公立はこだて未来大学

2025年度

システム情報科学実習

グループ報告書

Future University Hakodate 2025 Systems Information Science Practice Group Report

プロジェクト番号 / Project No.

4

プロジェクト名

クリエイティブAI

Project Name

Creative AI

グループリーダー / group Leader

1023147 田本 亜樹斗 / Akito Tamoto

グループメンバー / Group Member

1023022 伊藤 颯 / Hayate Ito

1023061 景山 涼介 / Ryosuke Kageyama

1023105 笹木 颯太 / Sota Sasaki

1023209 麥谷 悠悟 / Yugo Mugitani

指導教員

村井 源　中田 隆行　吉田 博則

Advisor

Hajime Murai　Takayuki Nakata　Hironori Yoshida

提出日

2026年1月21日

Date of Submission

Jan. 21, 2026

概要

クリエイティブAIでは，人間が持つ創造性をサポートするAIシステムの実現と「面白さ」や「美しさ」の科学的な理解の促進を目的としている．この目的を達成するために，ゲームを制作した．ゲームを選択した理由は，人工知能を活用する場面が多様だからである．例として，物語の生成やBGMの生成などが挙げられる．本プロジェクトでは「システム」，「シナリオ」，「視覚」，「音響」の四つのグループに分かれて活動した．

システム班はゲームの根幹となるシステムの開発と各般の成果物の結合をする「システム開発班」とAIを活用したシステムの開発をする「スキルツリー自動生成班」の2つのグループに分かれて活動した．本システム班の人工知能を活用したシステムとしてスキルツリー自動生成システムの開発に着手した．現在は，体験版としてプレイ可能な状態にあるが，システムの改善やスキルツリーの精度向上，デバッグを行うことで完成度の高いゲームの完成を秋葉原の課外発表会までを目標としている．

キーワード：人工知能，プログラミング，自動生成，ロールプレイングゲーム（RPG）

（※文責：田本亜樹斗）

Abstract

The Creative AI project aims to realize AI systems that support human creativity and to promote a scientific understanding of concepts such as "entertainment" and "aesthetic beauty." To achieve these goals, we developed a video game, as it offers a diverse range of opportunities for AI integration, such as narrative generation and dynamic BGM composition.

The project was divided into four specialized groups: System, Scenario, Visual, and Sound. The System group was further subdivided into two teams: the "System Development Team," responsible for the core game mechanics and the integration of deliverables from all groups, and the "Automated Skill Tree Generation Team," focused on AI-driven system development. Currently, a trial version is playable. Moving forward, our goal is to enhance the system, improve the accuracy of the skill tree generation, and conduct thorough debugging to present a high-quality final product at the extracurricular exhibition in Akihabara.

Keywords: artificial intelligence, programming, automatic generation, Role Playing Game(RPG)

（* responsibility for wording of article : Akito Tamoto）

目次

1．はじめに

1．1．背景・研究動機

本プロジェクト全体の目標である「AIによって人間の創造性を実現すること」を達成するために，システム班では成果物となる2Dロールプレイングゲームのシステム開発を行った．あわせて，ゲーム内要素としてスキルツリー自動生成システムの設計および実装を行った．

スキルツリーとは，スタート地点から，ゲーム内で得られるポイントを消費して，スゴロクのようにマスを進めながらスキルを獲得していく育成システムである．獲得するスキルの種類や順番をプレイヤーが自由に判断することができ，ゲームの戦略性の向上に繋がる．また，全てのマスが埋まった時の達成感を味わうこともできる．スキルツリーは，ゲームに新たな楽しみを加えることができる育成システムであり，様々なゲームで使用されている．

従来のゲーム作品において，多くのスキルツリーは制作者が手作業により作成および調整を行っており，いずれの作品においても緻密なレベルデザインが施されている．一方で，その設計および調整には多大な時間と労力を要するという課題が存在する．この課題に着目し，スキルツリー構築プロセスの自動化に価値を見出したことから，本システムを考案した．

また，本システムによって生成されるスキルツリーの構造は一定化されたものではなく，ゲーム開始時ごとに，機能性を損なわない範囲でランダムに生成される．この仕組みにより，限られたコストの中でスキルの取捨選択を行う戦略性や，プレイヤー自身の選択に基づいて成長方針を決定するというスキルツリー本来のゲーム体験を，繰り返し提供することが可能となる．

（※文責：伊藤颯）

1．2．関連研究・関連科目

本プロジェクトでは，以下の科目との関連がある．

- 情報表現入門：Processingの使用方法やコードの書き方の知識を活かし，スキルツリー自動生成システムのプロトタイプを作成．
- 情報処理演習Ⅰ：コードの書き方の知識を生かし，プログラミングやコードの可読性を向上．
- 確率・統計学：状態遷移，重回帰分析などの知識をもとに既存作品のデータ分析を実行．

- オペレーションズリサーチ：状態遷移，重回帰分析などの知識をもとに既存作品のデータ分析を実行．

（※文責：伊藤颯）

1．3．プロジェクト学習の目標

システム班は，他班で作成された人工知能を用いた制作物をロールプレイングゲームとして統合し，誰でもゲームを体験できる状態にすることが目的である．また，AIを活用したスキルツリー自動生成システムを作成することも目的とした．これらの課題のためにシステム班は「システム開発班」と「スキルツリー自動生成班」に分かれた．「システム開発班」はゲームエンジンのUnityを使用してゲームの基盤となるシステムの作成や他班の制作物の統合をし，「スキルツリー自動生成班」はスキルツリーの分析やプロトタイプの作成を行った．

（※文責：景山涼介）

2． 目的達成への手法，手段

2．1． システム開発班

システム開発班は，ゲームの完成を目的とし，開発初期に必要な機能を洗い出した．その結果，「メニュー」，「マップ」，「戦闘」，「会話」の四つが優先度が高いという結論に至った．この四つが優先だった理由として，これらの機能はゲームとしての最小限の構成要素であるからだ．開発体制としては，各メンバーが1つの機能を担当する「責任担当制」を採用した．一人一つの機能を担当することにし，担当箇所を早期に完遂したメンバーが「会話」を担当することにした．四つの機能が完成した後は，優先度の高い機能から順番に作業していくことにした．

プロジェクトを計画的に進行させるため，金曜日に翌週までの個別の目標を設定し，水曜日に進捗報告を行う体制を整えた．このサイクルを繰り返すことで，チーム全体の集中力を維持させるとともにメンバー間の進捗の乖離を早期に発見し，相互にフォローしあえる環境となった．

デバッグは，開発の遅延により当初の予定を圧迫するという課題が生じた．本作は1プレイあたりの所要時間が長く，個別の検証では網羅的な確認が困難であった．この状況を打破するため，特定の担当者に限定せず，メンバー全員で一斉にデバッグを行う「総力戦体制」へと移行

した．これにより，多角的な視点でのバグ発見が可能となり，短期間での品質向上と効率化を実現した．

（※文責：田本亜樹斗）

2．2． スキルツリー自動生成班

スキルツリー自動生成班では，ゲームを開始するたびに新たなスキルツリーを生成するシステムを制作するにあたり，ゲーム内におけるスキルツリーシステムの実装と既存作品のデータ分析の二つに役割を分担し，作業を行った．

また，データ分析を行う作品は売上および知名度の高い作品であるとともに，構造が再帰的なものではなく，十分な大きさのスキルツリーを有するものを中心に選定した．スキルツリーに類似するシステムが無い作品については，主にスキルの分析を目的として，詳細な数値が載った攻略サイトが存在する作品を選出した．分析対象とした作品の一覧を表1に示す．

表1　分析対象とした作品

対象作品	出典	データを使用した工程
ドラゴンクエストXI　過ぎ去りし時を求めて	スクウェア・エニックス 2017	外形の生成 各ノードの報酬指定 スキルの評価および配置
ドラゴンクエストX　オフライン	スクウェア・エニックス 2022	各ノードの報酬指定 スキルの評価および配置
ドラゴンクエストIX　星空の守り人	スクウェア・エニックス 2010	スキルの評価および配置
ドラゴンクエストVIII　空と海と大地と呪われし姫君	スクウェア・エニックス 2004	スキルの評価および配置
ドラゴンクエストIII　そして伝説へ…	スクウェア・エニックス 2024	スキルの評価および配置
FINAL FANTASY VII REMAKE	スクウェア・エニックス 2024	外形の生成 各ノードの報酬指定
FANTASIAN Neo Dimension	Mistwalker（開発） スクウェア・エニックス（発売）2023	外形の生成 各ノードの報酬指定

（※文責：伊藤颯）

2．3． Unity，Gitの学習

本システム班のメンバーの多くは，UnityおよびGitを用いた開発経験がほとんどなかった．そのため，開発初期段階ではツールの操作や開発フローに対する理解不足が課題となった．そこで，昨年度のクリエイティブAIのシステム班の先輩を招き講習を実施した．しかし，Gitの勉強

と環境構築に想定以上の時間を要したため，Unityの勉強に時間を当てることができなかった．そこで，Unityについては「実践を通じて習得する」方針へと転換した．

Gitの学習においては，先輩から提供された資料とともに，環境構築の手順，ブランチの作成・統合の手順，注意点などを学んだ．環境構築は，システム班とシナリオ班の一部のメンバーのみで行った．ほかのメンバーは，デバッグをするときにその都度追加していくことにした．また，共同開発におけるコンフリクトやデータ破損等の事故を防止するため，以下の運用ルールを策定した．

