Graph Analytics für Performance Messungen und adaptive Lernpfade

Bachelorarbeit

KIT – Karlsruher Institut für Technologie
Fraunhofer IOSB – Fraunhofer-Institut für Optronik, Systemtechnik und Bildauswertung

Rafael Baur

15. März 2022

Verantwortliche Betreuer: Prof. Dr.-Ing. Jürgen Beyerer
Prof. Dr.-Ing. Thomas Längle
Betreuender Mitarbeiter: Dipl.-Inf. Alexander Streicher

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die Arbeit selbständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht habe und die Satzung des Karlsruher Instituts für Technologie zur Sicherung guter wissenschaftlicher Praxis in der gültigen Fassung beachtet habe.

Karlsruhe, den 15. März 2022

(Rafael Baur)

Inhaltsverzeichnis

Kurzfassung

Deutsch Ideal Paths Models (IPMs) sind Referenzmodelle, die ideale Sequenzen an Lernobjekten durch ein Lernsystem als Pfade in Graphen, die sogenannten Ideal Paths (IP), modellieren. Ideal bedeutet, dass die IPs den Lernpfaden von Experten entsprechen und das Befolgen dieser IPs ein effektives und effizientes Lernen ermöglicht. Mithilfe von IPMs können Lernpfade von Nutzern durch individuelle Empfehlungen angepasst werden, um mithilfe dieser Adaptivität dem Nutzer beim effektiven und effizienten Erreichen des Lernziels zu helfen. Eine Empfehlung kann beispielsweise durch die Auswahl des IP erfolgen, der die höchste Übereinstimmung mit dem bisherigen Nutzerpfad hat. IPMs und Lernpfade können auf Basis von standardisierten Beobachtungsdaten wie xAPI-Statements modelliert werden. Diese Arbeit entwirft und vergleicht verschiedene Modellierungen für IPMs in Graph-Datenstrukturen und klärt Anwendungen am Beispiel von Learning Management Systems. Außerdem wird eine Möglichkeit, die Progress Performance zu berechnen, vorgestellt. Diese soll nutzerspezifisch die Effizienz des Lernens bezüglich eines IP angeben und ein IP soll empfohlen werden, wenn dieser über alle möglichen IPs im IPM die Progress Performance maximiert. Durch die Verifikation mittels synthetischer Daten kann festgestellt werden, dass Algorithmen für kürzeste Pfade die konstruierten IPs finden. Allerdings existieren Beispiele mit Randfällen, in denen zentrale Lernobjekte fehlen.

Abstract

English Ideal Paths Models (IPMs) are reference models that model ideal sequences of learning objects of a learning system, the so called Ideal Paths (IPs), using graphs. Ideal means that the IPs model sequences of experts and following them results in an efficient and effective way of learning. A user's learning path, which is the sequence of learning objects performed by the user, can be adapted by individual recommendations to enable the user to learn more effective and efficient. Such a recommendation can be obtained by selecting the learning path that has the highest similarity to the user's learning path. IPMs and learning paths can be modeled based on standardized observation data such as xAPI-statements. This work creates and compares different modelings of IPMs in graph data structures and shows a use case in Learning Management Systems. Furthermore a method to measure the Progress Performance is explained. The Progress Performance should measure how efficient a user learns on an IP and the IP that maximizes the Progress Performance can be used as a recommendation. A verification using synthetic data showed that shortest paths algorithms can find the constructed IPs. There exist examples for edge cases where the central learning objects are missing.

1 Einleitung

Seit einigen Jahren und insbesondere durch die CoVid-19-Pandemie wächst der Bedarf an digitalen Lehr- und Lernangeboten (Raza u. a. 2021). Die meistgenutzten digitalen Lernplattformen wie beispielsweise Moodle stellen Systeme zum Management und Bereitstellung von Lerninhalten in verschiedensten Formen sowie Austauschformate wie Foren zur Verfügung (Cavus 2015). Nicht im Fokus der meisten Anwender steht dabei, dass Nutzern oder Nutzergruppen angepasste Lerninhalte vorgeschlagen werden, stattdessen werden die Lernplattformen häufig zur Verteilung statischer Daten genutzt (Green u. a. 2020). Im ähnlichen Feld der Lernspiele (engl. Serious Games) ist es ebenso Gegenstand aktueller Forschung, wie nutzerspezifische Hilfestellungen berechnet und umgesetzt werden können (Streicher, Busch u. a. 2021).

