterns". Since chapter patterns are manually defined in ISLA, it may be argued that ISLA's tellability score is independent of the algorithms used for narrative planning. However, since the chapter patterns approach is an important difference (chapter 3.3) of ISLA from GLAIVE, it may also be argued that ISLA's tellability score is higher that GLAIVE's.

Here is an example output from ISLA:

- Illaoi **obtained** Unknown_Cylinder at Central Command *(Subgoal 1 achieved)*
- Illaoi became focused *(Positive emotion)*
- Illaoi **obtained** DNA_Sample at Central Command (Subgoal 2 achieved)
- Illaoi moves from Central Command to engineering
- Illaoi **drops** DNA_Sample at Engineering *(Subgoal 2 un-achieved)*

When compared to the pattern of nested subgoals and fleeting success in figure 2, it becomes apparent that ISLA's output outlines does indeed contain complex plot points.

4.3 Demonstration to Industry Professionals

In order to gauge ISLA's practical potential, demonstrations to a few industry professionals were held to showcase an earlier version of ISLA. This proceeded starting with a short presentation of ISLA's background and technical details, a live demonstration of ISLA's capability, and finally an open question and answer session in order to gather the attendees' sentiments. The following sections will highlight the results.

4.3.1 Survey Results (Industry Professionals)

Part 1: Average score, with a range between -3 and +3

1. Did you find ISLA easy to use? **0.33 - "Neutral"**
2. Did you find ISLA easy to use? **0.33 - "Neutral"**
3. Did you find ISLA helpful in layouting stories? **1.33 - "Slightly Helpful"**
4. How sensible is this particular generated story? **1.33 - "Slightly Sensible"**

Part 2:Average score, with a range between 0 and 5

1. How helpful is ISLA as a tool for authors / creative writers? **3 - "Useful, but needs more improvements in many areas"**
2. How helpful is ISLA as a tool for game developers / game content creators? **2 - "ISLA has a tiny glimmer of potential"**

4.3.2 Interview Highlights

"[This story] was more interesting for me. But it might also be because I can visualize possible 'reasons' behind the unfolding events more. Could be cause I favor the genre more."

"Lots of potential with what you have so far. Lots of potential functions in different ways."

Adjectives: arbitrary, dramatic, twisty, interesting, amusing, cute, structured, dull, objective, plain

Antonio Gabriel "Tobie" Abad IV

Creative Director, Head Game Designer of Taktyl Studios

—

"It seems harder to setup something than just write an outline then make a story from there. I do usually start with world building first, because the rules will always be established first before you can make stories out of it. I have a bit of difficulty using the tool. But it's great to know that you can control scenarios and not just everything is randomly generated, so I think this tool is best used for game designers that concentrate on narratives. Just needs a bit of tweaking."

Core suggestion: Fix UI

Dr. Beatrice Margarita V. Lapa

Professor at De La Salle-College of Saint Benilde, Senshi.Labs

—

"Great potential! Not for established authors, since they already have their own mental model for creating stories, but seems like a good tool for writers when used in writing exercises. ISLA looks useful for game development purposes."

Inception: Inspired the idea to use ISLA as a teaching tool

Juan Karlo Licudine

Co-Founder and Lead Game Developer Mindcake Games

4.4 ISLA as a Teaching Tool Experiment

We conducted a simple exercises designed to measure ISLA in two aspects: viability of ISLA as a teaching tool, and sensibility and helpfulness of ISLA's narrative outline outputs. The participants are two teachers teaching high school English, with a total of 100 students between them, across 4 sections.

The experiment proceeded as follows.

1. The teachers (playing as the client) and myself (playing as the ISLA platform's roll-out team) conducted initial meetings to gather simple parameters in order to design the class exercise/s that will be deployed later on. Details such as "Which story genres are acceptable to use" and "Does your class require active technical support during the exercises" are agreed upon.
2. The roll-out team pre-generates narrative outlines using ISLA before the class exercises.
3. The roll-out team dispenses the exercise page URLs to the participating teacher and students, and briefs them on how to use ISLA
4. The class proceeds as normal, and the students providing the survey answers near the end of the class. These student surveys measure the sensibility and helpfulness of ISLA's narrative outline outputs
5. The client teacher then answers a separate survey form to evaluate the viability and usability of ISLA as a teaching tool

4.4.1 Survey Results (Students)

4.4.1.1 Score Highlights

The lowest scores that ISLA received are for the usability and GUI aspects. A significant portion of the student participants reported that the information displayed by ISLA seems to be overwhelming, "needs improvement as it was hard to navigate", "requires specific prerequisite knowledge to use". Total score: 0.33 - Neutral.

Majority of the participants have recognized the usefulness of ISLA's output as basis for more fleshed-out storied. Both questions 4 and 5 received decent overall marks with 3.63/5 and 3.85/5 average respectively.