- ブランチ運用：ブランチ名は「feat/○○」の形式で命名．なるべく機能名で記述する．
- 統合管理：プルリクエストの管理を一人に集約し，整合性を維持．
- 大容量データの除外：フォントデータは，GitHub上ではなくGoogle Driveで共有．

これらのルールを徹底することで，複数人による並行開発の混乱を抑え，安全性と効率性を高めた．

（※文責：田本亜樹斗）

3．各機能の説明

3．1．マップ

3．1．1． 概要

本報告では，前期のマップシステムの活動について説明する．マップシステムはロールプレイングゲームの根幹に関わるシステムの一つである．マップ制作やキャラクター移動の開発を担当している．マップは森や村や洞窟などの個々のマップと全体マップというのが存在する．個々のマップから出ると全体マップに遷移し，行きたいマップを選択することで遷移することができる．マップ上はWASDキーで移動するようにした．

マップのイメージについては視覚班とシナリオ班と話し合い，意見の齟齬が生まれないようにした．マップチップはフリー素材と視覚班が作成した素材があり，そこからシナリオ班と逐一話し合いながら作成した．

個々のマップ（左が森，右が海）

全体マップ

（※文責：田本亜樹斗）

3．1．2． 実装

開発初期はタイルマップを使用する方向で開発を進め，キャラクターがマップ上を移動できるところまで実装を行った．しかし，開発の途中でデバッグ作業が非常に困難であるという課題が浮上した．

タイルマップ方式では，マップを構成する各タイルに対して個別に画像や当たり判定，イベント処理を設定する必要がある．そのため，設定ミスや意図しない挙動が発生した際には，マップ上の該当タイルを特定し，目視と手動操作によって確認・修正を行う必要がある．この

ような手法は，小規模なマップであれば有効であるが，今年はマップの規模が拡大したり，イベントが複雑化したりするので，デバッグ作業に多大な労力を要することが明らかとなった．一方で，スクリプトエンジン方式では，マップの構造やイベント処理をスクリプトファイル上に記述する形式で管理できるため，データ構造が明示され，状態の確認や修正が容易である．特定のイベントやマップデータをスクリプト上で検索・抽出できるほか，ログ出力を通じて条件分岐や動作の検証を行うことが可能であり，再現性のあるデバッグが実現できる．以上の理由から効率的な開発を重視して，スクリプトエンジンを使用する方式へと方針を変更した．

スクリプトエンジンは昨年度のプロジェクトでも使用されていたため，今年度はそのコードをベースとして開発を進めた．

UnityのTi形式のデータは左上を原点として記述されるのが一般的である．本システムでは，「GetMapSize().y - 1 - position.y」という計算式を導入することで，配列のインデックスとUnity上の座標系を自動で反転・同期させている．これにより，直感的なマップ配置と正確な描画の両立を実現した．

（※文責：田本亜樹斗）

3．1．3． マップ制作

マップのイメージ共有を視覚班とシナリオ班とともに行った．まず，シナリオ班のストーリーに合ったマップ制作が重要だと考え，シナリオ班のマップのイメージを全員で共有・理解を行った．そこから意見のすり合わせを行い，齟齬が生まれないようにした．マップは，プレイヤーが飽きたり，ストレスがかからないようにするために密度を高めるという判断にした．

しかし，シナリオ班が思い描くマップをうまく実現できず，進行がゆっくりとしていた．作成するマップが多く，一人では制作が困難と判断し，マップの話し合いで参加した視覚班とシナリオ班のメンバーとともに制作した．この時，マップチップを各々がUnityに入れたことにより，マップチップが複数存在してしまっていた．また，Gitで結合した際にマップチップのデータやTilemapsの紐づけが一部欠落している問題が発生していた．現在はは正常にゲームが動作しているため修正は行っていない．

（※文責：田本亜樹斗）

3．1．4． キャラクターの移動

キャラクター移動は，去年のコードを用いている．追加要素として，プレイヤーの移動に合わせて，常にキャラクターを画面中央に捉える追従システムを実装した．本作のマップは有限なサイズで構成されている．そのため，プレイヤーがマップの端に到達した際に，マップ外の何もない領域が表示されないようにする制限ロジックを実装した．これにより，マップ外に余分な装飾を描く必要がなくなり，各マップの有効範囲内での制作に集中することが可能となった．

（※文責：田本亜樹斗）

3．1．5． 全体マップ

3．1．5．1． 概要と役割

全体マップは，森や海など独立した各マップをつなげる「ハブ」の役割を担っている．各マップで外に出ると全体マップが表示され，行きたいマップをWASDで選択することでマップ間の移動を可能にしている．

（※文責：田本亜樹斗）

3．1．5．2． 経緯

最初は，全体マップでも移動が可能で，行きたいマップまで移動してマップのアイコンに触れることでマップに遷移する方向で考えていたが，デバックの段階で労力がかかるということで今のような選択して遷移する方向に変わった．これにより，プレイヤーの移動ストレスを軽減し，デバッグ作業の効率化とゲームテンポの向上を両立させることができた．

（※文責：田本亜樹斗）

3．1．5．3． データ管理

マップの各地点の情報は，UnityのScriptableObjectを用いた「WorldMapData」で一括管理している．これにより，以下のようにプログラムを書き換えることなくインスペクター上から各マップの設定を行うことができる．以下の図は実際のinspecterの画面である．

- Point Name：マップの名称
- Scene Name：遷移先のシーン名

- Area ID：解放条件となる識別子でここに書いたフラグが立つと解放される
- Position：全体マップ上でのアイコンの配置座標
- Unlocked Default：初期状態で解放されてるかの設定（デバックの際に使用）
- Neighor Indices：各マップの経路設定
- icon：全体マップ上でのアイコンの画像設定
- Spawn Position：マップに遷移した時のプレイヤーの出現地点

（※文責：田本亜樹斗）

3．1．6． 工夫

選択中のアイコンを強調するアニメーションを実装したことにより，プレイヤーが現在どこを選択しているのか一目でわかるようになっている．

（※文責：田本亜樹斗）

3．2．メニュー

3．2．1． 概要

マップを移動している最中に，キャラクターの状態や，現在所持しているアイテムを確認する役割がある．ただし，会話中や戦闘中などのイベントが発生している時は，メニューを開く

ことができないようにした．メニューを開くキーを押すことで「メニュートップ画面」に遷移し，そこから以下の四つの画面へ遷移する．

- キャラクター詳細画面
- スキル一覧画面
- アイテム一覧画面
- スキルツリー画面

加えて，「スキル一覧画面」と「アイテム一覧画面」からは，「スキル/アイテム使用画面」へ遷移することができる．なお，各画面における詳しい説明は別の項目に記述した．

背景画像やUIの配置については，視覚班の方と相談を重ねた．前期では，おおよそのUI配置について共有し，後期では，視覚班から送られてきた素材について見やすさなどを考慮して改善を重ねた．

（※文責：麥谷悠悟）

3．2．2． メニュー専用機能

メニューの各画面において，共通して使用できる機能を実装した．

（※文責：麥谷悠悟）

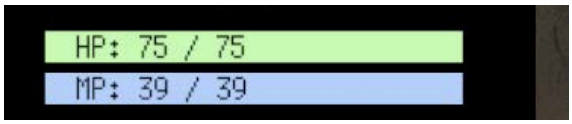
3．2．2．1． タブ機能

「アイテム一覧画面」以外の画面において，一つの画面にパーティメンバー全員の情報を表示すると，スペースが足りなかったり，情報が見えづらかったりなどの問題点が発生する．それを回避するためにも，一つの画面に一人のパーティメンバーの情報を表示する設計を行った．その際，左右キーを押すことで，表示するパーティメンバーを切り替える機能（以後，タブ機能と呼ぶ）を実装した．

（※文責：麥谷悠悟）

3．2．2．2． HP/MPバー機能

RPGを製作したため，キャラクターには体力を表す「HP」と特別な技（スキル）を使用するために必要なポイントを表す「MP」が存在する．文字のみを使用することで，現在のHPとMPの値を記述しても良かったが，どのくらい値が減っているのかが分かりにくい．そこで，視覚的に分かりやすくするために，「HPバー」と「MPバー」を製作した．



即興で製作したため，Unityの素材のみという簡素な仕上がりとなってはしまったが，必要な情報は最低限表示できている．余力があれば，視覚班の方と協議して，専用の素材を製作するのも手である．

（※文責：麥谷悠悟）

3．2．3． 画面説明

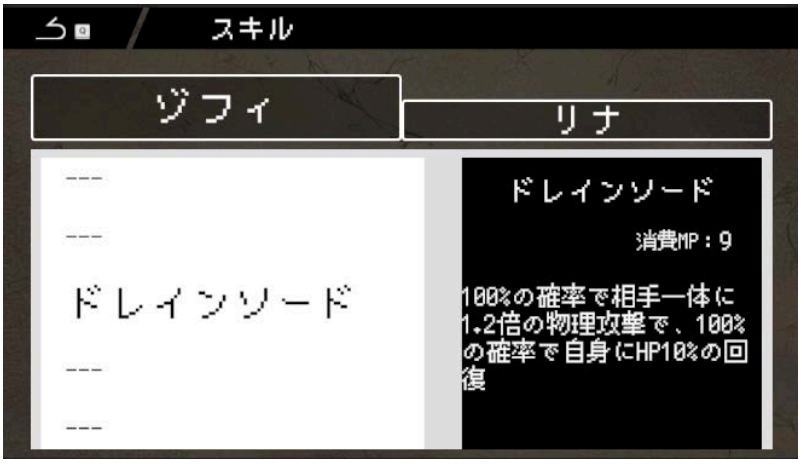
3．2．3．1． キャラクター詳細画面

この画面では，キャラクターの現在のステータス（HPや攻撃力など）を確認することができる．「タブ機能」を用いて，左右キーで参照するキャラクターを切り替えることが可能である．加えて，HPやMPはそれぞれ「HPバー」「MPバー」を使用することで，減り具合を視覚的に分かりやすくした．