Adaptivität ist die Möglichkeit eines Lernsystems sich an die Bedürfnisse des Lerners anzupassen (Despotović-Zrakić u. a. 2012), beispielsweise durch Änderung des Schwierigkeitsgrades der Aufgaben oder durch Hinweise und Empfehlungen von Lerninhalten. Eine Möglichkeit, um Adaptivität durch Empfehlungen zu erreichen, sind sogenannte Ideal Path Models (IPMs), die Ideal Paths (IPs), die Referenz- oder Sollpfade durch ein Lernsystem modellieren. Ein Lernpfad ist eine Sequenz an Lernobjekten, die ein Nutzer zum Erreichen eines Lernziels durchläuft. Ein Ideal Path (IP) kann man daher als Lernpfad eines Experten interpretieren. Lernpfade sind also feste Strukturen, wogegen adaptive Lernpfade um das Konzept für adaptive Empfehlungen zur Anpassung der zukünftigen Lerneinheiten an die Bedürfnisse des Nutzers erweitert sind.

Sowohl Lernpfade als auch Ideal Paths können als Pfade in Graphen dargestellt werden und dadurch mit Mitteln der Graph Analytics verglichen werden.

Um zu entscheiden, ob ein Nutzer überhaupt auf Hilfe angewiesen ist, benötigt man eine Metrik, wie effektiv der Nutzer gerade lernt. Für einen Nutzer, der bereits auf einem IP lernt, sind keine Maßnahmen zum effektiveren Lernen des Nutzers notwendig. Allgemeiner soll mit der Progress Performance ein Wert gemessen werden können, wie effektiv ein Nutzer lernt, indem Übereinstimmungen von seinem Pfad mit den IPs aus dem Referenzmodell (Ideal Path Model (IPM)) berechnet wird. Der IP, bezüglich dessen der Nutzer die beste Performance hat, lässt sich dann auch als Empfehlung für eine adaptive Anpassung mit diesem Referenzmodell betrachten.

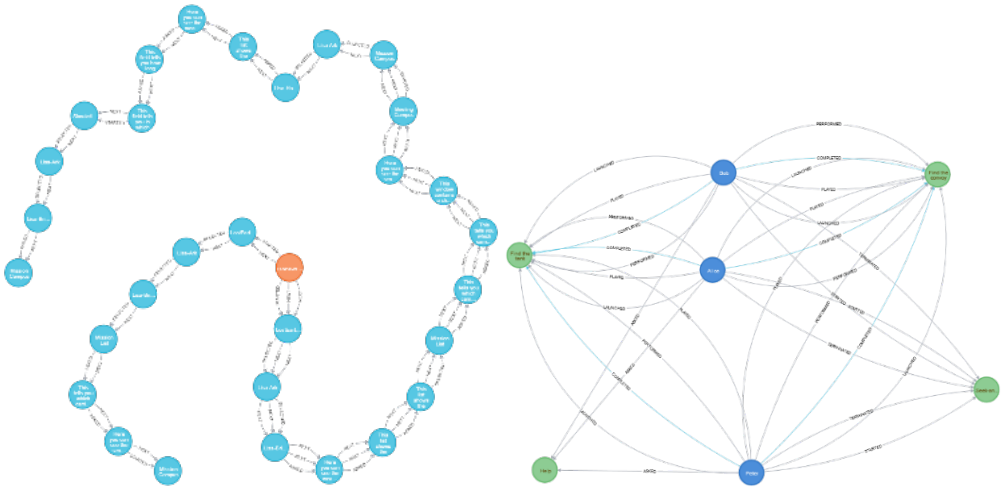


Abbildung 1.1: Erster Ansatz zur Modellierung (Streicher, Schönbein u. a. 2021)

1.1 Problemstellung

Für eine individuelle Empfehlung, was ein Nutzer als nächstes Lernen soll, benötigt man Informationen, wie der Nutzer lernt. Dazu können Beobachtungdaten zum bisherigen Lernen des Nutzers verwendet werden. Mit Graph Analytics sollen IPs (Streicher, Schönbein u. a. 2021) auf der Menge aller Nutzerdaten gefunden werden. Außerdem soll auf der Basis der IPs eine Metrik für die Progress Performance entwickelt werden. Dadurch kann man aus einer Menge möglicher IPs (Streicher, Schönbein u. a. 2021) für eine individuelle Empfehlung den zum Nutzer passendsten IP auswählen. Dazu benötigt es eine geeignete Modellierung der Beobachtungsdaten in Graphen. Möglichkeiten für Modellierungen sollen in der Arbeit gefunden und von theoretischer Seite verglichen werden.