4.4.1.2 Survey Questions

Average score, with a range between -3 and +3
N = 27

1. Did you find ISLA easy to use? **0.33: "Neutral"**
2. Did you find ISLA helpful in layouting stories? **1.30: "Slightly Helpful"**
3. How sensible is this particular generated story? **1.26: "Slightly Sensible"**

Average score, with a range between 0 and 5
N = 27

1. In your opinion, how helpful is ISLA as a tool for authors / creative writers? **3.63 - "Useful, but needs more improvements in many areas"**
2. In your opinion, how helpful is ISLA as a tool for game developers / game content creators? **3.85 - "Useful, but needs more improvements in many areas"**

4.4.1.3 Feedback Highlights

"Although ISLA is not user friendly because it requires specific knowledge about codes, it is still helpful to students like me because it generated a story outline for me who cannot connect my own ideas when writing a story. The action/initial state generated can help the author in producing more ideas for their story yet the resulting state is confusing. Overall, the website is confusing as well as the terms used yet I see some potential."

"Personally, I think I would opt to use ISLA if I had more time to understand its features (for I only get the gist of it). I like how it gives the author a, somewhat, clear summary of the flow of their story and other possible events within the story."

"I think this is perfect for games that follow a storyline. I do not really see the potential of ISLA for professional writers."

4.4.2 Survey Results (Teachers)

ISLA is generally acceptable to the participating teachers, and have acknowledged ISLA's capacity as a teaching tool. Initial comments have described ISLA's impact as "time-saving" and "creates interest in the subject".

5. FUTURE WORK

5.1 Functional Improvements

5.1.1 Problem extrapolation from sparse input

Pattern matching a user's limited input to ISLA's knowledge-base may yield better narrative results, as opposed to selecting from predefined chapter patterns. Context extraction from user input may yield a better experience when used as a stand-alone system or as a supporting component in a bigger ecosystem (e.g. a game).

5.1.2 NLP functions

Extending ISLA's capability may require concepts from natural language processing (NLP) and natural language generation (NLG) domains as detailed in McIntyre's work [16]. Applying such capabilities to ISLA will greatly increase the efficacy at which the output is understood, and will therefore be more useful to a wider range of audiences.

5.1.3 Machine Learning

Since it is possible to define both initial state, domain, and final state, it is interesting to employ supervised machine learning to help fine-tune internal algorithm junctions such as narrative structure refinement.

5.1.4 Other Algorithmic Improvements

Implementation of character beliefs by applying and utilizing epistemic edges on the plan graph, initial state refinement, state consistency mechanics, and individual functional actor personalities may be looked into in order to improve the overall experience and performance of ISLA in the future.

5.2 Non-functional Improvements

5.2.1 Performance

A key hindrance in ISLA's further development is the extreme runtimes she experiences when performing planning operations. Several factors are being looked into to address this: parallel processing, programming language (compiled vs. interpreted), core code re-engineering and optimization.

5.2.2 Others

Output quality, extensibility, and usability are all aimed to be primary design concerns that needs to be considered also on succeeding incarnations of ISLA. Improving upon the GUI to be more user-friendly requires the assistance of knowledge from the UI/UX field, especially with issues such as conveying fundamentally technical information to a non-technical audience.

6. CONCLUSIONS

A forward-chaining narrative planner, which supports intentionality, is constructed. This narrative planner, called the Intelligent Story Layout Assistant (ISLA), is able to construct story layouts that achieve the author's goals while making sure that *most* steps in the plan have clear motivations. These layouts, or solution plans, are based on a handcrafted knowledge-base of story universe elements. The current output is capable of displaying the underlying motivations of each action, and partial/complete state transition information.

The knowledge-base can be improved by manually encoding more domains, and by refining knowledge-bases elements by employing machine learning techniques. Performance issues are obvious concerns. Compatibility with external systems will require further re-engineering and research.

In our analysis, ISLA's output proved sufficiently sensible on most cases, but has significant challenges particularly with performance, user-friendliness. ISLA's usefulness to professional writers have colloquially low score; on the other hand, ISLA is surprisingly useful in the realm of teaching as a tool to teach students how to write creative stories of their own. ISLA's narrative outline outputs performed decently on the student surveys even in the presence of the aforementioned difficulty is presenting fundamentally technical aspects of the narrative to a predominantly non-technical audience. ISLA currently has the *underlying* data structures needed to further assist the author in fleshing out the produced story layout. These have been exposed to some extent and has been recognized as helpful information for content authors.

References

[1] AGUIRRE, M. An Outline of Propp's Model for the Study of Fairytales. *The Northranger Library Project* (2011).

[2] BAHAMÓN, J. C., BAROT, C., AND YOUNG, R. M. A Goal-based Model of Personality for Planning-based Narrative Generation. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (2015), AAAI'15, pp. 4142–4143.

[3] BAL, M. *Narratology: Introduction to the Theory of Narrative, Second Edition*. University of Toronto Press Incorporated, 1999.

[4] BEROV, L. Towards a Computational Measure of Plot Tellability. In *10th International Workshop on Intelligent Narrative Technologies* (10 2017), pp. 169–175.

[5] BOSSER, A.-G., COURTIEU, P., FOREST, J., AND CAVAZZA, M. Structural Analysis of Narratives with the Coq Proof Assistant. In *Interactive Theorem Proving - Second International Conference, ITP 2011, Berg en Dal, The Netherlands* (08 2011), pp. 55–70.