（※文責：麥谷悠悟）

3．2．3．2． スキル一覧画面

この画面では，現在キャラクターが習得しているスキルを閲覧することができる．左側にスキルの一覧，右側に選択されているスキルの詳細が表示される．ここでも「タブ機能」を用いて，参照するキャラクターを切り替えることができる．



これから実装したい機能として，スキルを使用し，その効果を反映させることが挙げられる．詳細は「スキル/アイテム使用画面」に記述した．

（※文責：麥谷悠悟）

3．2．3．3． アイテム一覧

この画面では，現在パーティが所持しているアイテムを閲覧することができる．なお，アイテムはキャラクターごとに所持する仕様ではない．ここでは，「タブ機能」の用途を広げ，「通常アイテム」と「たいせつなもの」の二種類を左右キーで切り替えることができる．「通常アイテム」は，使用するとHPが回復するものなどの戦闘向けのアイテムを指し，「たいせつなもの」は，扉の鍵など，戦闘では使用しないが，ストーリー進行上必要なアイテムを指す．

「スキル一覧画面」同様，アイテムを使用し，その効果を反映するには至っていない．こちらもゲーム完成に向けて仕上げていきたい．

（※文責：麥谷悠悟）

3．2．3．4． スキルツリー画面

この画面では，キャラクターごとにスキルを習得することができる．スキルツリーの詳細については，「スキルツリー自動生成システム」の項目に記述している．

（※文責：麥谷悠悟）

3．2．3．5． スキル/アイテム使用画面

　この画面では，スキルもしくはアイテムを使用することができる．最初に，そのスキルもしくはアイテムを誰に使用するかを選択し，その後，「○○のHPが□□回復した！」というテキストを表示する．ただし，「スキル一覧画面」でも記述したが，実際にそれらの効果を適応するまでには至っていないため，ゲーム完成までに仕上げたい．

（※文責：麥谷悠悟）

3．3．戦闘

3．3．1． 概要

　戦闘システムは，敵キャラクターとの戦闘をしてキャラクターのレベルを上げたりゲーム内通貨を稼いだりする役割がある．また，敵キャラクターの見た目や行動を見て世界観を知ることができる役割がある．戦闘はコマンド選択式かつターン制になっている．コマンドは”こうげき”，”スキル”，”アイテム”，”にげる”，の四つである．各コマンドは別の項目で詳しく説明する．ターン制での行動順は各キャラクターの素早さを参照し値の大きい順に行動が処理される．戦闘のパラメーターは敵味方共通でHP，MP，攻撃力，防御力，魔法攻撃力，魔法防御力，素早さとなっている．スキルには行動妨害やHPが持続減少する状態異常や，攻撃などのステータスが上下するバフとデバフを付与するものがある．戦闘UIは画面下に味方キャラクターのHP，MP，名前が表示されており，画面右にはコマンド選択，画面中央には敵キャラクターの画像，画

面上部には敵味方の行動を文章で表示するメッセージウィンドウがある．戦闘終了時は戦闘時に増減した各数値を保存して次の戦闘に反映できるようにした．

（※文責：景山涼介）

3．3．1．1． 定義データ

戦闘をするにあたって，敵味方のデータを定義しなければ戦闘機能のコードが書けないため最初に作成した．固定値である敵味方のデータ呼び出しを簡単にするためにUnity内の機能にあるScriptableObjectを使用した．ScriptableObjectとは専用のC#スクリプトを作ることでUnity上で値を入力できるObjectを生成できるようになる機能のことである．作ったScriptableObjectは敵のステータス&行動パターン，味方の名前とID，スキルデータ，アイテムデータ，味方のレベルごとのパラメーターテーブル，経験値テーブル，エンカウントデータである．またScriptableObjectを呼び出すためにUnityの”Addressables”というアセットを使用した．このアセットはAddressable Groupを作りそこにScriptableObjectを登録すると呼び出せるものである．戦闘中に変化する値は別でC#スクリプトを作った．

（※文責：景山涼介）

3．3．1． 戦闘機能

3．3．2．1． UI

戦闘機能を実装するにあたって最初にUIの制御をするコードを作った．コマンドセレクトはEnum(列挙型)で作成した．コマンドを選択するとそのコマンドを受ける対象を選択するウィンドウが表示される．味方のステータス表示はScriptableObjectを参照して表示している．敵のSprite表示はScriptableObjectを参照して表示している．また，敵の数に応じて配置を調整できるようにUnity上で設定している．メッセージ表示は敵味方が行動したときのみに表示させるようにした．背景はマップの情報を受け取って対応したものを表示させている．

（※文責：景山涼介）

3．3．2．2． 戦闘処理

戦闘が始まるとまず敵の数が乱数によって決められる．そこから敵味方のデータをUIと戦闘用の変数に代入する．こうげきコマンドは攻撃を行うキャラクターの攻撃力と攻撃を受けるキャラクターの防御力を参照してダメージ計算式に応じてHPを減らす．ダメージ計算式は((攻撃力/2)-(防御力/4)*(0.8～1.2のランダムな値))である．スキルコマンドも同様に攻撃を行うキャラクターの攻撃力または魔法攻撃力と，攻撃を受けるキャラクターの防御力または魔法防御力を参照してダメージ計算式に応じてHPを減らす．また，スキルに設定されているMP消費量を参照して，スキルを使ったキャラクターのMPを減らす．スキルの種類には物理攻撃，魔法攻撃，回復，バフ，デバフ，状態異常があり，範囲は単体と全体がある．アイテムコマンドは使うアイテムを選択して，使用するキャラクターを選ぶ．アイテムの種類にはHP回復，MP回復，復活がある．逃げるコマンドは一体目の味方キャラクターと各敵キャラクターの素早さの値を参照して，ランダムな数値と50+(味方素早さ-敵素早さ)を計算してランダムな数値の方が大きいときに逃走失敗，小さければ逃走成功になる．行動順は各キャラクターの素早さを取得して大きい順に行われる．すべてのキャラクターの行動が終わるとまたコマンド選択に戻る．敵のHPが無くなると戦闘が終了する．

（※文責：景山涼介）

3．4．アニメーション

3．4．1． ゲーム進行中のムービーについて

ゲーム進行中に流れるムービーは，UnityのアニメーションとTimelineを用いて作成を行った．Unityのアニメーションはキャラクターの歩行アニメーションやアニメーション終了後のフェイドアウトなどの作成に用いた．TimelineはUnityのアニメーション機能で作成したアニメーションを組み合わせて一つのゲーム進行中に流れるムービーを作成するのに用いた．

（※文責：笹木颯太）

3．4．2． 戦闘中のスキルエフェクトについて

スキルエフェクトは，Unityのアニメーションで作成したエフェクトを関数（SkillEffectChangeのPlaySkillAnimation）で再生するようにした．PlaySkillAnimationはアニメーターで設定したcoditionsで設定したIDによって再生するエフェクトを切り替えることができる．

（※文責：笹木颯太）

3．5．タイトル画面

ゲームを起動すると「クリックしてスタート」が表示されプレイヤーにゲーム画面を一回クリックするように促した．この理由としては，Unityroomではゲーム画面を一回クリックしないとサウンドが流れないので，完成したゲームをUnityroomに投稿することを見据えてゲーム画面のクリックを促すような設計にした．

次に「クリックしてスタート」を押すと「はじめる」ボタンと「続きから」ボタンを表示させるようにした．「はじめる」を押すとセーブデータをリセットしてゲームをはじめられるようにした．現段階ではゲームの続きから始めることはできないので，今後「続きから」を押すとセーブされたところの続きからゲームを始められるようにする予定である．また，ゲームをクリアした状態で「続きから」を押すと，前回クリアした状態で獲得していたスキルポイントを持った状態でゲームが始まるようにする予定である．さらに，スキルは何も獲得していない状態になるのでまた自動生成されたスキルツリーでスキルを獲得する必要がある．