1.2 Lösungsansatz

Der erste Schritt zur Anwendung von Graph Analytics ist die Modellierung des Problems als Graph. In diesem Fall werden also die Beobachtungsdaten der Nutzer als Lernpfade in einem Graph modelliert, eine erste Idee dazu ist in Abbildung 1.1 dargestellt. Außerdem werden die IPs modelliert, die ein Ideales, also möglichst effizientes Lernen abbilden. Die IPs (Streicher, Schönbein u. a. 2021) stellen kürzeste Pfade dar, die alle für einen spezifischen Lernertyp nötigen Lerneinheiten enthalten.

Zur einfachen Integration soll ein Microservice mit einem Eingabe-Verarbeitung-Ausgabe-Schema entwickelt werden.

- Die Eingabe besteht aus Benutzermodellen sowie Interaktionsdaten.
- In der Verarbeitung wird das Nutzerverhalten anhand der Modelle sowie aktueller Interaktionsdaten klassifiziert und ein passender IP sowie ein Performancewert ermittelt.
- Die Ausgabe besteht aus einer Lernempfehlung und dem Performancewert, sodass ein virtueller Agent daraus Hinweise erstellen und dem Nutzer anzeigen lassen kann.

1.3 Zielsetzung

Die Ziele dieser Arbeit sind im theoretischeren Bereich des Projektes.

- Modellierung von Nutzungspfaden und IPs.

  Es werden verschiedene Möglichkeiten zur Modellierung von Pfaden, also Nutzungs- und Lernpfade sowie IPs, aufgezeigt und verglichen. Dabei wird die Erfahrung des Nutzers in Form von historischen Interaktionsdaten berücksichtigt, um entsprechend des Niveaus Empfehlungen auszusprechen. Außerdem soll die Möglichkeit, datengetrieben mittels kürzester Pfade die IPs zu finden, untersucht werden.

- Numerische Performance-Angabe

  Um zu bewerten, wie zielführend ein Lernpfad eines Nutzers ist, kann dieser mit IPs verglichen werden. Daraus wird mithilfe einer Funktion $\text{Perf}^P : \mathcal{P}_U \times 2^{\mathcal{P}_I} \rightarrow [0,1]$ für die Menge $\mathcal{P}_U$ der Nutzerpfade und $\mathcal{P}_I$ der IPs ein numerischer Wert für die sogenannte Progress Performance errechnet. Diese kann um die Ausgabe eines IP mit dieser Progress Performance erweitert werden.

1.4 Projektumfeld

Diese Arbeit findet im Rahmen des BMBF INVITE TRIPLEADAPT (*Innovationswettbewerb INVITE - BMBF* 2022) statt, was ein Projekt für den Innovationswettbewerb des Bundesministeriums für Bildung und Forschung zur Entwicklung von neuen Technologien für Lernumgebungen für berufliche Weiterbildung ist. Ein Ziel des Projekts ist es, Methoden zur Adaption von Lernpfaden zu erforschen, die als Grundlage im digitalen Drilling (*Innovationswettbewerb INVITE - BMBF* 2022) dienen. Der digitale Drilling ist die Erweiterung des digitalen Zwillings um die Ebene der menschlichen Wahrnehmung eines realen Objektes. Damit ist er eine Repräsentation des kognitiven Wissensvermittlungsraums. Lernpfade sind Sequenzen von historischen und

empfohlenen Interaktionen und adaptive Lernpfade sollen flexibel an die Bedürfnisse des Nutzers angepasst werden mit dem Ziel, das Lernverhalten des Nutzers effizienter und effektiver zu gestalten (Streicher, Schönbein u. a. 2021). Aus vorgegebenen Lernpfaden können Lernempfehlungssysteme geeignete nächste Lerninhalte für einen Nutzer ermitteln. Personalisierte Lernpfade werden typischerweise durch didaktische Vorgaben, inhaltliche Ähnlichkeiten zwischen Lernobjekten, dem erfolgreichen Lernverhalten anderer und durch andere Lernobjekte erlangtes Wissen sowie bestehendes Vorwissen beeinflusst. Ein Graph-basierter Ansatz zur Modellierung erlaubt die Analyse von adaptiven Lernpfaden mit aus diesem Umfeld bekannten Methoden.

Ein weiteres Entwicklungsziel des Projekts ist es, übertragbare Modellierungskonzepte für Referenzmodelle, sogenannte IPMs zu entwickeln. Auf dieser Basis können Adaptivitätsalgorithmen den Bedarf und die Art der benötigten Hilfe berechnen. Die erzeugten Lernpfade werden mithilfe eines auf Lernempfehlungen spezialisierten Frameworks evaluiert.

Die dynamische und datengetriebene Erzeugung von IPs ist aktuell auch Gegenstand weiterer Forschungsarbeiten und verwendet Techniken der künstlichen Intelligenz wie Data Mining, Graph Analytics und automatische Regelsysteme mit Machine Learning Klassifikatoren sowie Messverfahren zum Nutzungsverhalten.