[6] COLES, A. I. Heuristics and Metaheuristics in Forward-Chaining Planning. *Ph.D. Dissertation, University Strathclyde* (2007).

[7] DIETIKER, L. Mathematical texts as narrative: Rethinking curriculum. *For the Learning of Mathematics 33*, 3 (2013), 14–19.

[8] GERVÁS, P., LEÓN, C., AND MÉNDEZ, G. Schemas for Narrative Generation Mined from Existing Descriptions of Plot. In *Proceedings of Computational Models of Narrative* (May 2015).

[9] GOGUEN, J. Social and Semiotic Analyses for Theorem Prover User Interface Design. *Formal Aspects of Computing 11* (1999), 11–272.

[10] HOFFMANN, J., AND NEBEL, B. The FF Planning System: Fast Plan Generation Through Heuristic Search. *Journal of Artificial Intelligence Research 14* (May 2001), 253–302.

[11] JORDANOUS, A. A Standardised Procedure for Evaluating Creative Systems: Computational Creativity Evaluation Based on What it is to be Creative. *Cognitive Computation 4*, 3 (2012), 246–279.

[12] KAMAREDDINE, F., MAAREK, M., RETEL, K., AND WELLS, J. B. Narrative Structure of Mathematical Texts. In *Towards Mechanized Mathematical Assistants* (Berlin, Heidelberg, 2007), M. Kauers, M. Kerber, R. Miner, and W. Windsteiger, Eds., Springer Berlin Heidelberg, pp. 296–312.

[13] KYBARTAS, B., AND BIDARRA, R. A Survey on Story Generation Techniques for Authoring Computational Narratives. *IEEE Transactions on Computational Intelligence and AI in Games 9*, 3 (Sept. 2017), 239–253.

[14] LEHNERT, W. G. Plot Units and Narrative Summarization. *Cognitive Science 5*, 4 (1981), 293–331.

[15] MANI, I. Computational Narratology. https://www.lhn.uni-hamburg.de/node/43.html, January 2013.

[16] MCINTYRE, N. Learning to Tell Tales: Automatic Story Generation from Corpora. *Ph.D. Thesis, University of Edinburgh* (2011).

[17] MEEHAN, J. R. TALE-SPIN, an Interactive Program That Writes Stories. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence - Volume 1* (San Francisco, CA, USA, 1977), IJCAI'77, Morgan Kaufmann Publishers Inc., pp. 91–98.

[18] MILLER, B., AND PARK, J. S. Computing Narrative. In *Proceedings of the Workshop on Computational Humanities Research (CHR 2020), Amsterdam, The Netherlands, November 18-20, 2020* (2020), vol. 2723 of *CEUR Workshop Proceedings*, pp. 182–190.

[19] RIEDL, M. O., AND YOUNG, R. M. Narrative Planning: Balancing Plot and Character. *J. Artif. Int. Res. 39*, 1 (Sept. 2009), 217–268.

[20] SANGHRAJKA, R., HIDALGO, D., CHEN, P., AND KAPADIA, M. LISA: Lexically Intelligent Story Assistant. In *Proceedings of the 13th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2017* (2017), pp. 221–227.

[21] SANGHRAJKA, R., WITON, W., SCHRIBER, S., GROSS, M., AND KAPADIA, M. Computer-Assisted Authoring for Natural Language Story Scripts. In *30th Conference on Innovative Applications of Artificial Intelligence (IAAI-18)* (2018).

[22] SHIRVANI, A., WARE, S. G., AND FARRELL, R. A Possible Worlds Model of Belief for State-Space Narrative Planning. In *Proceedings of the Thirteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-17), October 5-9, 2017, Snowbird, Little Cottonwood Canyon, Utah, USA.* (2017), pp. 101–107.

[23] SIMONSON, D. E. Investigations of the Properties of Narrative Schemas. *Georgetown University - Graduate School of Arts and Sciences* (2018).

[24] TOBIAS, R. B. *20 Master Plots: And How to Build Them*. F+W Media, 2012.

[25] WARE, S. G., AND YOUNG, R. M. CPOCL: A Narrative Planner Supporting Conflict. In *Proceedings of the 7th AAAI International Conference on Artificial Intelligence and Interactive Digital Entertainment* (2011), pp. 97–102.

[26] WARE, S. G., AND YOUNG, R. M. Glaive: A State-space Narrative Planner Supporting Intentionality and Conflict. In *Proceedings of the Tenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (2014), AIIDE'14, AAAI Press, pp. 80–86.

[27] WEINBERG, A., WIESNER, E., AND FUKAWA-CONNELLY, T. The Narrative Structure of Mathematics Lectures. In *North American Chapter of the International Group for the Psychology of Mathematics Education, 2015* (2015), pp. 1306–1313.

[28] YOUNG, R. M. Notes on the use of Plan Structures in the Creation of Interactive Plot. In *M. Mateas and P. Sengers (Eds.), Narrative Intelligence: Papers from the AAAI Fall Symposium* (1999), pp. 164–167.