（※文責：笹木颯太）

3．6．会話

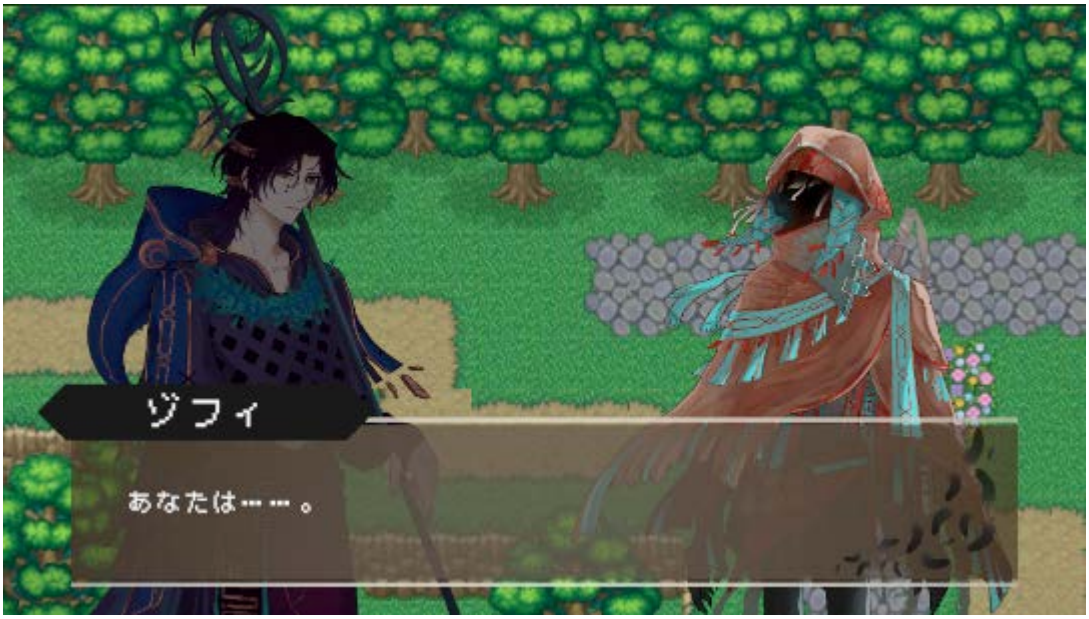
3．6．1． 概要

RPGにおいて重要な要素の一つである「会話」を，昨年度のコードを元に実装した．UIについて，前期では視覚班と配置を共有し，後期では送られた素材を実際に配置し，必要があれば改善してもらうようにやり取りした．

（※文責：麥谷悠悟）

3．6．2． 画面構成

今回実装した画面構成は以下の通りである．



（※文責：麥谷悠悟）

3．6．2．1． 会話内容

画面下部に，実際にキャラクターが話す内容を表示する．一度に全文を表示するのではなく，従来のRPGのように一文字ずつ表示されていく．ただし，表示されている最中に決定キーを押すことで，全文を表示することができる．これにより，特に今年度はマルチエンディングを有しているため，周回プレイにおいて快適さが保たれる．

（※文責：麥谷悠悟）

3．6．2．2． 話者

先ほどの「会話内容」において，誰が話しているかを「会話内容」の枠の左上に表示する．次の項目の「キャラクター画像」と関連がある．

（※文責：麥谷悠悟）

3．6．2．3． キャラクター画像

会話中の背景の右側もしくは左側に，キャラクターの立ち絵を表示する．「話者」と表示されているキャラクターが一致している際は，そのキャラクターの立ち絵が明るく表示される．一方で，一致していない場合は，立ち絵が暗く表示される．これにより，誰が話しているかが視覚的に分かりやすくなった．

（※文責：麥谷悠悟）

3．6．2．4． 背景画像

「キャラクター画面」のさらに後ろに，背景を描写することが可能．特に指定がない場合，マップを背景に会話を実行する．

（※文責：麥谷悠悟）

3．6．2．5． 選択肢

会話の中に選択肢があった場合にのみ表示される．選択肢を表示させる方法は，「スクリプトエンジン」の「会話エンジン」に記述した．プレイヤーは，最大三つの選択肢の中から任意の選択を行うことができる．

（※文責：麥谷悠悟）

3．7．サウンド

BGMと効果音の再生を行うために去年のSoundManager.csを用いた．BGMはPlayBGM，効果音はPlaySEで再生するようにした．どちらも引数はID（Listの要素番号）と音量（float）である．

（※文責：笹木颯太）

3．8．スクリプトエンジン

3．8．1． 概要

スクリプトエンジンとは，Jsonファイルに内容を記述し，それを読み込んでゲーム内のオブジェクト等に反映させるものを指す．なお，昨年度で使用された機能を用いつつ，今年度用に必要な機能を追加/削除することで実装した．スクリプトエンジンを構成するエンジンについて，以下のものが挙げられる．

- オブジェクトエンジン
- マップエンジン
- 会話エンジン
- 戦闘エンジン

各エンジンはMessagePack for C#を使用して，開発ではStreamingAssetsフォルダで，成果物はResourcesフォルダを使用してJsonファイルを読み込んだ．成果物にResourcesフォルダを使用した理由は，プラットフォームの制限により，StreamingAssetsフォルダが使用できないためである．

このような手法を取った理由として，時間的問題が挙げられる．例えば，10月にシナリオが完成し，そこからゲームに統合するとなると，完成が間に合わない可能性がある．そこで，スクリプトエンジンを用いることで，プログラミング能力を必要としない簡単な記述でストーリー進行などが可能となる．これにより，システム班ではプログラムを担当し，その後シナリオ班ではスクリプトを担当することができるため，今回の短期間のゲーム制作において有効であった．

（※文責：麥谷悠悟）

3．8．2． 各エンジンの説明

3．8．2．1． オブジェクトエンジン

オブジェクトデータは，どこに何のイベントを発生させたいかを記述したものである．具体的には次のような要素を含む．

- 座標（Location）
    - シーン名
    - x座標，y座標
- イベント（EventName）
- トリガータイプ（TriggerType）
- フラグ状態（FlagCondition）
    - 発生条件
    - フラグ変更

各要素について，次の項目から詳しく記述する．

（※文責：麥谷悠悟）

3．8．2．1．1． 座標

座標には，「シーン名」と「x座標」，「y座標」が記述される．これはイベントを起こす場所を指しており，複数指定することも可能．

（※文責：麥谷悠悟）

3．8．2．1．2． イベント

実際にどのようなイベントを発生させるのかを記述する．今年度は以下のようなイベントを用意した．

- シーン変更：任意のシーンへ移動する．
- シーン内移動：同じシーンの中で，任意の座標へ移動する．
- 会話：会話データが記述されているJsonファイルを読み込み，会話を開始する．
- アイテム獲得：任意のアイテムを獲得する．
- バトル開始：戦闘データが記述されているJsonファイルを読み込み，戦闘を開始する．
- アニメーション開始：任意のアニメーションを呼び起こし，再生する．
- 仲間加入：任意のキャラクターをパーティメンバーに加える．
- 全回復：パーティメンバー全員のHPやMPを回復する．
- 全体マップ表示：プレイヤーがどのシーンに移動するかを決められるマップを表示する．

（※文責：麥谷悠悟）

3．8．2．1．3． トリガータイプ

イベントが発生するタイプを整数で記述する．例えば，「決定キーを押すことでイベントが始まる」ものや，「特定の座標に行っただけでイベントが始まる」ものなどが存在する．ここではそれらを指定する．

- 「０」：プレイヤーの座標が，オブジェクトデータの座標と一致したとき
- 「１」：プレイヤーの座標がオブジェクトデータの座標の上下左右１マスにいて，プレイヤーがオブジェクトデータの座標がある方に向いていて，決定ボタンをおしたとき

- 「２」：プレイヤーの座標がオブジェクトデータの座標と一致している，または，上下左右１マスにいてプレイヤーがオブジェクトデータの座標がある方向に向いているときに決定ボタンを押したとき（＝「１」と「２」のハイブリット）
- 「３」：プレイヤーの座標がオブジェクトデータの座標と一致していて，決定キーを押したときに発生

（※文責：麥谷悠悟）

３．８．２．１．４． フラグ状態

「イベントが発生するフラグの条件」と，「イベント終了後にどのフラグの変更を行うか」を記述する．これにより，選択肢によって今後のイベントが変わるため，マルチエンディングに必要不可欠な要素である．

（※文責：麥谷悠悟）

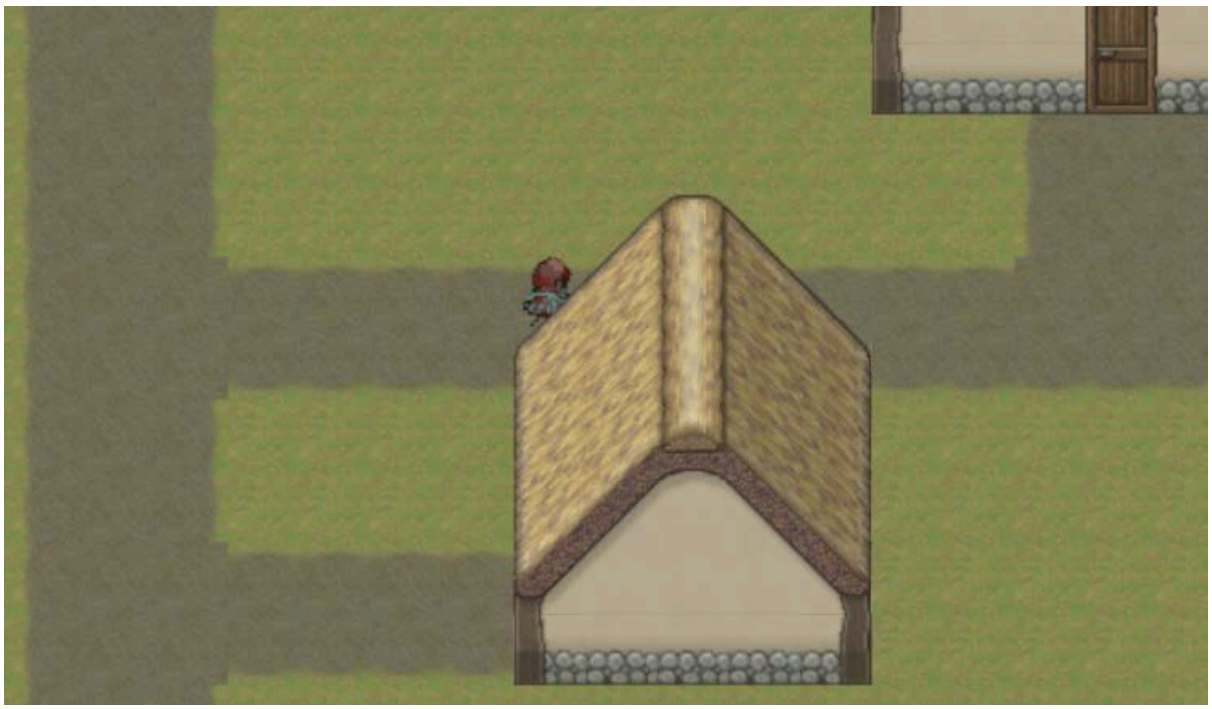
３．８．２．２． マップエンジン

マップエンジンとは，JSONファイルに記述されたマップデータをシーン読み込み時に読み込んで各シーンのTilemapオブジェクトにタイルを配置するシステムである．プレイヤーとオブジェクトの前後関係を正しく表現するため，マップデータを以下の3層+衝突判定層に分割して管理している．

- Tiles：衝突判定用データ．WalkableTilesで歩行可能な文字を決める
- StylesFront：最前面．プレイヤーを隠す建物の上部や岩などで使用
- StylesMiddle：中間面．プレイヤーの背後だが，StyleBackの上に配置の際に使用
- StylesBack：最背面．床や地面などを配置

マップデータとタイルの対応については，一文字とマップチップを紐づけている．このとき，誰でも簡単に編集できるようにStreamingAssetsフォルダを利用している．

以下の図のように屋根をStyleFrontに，地面をStylesMiddleとStylesBackに配置することで奥行きを再現することができる．



（※文責：田本亜樹斗）

3．8．2．3． 会話エンジン

会話データは，どのキャラクターたちがどのようなことを話すのかを記述したものである．具体的には次の要素を含む．

- 内容（Content）
    - 話者（Speaker）
    - セリフ（Text）
    - 背景画像指定（ChangeImage）
        - 配置場所（ImageName）
        - 画像名（SpriteName）
    - 選択肢（QuestionData）
        - 回答（Answer）
        - フラグ変更（NextFlag）

        - 次の会話データ（NextTalkData）
    - 音楽（BGM）
    - 効果音（SE）

内容（Content）を配列にすることで，話者が入れ替わる自然な会話を可能とした．各要素について，次の項目から詳しく記述する．

（※文責：麥谷悠悟）

3．8．2．3．1． 話者，セリフ

話者（Speaker）には，次に指定するセリフを誰が話しているかを記述する．セリフ（Text）にはその名の通り，セリフを記述する．

（※文責：麥谷悠悟）

3．8．2．3．2． 背景画像指定

背景画像指定（ChangeImage）には，画像を配置する場所（ImageName）と画像名（SpriteName）が存在する．配置場所（ImageName）には，「LeftPanel」「RightPanel」「BackgroundPanel」のいずれかを指定する．画像名（SpriteName）には，キャラクターの立ち絵のファイル名を記述する．

（※文責：麥谷悠悟）

3．8．2．3．3． 選択肢

選択肢（QuestionData）には，疑問に対する回答（Answer），回答したことによるフラグ変更（NextFlag），次の会話データ（NextTalkData）を指定する．これにより，選択肢によるマルチエンディングが可能となった．

（※文責：麥谷悠悟）

3．8．2．3．4． 音楽，効果音

音楽（BGM）を指定することにより，任意のBGMを流すことができる．効果音（SE）も同様に流すことができる．

（※文責：麥谷悠悟）

3．8．2．4． 戦闘エンジン

戦闘データは，どの敵キャラとバトルするのかを記述する．具体的には，次の要素を含む．

- 敵キャラクター（EnemyIds）
- 戦闘音楽（BGM）
- エンカウントメッセージ（EncounterMessage）
- 勝利時フラグ（WinFlags）
- 敗北時フラグ（LoseFlags）

各要素について，次の項目から詳しく記述する．

（※文責：麥谷悠悟）

3．8．2．4．1． 敵キャラクター

バトルする際の敵キャラクターのIDを指定する．最大５体まで記述可能．

（※文責：麥谷悠悟）

3．8．2．4．2． 戦闘音楽

バトル中の戦闘音楽を指定する．雑魚敵なら雑魚敵用のBGM，中ボスなら中ボス用のBGMを流すなど，柔軟な対応が可能である．

（※文責：麥谷悠悟）

3．8．2．4．3． エンカウントメッセージ

バトル開始時に表示するテキストメッセージを指定する．「○○が現れた！」だけでは面白味がないと感じたため，自由に表示できるようにした．

（※文責：麥谷悠悟）

3．8．2．4．4． 勝利時フラグ，敗北時フラグ

勝利時フラグ（WinFlags）は，戦闘に勝利した際に，どのフラグをどのように変更するのかを記述する．敗北時フラグ（LoseFlags）は，戦闘に敗北した際のフラグ変更を記述する．これにより，戦闘の勝ち負けによるマルチエンディングも可能となった．

（※文責：麥谷悠悟）

4． スキルツリー自動生成システム

本章で扱う用語についての解説を以下に記載する．

- ノード：グラフにおける一般的な意味と同じく，スキルツリーの点に該当する部分を示す．プレイヤーがゲーム内で獲得できる報酬が配置される．
- エッジ：グラフにおける一般的な意味と同じく，スキルツリーの線に該当する部分を示す．プレイヤーが進むことのできる順路を表す．
- 列：スキルツリー内における原点からの距離を示す．この距離が同じであることを「同列にある」と表す．
- スキル：ゲームシステムの一つであり，戦闘中にコスト（本作品ではMP）を消費して通常より強力な効果を持つ技を示す．
- ステータス：攻撃力，防御力など，キャラクターの能力を数値で表したパラメータを示す．スキルツリーで獲得するほか，レベルアップでも数値が上昇する．
- 効果量：攻撃系のスキルならばダメージ量を，回復系のスキルならば回復量を示す．
- 発動確率：スキルが失敗せずに発動する確率を示す．
- 対象範囲：スキルの効果が及ぶ対象を示す．本分析では「敵一体」「敵一グループ」「敵全体」もしくは「自身のみ」「仲間一人」「仲間全体」でダミー変数化している．

- 発動回数：スキルの効果が発動する回数，あるいは継続するターン数を示す．

（※文責：伊藤颯）

4．1． 仕組み

今回スキルツリーの自動生成をするにあたって自作のアルゴリズムを考え，それに基づいて必要なデータを集めた．データ分析によって出された確率を基に，各階層でのノードの配置や各ノード間での結びつき，スキル・ステータスアップの獲得の配置を行いスキルツリーを自動的に生成するシステムを作成した．また，各ノード間での結びつきを生成するにあたって，スキルツリーの途中に行き止まりを作るようなアルゴリズムを考えた．今回このようなアルゴリズムにした理由は，RPGのスキルツリーにおいてスキルツリーの途中に行き止まりがあるものが多かったことに起因する．

（※文責：笹木颯太）

4．1．1． 各階層でのノードの配置

ノード数はスキルの数によって決めるようにした．スキル・ステータスアップの統計的比率を基にノードの数を確定する．次にノードを配置していく．初めにノードを配置していくにあたってスキルツリーの序盤・中盤・終盤に配置の順番を分ける．次に，一層分におけるノード

数の確率を基にノードの配置を行う．なお，一層分におけるノード数の確率は序盤・中盤・終盤で異なる．

（※文責：笹木颯太）

4．1．2． 各ノード間での結びつき

ノード配置の完了後に最終層のノードから順に一つ前の層の1ノードとランダムにエッジを結びつける．次に，分析に基づいた確率に沿ってノードに接続できるエッジの数を決めていく．最後にすべてのノードがエッジ数の制限数に達するまで一番前のノードから順に一つ後ろの層のノードとランダムにエッジを結ぶ．

（※文責：笹木颯太）

4．1．3． スキル・ステータスアップの獲得の配置

ノード間の結びつき完了後に，最初のノードから順に分析による確率に沿ってスキルまたはステータスアップを配置するかを決める．次に，評価値に沿ってスキルを配置するノードにスキルの内容を決める．最後に，ステータスアップを配置するノードにアップさせるステータスの内容をランダムに決める．

（※文責：笹木颯太）

4．2．分析結果

本章では，開発の各工程ごとに実施した分析内容について述べる．なお，各分析結果の図表は付録の項に掲載する．

外形の形成およびノードの報酬指定に関する分析では，スキルツリーを「勇者型」「戦士型」「魔法使い型」の三種類に分類し，各分析対象作品から該当するスキルツリーを選出したうえで分析を行った．分類基準として，当該スキルツリーを有するキャラクターが作品の主人公である場合を「勇者型」，HPや攻撃力などのステータスが高く，武器による物理攻撃を主軸とするキャラクターの場合を「戦士型」，魔法攻撃に特化したキャラクターの場合を「魔法使い型」と定義した．また，スキルツリーを，列数を基準として三つに区分し，それぞれを原点に近いものから「序盤」「中盤」「終盤」と定義した．

（※文責：伊藤颯）

4．2．1． 外形の生成

各階層でのノードの配置や各ノード間での結びつきに関する分析である．同一列に存在するノード数を集計し，その出現傾向を確率分布として整理した．結果を図1〜4に示す．

また，同一列に存在するノード数ごとに分類し，各ノードに対して前列ノードから接続される入力エッジ数と，当該ノードから後列ノードへ接続される出力エッジ数の組み合わせパターンを確率的に整理した．結果を表2，表3，表4，に示す．

さらに，スキルツリー全体を列数に基づいて「序盤」「中盤」「終盤」の三段階に等分し，それぞれに属するノード数を算出したうえで，その割合を示した．結果を図5に示す．

同列のノード数確率については，どの類型においても序盤と終盤は少なく，中盤では多いという傾向が見られた．

スキルツリー全体におけるノード分布については，同列のノード数確率と同様に，中盤に多くのノードが配置されていた．

（※文責：伊藤颯）

4．2．2． 各ノードの報酬指定

スキルツリーを「序盤」「中盤」「終盤」の三段階に区分したうえで，各ノードに設定されている報酬内容を同一列に存在するノード数ごとに調査し，その構成比を算出した．結果を表5に示す．

さらに，ノードに指定されている報酬の種類を状態とみなし，その状態遷移の確率を算出した．結果を表6に示す．

どの種類も序盤から中盤にかけてスキルの比率が下がり，中盤から終盤にかけて上がるという推移であることが示された．

また，スキルツリー全体としては勇者がステータスの比率が多く，戦士はスキルの比率が多い．魔法使いは均衡しているがスキルの比率がやや高い傾向にあることが確認された．

（※文責：伊藤颯）

4．2．3． スキルの評価および配置

スキルの評価値を算出する際に用いる加重和法の係数を決定するため，分析対象とした作品に登場するスキルのうち，攻撃系，回復系のカテゴリに分類できるスキルのデータを収集した．

次に，各スキルについて「効果量」「発動確率」「対象範囲」「発動回数」の四要素を抽出し，それぞれに対して Min-Max 正規化を行った．対象範囲の要素についてはダミー変数化した値を使用した．

正規化後のデータを用いて重回帰分析を実施し，得られた各要素の回帰係数を加重和法における各要素の係数として適用した．重回帰分析の結果を表7，表8に示す．

攻撃系スキルについては，スキルの効果量が最も影響の強い要素であることが示された．回復系スキルについても同じく，効果量が最も強い影響を有しており，発動確率はデータに有意性が見込めず，影響がないことがわかった．

（※文責：伊藤颯）

4．3． 課題

現状のスキルツリー自動生成システムには，外形の生成を行う工程とスキルの評価と配置を行う工程において，いくつか改良すべき課題が確認されている．システムの工程ごとに述べていく．

（※文責：伊藤颯）

4．3．1． 外形の生成

外形の生成を行う工程について述べる．スキルツリーのノードおよびエッジを配置する段階において，スキルツリーの中盤付近で，ノードからの出力エッジ数が0となるケースが確認された．このようなケースが発生すると，本来意図しない位置にスキルツリーの終点が生成されてしまうという問題が生じてしまう．

（※文責：伊藤颯）

4．3．2． スキルの評価および配置

現行システムにおけるスキル評価では，各スキルを構成する四つの要素のみに基づいて評価値を算出している．しかし，分析対象としたRPGにおけるスキルには，属性相性，デメリット効果，独自性の高い効果など，評価に影響を及ぼす要素が他にも多数存在する．本来考慮すべき要素を十分に反映できていない点から，現在の評価方式は柔軟性および適応性に欠けているといえる．

また，バフ系，デバフ系，状態異常系のスキルについては，戦闘における影響度がゲームのデザインに大きく依存するため，分析によって一貫した評価値を得ることが困難であった．その結果，現行システムでは，攻撃系および回復系スキルの平均値を適用するという暫定的な対応を行っている．この問題への対応として，分析データのみでは対応が困難な要素については，ユーザー自身に数値の決定を委ねることにする．

さらに，スキルの配置方法についても課題が存在する．現行システムでは，生成されるスキルツリーの外形は多様に変化する一方で，スキルの配置は評価値という固定的な基準に基づいて決定される．そのため，配置結果においては変化に乏しい傾向が見られる．この問題の改善策として，評価値に対して一定範囲内でランダムかつ微小な変動を加えることで，配置にランダム性を導入する手法を検討している．

（※文責：伊藤颯）

4．4． システムについての考察

本システムの開発を通して，スキルツリーの外形に関しては他作品の分析データを適用しても顕著な問題は確認されなかった．一方で，スキルの評価および配置の方法に関しては複数の問題が確認された．このことから，スキルツリーの外形には構造上の明確な傾向が存在するのに対し，その内容については作品固有の特徴が色濃く反映されていることが示唆される．よって，スキルツリーの外形部分には汎用的な機能性が求められる一方で，内容部分には高い独自性が求められると考えられる．

以上を踏まえて，このスキルツリー自動生成システムにはスキルの評価および配置の工程において，ユーザーが調整可能な余地を十分に確保することが重要であることがわかった．

しかし，本作品が未完成である都合上，本システムの評価を行えていないのが現状である．そのため，本作品が完成し次第，定性的評価を行っていく必要がある．

（※文責：伊藤颯）

5． 統合作業

5．1． 成果物の統合

成果物の統合はシナリオ班，視覚班，音響班の作った素材をGoogle ドライブにアップロードし，システム班がそれをUnityに取り込んだ．また，Unityで実際に動作してみて改善が必要であると判断した素材については適宜素材を作ったメンバーへ報告し作り直すかの相談をした．

（※文責：景山涼介）

5．2． デバッグ

デバッグは手の空いた人からしてもらい，GitからGithubにあるUnityのデータをダウンロードしてもらったうえで，各々のPCで動かして行った．バグや不具合があった場合はGithubのIssuesに報告してもらった．

（※文責：景山涼介）

5．3． 他班との連携

システム班は各班から集めた成果物を統合する役目も担っている．そのため，ほかの班との連携が重要になってくる．シナリオ班とは，物語やマップなどのイメージのすり合わせと会話データやマップデータなどのテキストデータの作成の依頼を行った．マップのイメージのすり合わせの際は，視覚班の代表とともに話し合った．すり合わせの際，お互いのイメージが一致しているかを確認するのが重要だった．なぜなら，今後の作業で手戻りが発生してしまう可能性があったからだ．視覚班には，画面UIやマップチップ，キャラチップの依頼をした．音響班には，BGMやSEの依頼をした．音響班への依頼が遅れてしまっていたので，音響班が様々な音源の項目を作成し，システム班がそこから必要な音源を音響班に伝え，作成する方向になった．この方向にしたことで，実装までの流れはスムーズになった．各班の成果物はGoogle driveで共有し，そこからUnityに移行した．しかし，移行する人をしっかりと決めていなかったので，画像データが複数存在してしまっていた．

（※文責：田本亜樹斗）

6． 結果

6．1． 目標の達成度

私たちは，当初計画していた「メニュー」，「マップ」，「戦闘」，「会話」の4つの主要機能を最低限の実装まで実現することができた．また，タイトルやサウンド，アニメーションの機能を実装することができ，体験版として最終発表の場で様々な方がゲームに触れることができた．だが，アイテムの機能やフラグリストやテキストデータやオブジェクトデータが未完成で終わったため，完全な完成にはならなかった．そのため，秋葉原の課外発表会を完成の目標に変更した．また，Git/GitHubを活用した複数人での共同開発体制を構築できた．

（※文責：田本亜樹斗）

6．2． 進捗管理と他班との連携

音響班の音声の発注方法の変更やマップ制作を複数人で行うことなど，当初の予定ではうまくいかないと判断し，すぐに臨機応変に対応できたのはよかった．だが，スケジュール管理と一人一人の作業量の偏りにより，開発に時間がかかってしまっていた．

（※文責：田本亜樹斗）

6．3． スキルツリー自動生成システム

本システムは，運用することが可能な段階まで完成させることができた．しかしながら，現段階で本作品は未完成であり，定性的な評価を行うことが困難である．そのため，現時点では把握できていない課題が存在する可能性が高い．

（※文責：伊藤颯）

7． 考察

7．1． システム開発班

Unity未経験者が多いなか，体験版までの完成に至れたのは，Unityの学習を実践重視に切り替えたことや，昨年度のコードを用いる中でコードの理解に務めたことが大きな要因だと推測される．一方で，画像やマップチップの管理を早々に決めなかったことによりデータや紐づけの欠落や「メニュー」，「マップ」，「戦闘」，「会話」の四つの主要機能をさらに細分化しなかったことによる作業量の偏りが開発の遅延と混乱を招く一因となった．並行作業でのルール

作りの重要性を学ぶ機会となった．作業量の偏りにより，作業終了の個人差がかなり大きかった．

（※文責：田本亜樹斗）

7．2． スキルツリー自動生成班

　システム担当が必要としているデータと分析担当が収集したデータに齟齬が生じるなど，開発段階でいくつかの問題が発生した．お互いの意思疎通や情報交換が不十分であったと考えられる．しかし，システム全体のアイデアやアルゴリズムを早期の段階で練ることができたため，開発途中での変更や問題の対処を行うことができた．このことから，開発の見通しや設計を，綿密に話し合ったうえで早期に済ませ，時間的猶予を持つことが重要であると学ぶことができた．

（※文責：伊藤颯）

8． 参考文献・付録

8．1． 参考文献

[1]ドラゴンクエスト大辞典を作ろうぜ！！第三版 Wiki*［オンライン］．https://wikiwiki.jp/dqdic3rd/ （参照 2026-01-14）

[2]極限攻略データベース［オンライン］．https://kyokugen.info/ （参照 2026-01-14）

[3]Hrs-Game｜ゲーム攻略サイト［オンライン］．https://main-hrs-game.ssl-lolipop.jp/ （参照 2026-01-14）

[4]Unityゲーム開発入門！スキルツリーの実装方法 #1．Unityゲームスタジオ スタジオしまづ［YouTube動画］．2019-08-20．https://www.youtube.com/watch?v=u3m2eB3SrQU （参照 2026-01-14）

[5]ドラゴンクエストXI 過ぎ去りし時を求めて 『ビデオゲーム』 (2017). スクウェア・エニックス

[6]ドラゴンクエスト X オフライン 『ビデオゲーム』 (2022). スクウェア・エニックス

[7]ドラゴンクエストIX 星空の守り人 『ビデオゲーム』 (2010). スクウェア・エニックス

[8]ドラゴンクエストⅧ 空と海と大地と呪われし姫君 『ビデオゲーム』 (2004). スクウェア・エニックス

[9]ドラゴンクエストIII そして伝説へ… 『ビデオゲーム』 (2024). スクウェア・エニックス

[10]FINAL FANTASY Ⅶ REMAKE 『ビデオゲーム』 (2024). スクウェア・エニックス

[11]FANTASIAN Neo Dimension 『ビデオゲーム』 (2023). Mistwalker（開発）, スクウェア・エニックス（発売）

（※文責：伊藤颯）

8．2． 付録

表2　ノード数ごとの入出力エッジ（勇者型）

ノード数		出力0	出力1	出力2	出力3	出力4
1	入力1	0.07	0.27	0.13	0.13	0.00
	入力2	0.20	0.00	0.13	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.07	0.00	0.00	0.00	0.00
2	入力1	0.17	0.39	0.33	0.00	0.00
	入力2	0.00	0.00	0.00	0.11	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
3	入力1	0.18	0.29	0.32	0.04	0.00
	入力2	0.11	0.00	0.00	0.00	0.00
	入力3	0.00	0.07	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
4	入力1	0.25	0.40	0.15	0.00	0.00
	入力2	0.10	0.10	0.00	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
5	入力1	0.20	0.25	0.30	0.10	0.00
	入力2	0.00	0.00	0.05	0.10	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
6	入力1	0.00	0.25	0.75	0.00	0.00
	入力2	0.00	0.00	0.00	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
7	入力1	0.10	0.67	0.03	0.00	0.00
	入力2	0.13	0.07	0.00	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00

表3　ノード数ごとの入出力エッジ（戦士型）

ノード数		出力0	出力1	出力2	出力3	出力4
1	入力1	0.10	0.20	0.50	0.10	0.00
	入力2	0.00	0.00	0.10	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
2	入力1	0.32	0.14	0.36	0.00	0.00
	入力2	0.00	0.00	0.00	0.00	0.09
	入力3	0.09	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
3	入力1	0.48	0.37	0.04	0.00	0.00
	入力2	0.04	0.00	0.00	0.00	0.00
	入力3	0.00	0.07	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
4	入力1	0.09	0.26	0.42	0.05	0.00
	入力2	0.00	0.12	0.02	0.05	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
5	入力1	0.25	0.25	0.19	0.00	0.00
	入力2	0.06	0.25	0.00	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
6	入力1	0.04	0.54	0.17	0.08	0.00
	入力2	0.00	0.13	0.04	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
7	入力1	0.00	0.76	0.05	0.00	0.00
	入力2	0.00	0.19	0.00	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00

表4　ノード数ごとの入出力エッジ（魔法使い型）

ノード数		出力0	出力1	出力2	出力3	出力4
1	入力1	0.19	0.15	0.26	0.07	0.00
	入力2	0.15	0.00	0.04	0.00	0.07
	入力3	0.04	0.00	0.00	0.00	0.00
	入力4	0.04	0.00	0.00	0.00	0.00
2	入力1	0.08	0.33	0.25	0.08	0.00
	入力2	0.08	0.00	0.00	0.00	0.00
	入力3	0.00	0.17	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
3	入力1	0.05	0.24	0.29	0.14	0.00
	入力2	0.14	0.00	0.14	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
4	入力1	0.10	0.50	0.10	0.05	0.00
	入力2	0.00	0.20	0.03	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.03	0.00	0.00	0.00	0.00
5	入力1	0.13	0.33	0.27	0.00	0.00
	入力2	0.07	0.00	0.07	0.13	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
6	入力1	0.00	0.63	0.11	0.00	0.00
	入力2	0.26	0.00	0.00	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00
7	入力1	0.05	0.76	0.05	0.05	0.00
	入力2	0.00	0.10	0.00	0.00	0.00
	入力3	0.00	0.00	0.00	0.00	0.00
	入力4	0.00	0.00	0.00	0.00	0.00

表5　ノードの報酬割合

分類	段階	スキル	ステータス
勇者型	序盤	0.439	0.561
	中盤	0.387	0.613
	終盤	0.454	0.546
	全体	0.419	0.581
戦士型	序盤	0.509	0.491
	中盤	0.465	0.535
	終盤	0.817	0.183
	全体	0.583	0.417
魔法使い型	序盤	0.586	0.414
	中盤	0.376	0.624
	終盤	0.653	0.347
	全体	0.517	0.484

表6　同列中のノード数ごとによるノード報酬割合

分類	スキル個数	ステータス個数	割合
勇者型	1	0	0.525
	0	1	0.475
	2	0	0.051
	1	1	0.475
	0	2	0.475
	3	0	0.182
	2	1	0.364
	1	2	0.273
	0	3	0.182
	4	0	1.000
	2	3	1.000
戦士型	1	0	0.667
	0	1	0.333
	2	0	0.400
	1	1	0.222
	0	2	0.378
	3	0	0.200
	2	1	0.200
	1	2	0.600
	0	3	0.000
	4	0	0.000
	5	1	1.000
魔法使い型	1	0	0.795
	0	1	0.205
	2	0	0.233
	1	1	0.465
	0	2	0.302
	3	0	0.090
	2	1	0.273
	1	2	0.000
	0	3	0.636
	2	2	0.500
	1	3	0.500
	4	1	1.000

表7　重回帰分析結果（攻撃系スキル）

	係数	標準誤差	t値	p値	下限95%	上限95%	VIF
切片	0.0895	0.283	0.316	0.753	-0.472	0.651	
効果量	0.843	0.075	11.238	0	0.694	0.992	1.017983
発動確立	-0.1156	0.283	-0.408	0.684	-0.677	0.446	1.004884
対象範囲	0.0858	0.022	3.965	0	0.043	0.129	1.017733
発動回数	0.1077	0.022	4.873	0	0.064	0.152	1.009563

回帰統計（攻撃系スキル）	
重決定R2	0.587
補正R2	0.572
残差標準誤差	0.196
観測数	117

表8　重回帰分析結果（回復系スキル）

	係数	標準誤差	t値	p値	下限95%	上限95%	VIF
切片	-0.136	0.118	-1.155	0.251	-0.369	0.097	
効果量	0.5195	0.063	8.31	0	0.396	0.643	1.056845
発動確立	-0.136	0.118	-1.155	0.251	-0.369	0.097	89.767695
対象範囲	0.1912	0.045	4.211	0	0.101	0.281	1.089217
発動回数	-0.0126	0.021	-0.593	0.555	-0.55	0.029	1.03244

回帰統計（回復系スキル）	
重決定R2	0.465
補正R2	0.451
残差標準誤差	0.134
観測数	34

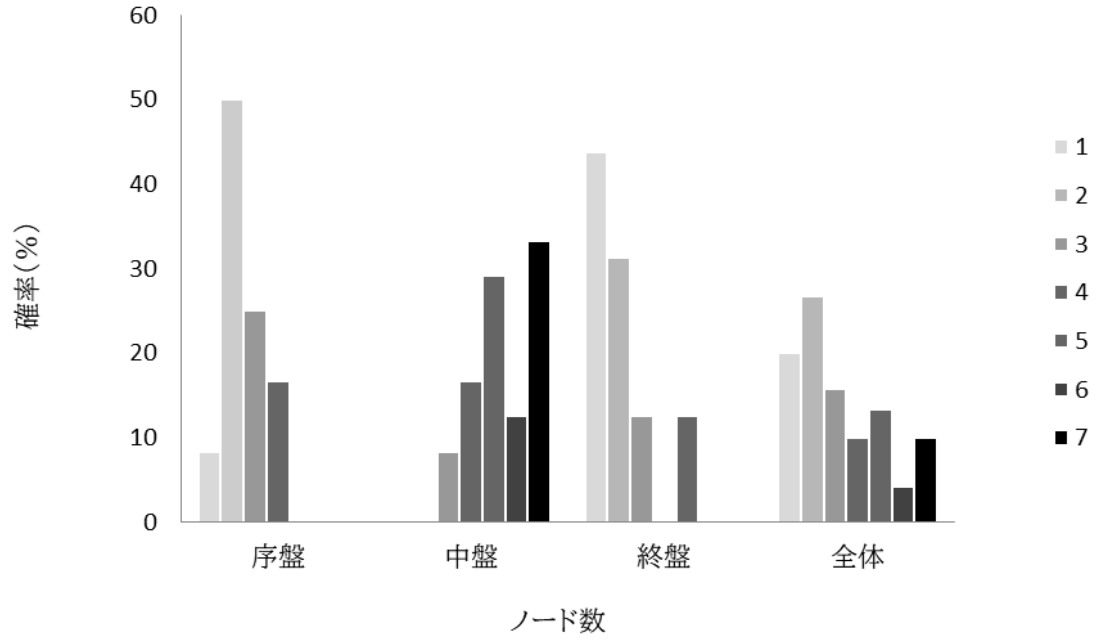


図1　同列のノード数確率　勇者型

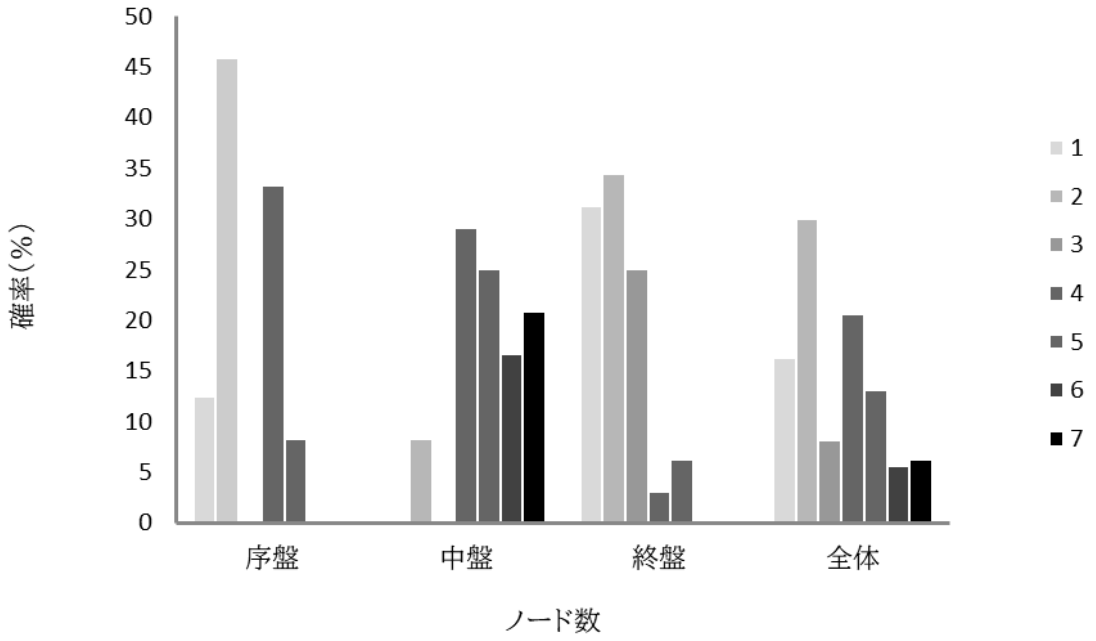


図2　同列のノード数確率　戦士型

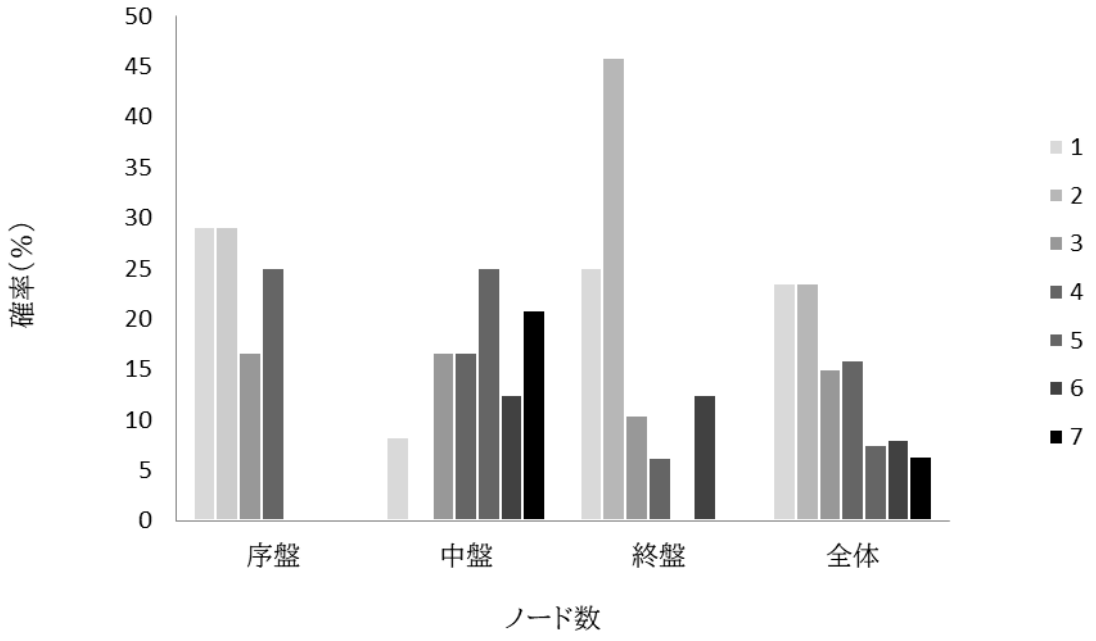


図3　同列のノード数確率　魔法使い型

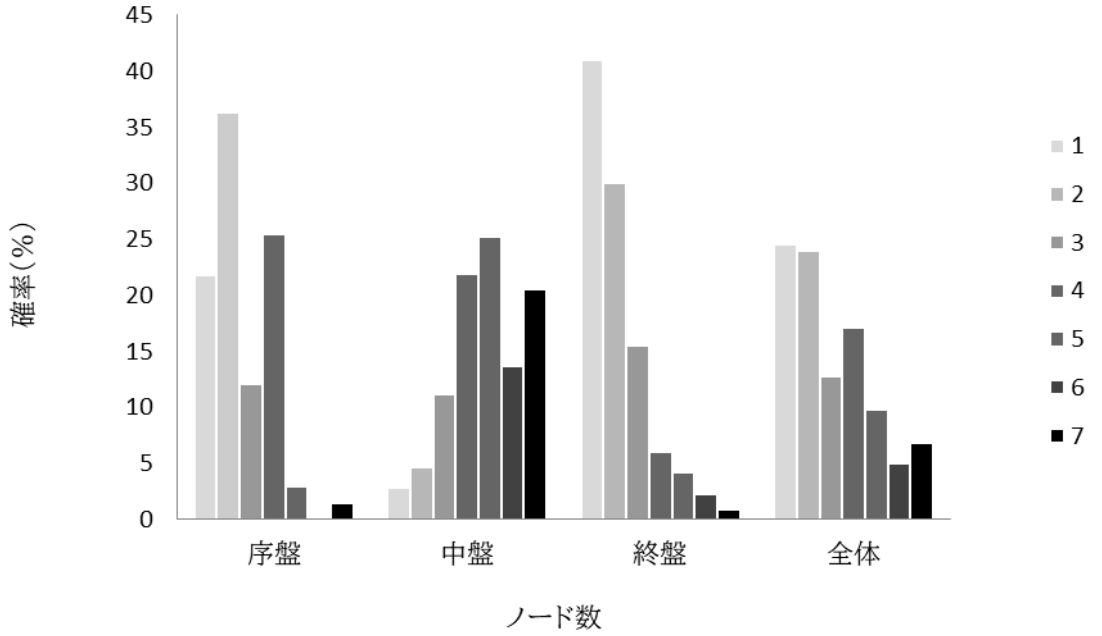


図4　同列のノード数確率　総合

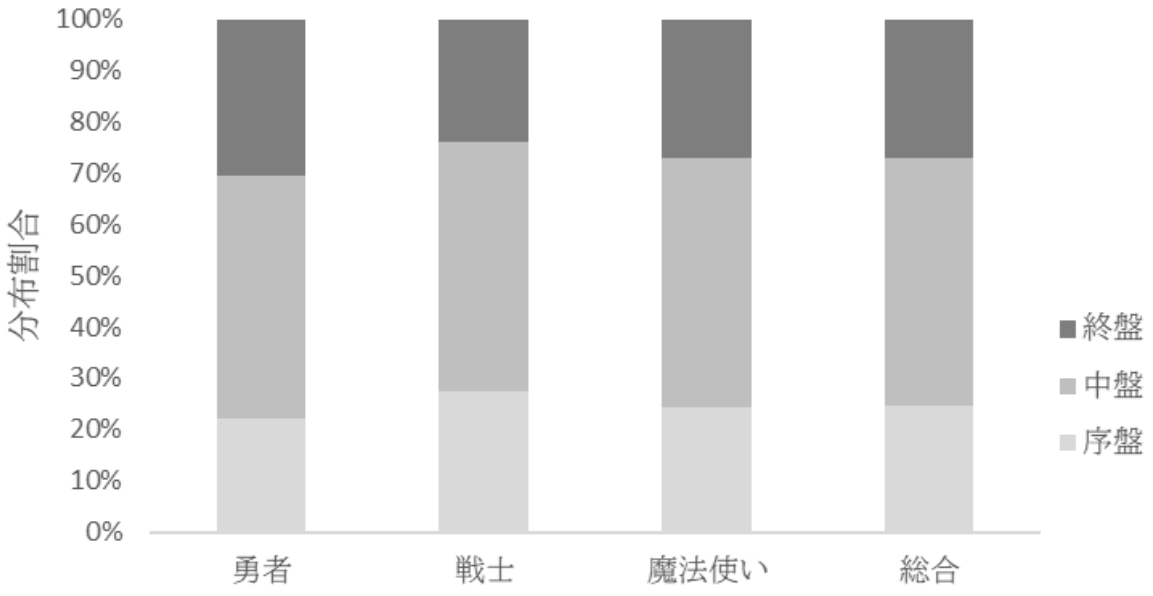


図5　スキルツリー全体におけるノード分布