1.5 Gliederung

Diese Arbeit beginnt in der Einleitung mit der Motivation und Problemstellung sowie einer kurzen Beschreibung des Lösungsansatzes. Zur Motivation wird außerdem das Projektumfeld geklärt.

Im Grundlagenkapitel werden dann einige verwendete Begriffe und Konzepte aus früheren Arbeiten eingeführt. Darunter befinden sich insbesondere die im Lösungsansatz verwendeten Konzepte wie Graphen, Graph Analytics, IPMs, Adaptivität sowie Learning Analytics genauso wie die im Prototyp verwendeten Technologien wie experience API (xAPI) und die Graphdatenbank neo4j.

Das Kapitel State-of-the-Art stellt weitere Arbeiten in dem Themenfeld und deren Ergebnisse vor. Anschließend werden im Kapitel Lernpfadmodellierung mittels Graphen einige Ansätze zu IPMs sowie Modellierungen von Lernpfaden erörtert und verglichen.

Für eine dieser Modellierungen wird im Kapitel Algorithmische Analyse der Lernpfadmodelle eine Methode zur Berechnung der Progress Performance eingeführt, die einen numerischen Wert dafür angibt, wie zielführend ein Nutzer gerade lernt.

Im nächsten Schritt werden im Kapitel Anwendung & Verifikation die Modelle in einem Prototypen implementiert. Außerdem werden die Grenzen der gewählten Methoden mit syn-

thetischen Beispieldaten gezeigt. Weiterhin werden in einer Diskussion die Ergebnisse in das Projektumfeld eingeordnet und Forschungsfragen abgegrenzt, die nicht Teil der Arbeit sind und offen bleiben.

In einem Fazit und Ausblick werden die Ergebnisse der Arbeit zusammengefasst und bewertet und weiterhin offene Forschungsfragen, die nicht Ziel dieser Arbeit waren oder durch die Ergebnisse der Arbeit aufkommen, genannt.

2 Grundlagen

2.1 Graphen

Graphen sind ein mathematisches Modell aus Knoten und Kanten, die unter anderem in der Informatik für Netzwerkprobleme Anwendung finden (Hodler 2019).

Kombinatorisch definiert bestehen Graphen $G = (V,E)$ aus einer Knotenmenge V (engl. vertices) und einer Kantenmenge E (engl. edges). Häufig werden wie in Abbildung 2.1 Knoten als Punkte und Kanten als Linien zwischen den Punkten dargestellt (Diestel 2017). Keine Graphen im Sinne dieser Definition sind Diagramme, wie in Abbildung 2.1 rechts dargestellt (Hodler 2019).

Vereinfachend schreibt man $|G|$ für die Knotenzahl $|V|$ und $||G||$ für die Kantenzahl $|E|$ eines Graphen $G = (V,E)$.

Einfache Graphen erlauben zwischen jedem Paar von Knoten höchstens eine Kante, also $E \subset V \times V$. In Multigraphen hingegen können mehrere Kanten zwischen einem Knotenpaar liegen, in Pseudographen müssen die beiden Knoten sich nicht unterscheiden. Beispiele sind in Abbildung 2.2 dargestellt.

Ein Teilgraph (engl. subgraph) $H \subseteq G$ ist ein Graph H mit einer Teilmenge $V(H) \subseteq V(G)$ der Knoten aus G als Knotenmenge und einer Teilmenge $E(H) \subseteq E(G)$ der Kanten in G als Kantenmenge (Diestel 2017). In Abbildung 2.3 ist demnach beispielsweise der rechte Graph ein Teilgraph des linken Graphen.

In dieser Arbeit werden insbesondere Pfade untersucht. Pfade als Graphen betrachtet haben eine Knotenmenge $V = \{v_0, \ldots, v_n\}$ und die Kantenmenge $E = \{v_0v1, \ldots, v_{n-1}v_n\}$, also jeder Knoten außer dem ersten und letzten hat genau zwei Kanten zu einem Vorgänger und einem Nachfolger (Diestel 2017). Ein Beispiel mit gerichteten und ungerichteten Pfaden ist in Abbildung 2.4.

Ein Walk bezeichnet eine alternierende Sequenz an Knoten und Kanten in einem Graph G, wobei eine Knoten v und die im Walk nachfolgende kante vw sowie eine Kante xy und der nachfolgende Knoten y inzident sein müssen (Diestel 2017). In Abbildung 2.5 ist beispielsweise $B, BC, C, CD, D, DC, C, CA, A$ ein Walk im ungerichteten Graphen.

Ein Kreis im Sinne der Graphentheorie bezeichnet einen Pfad mit gleichem Start- und

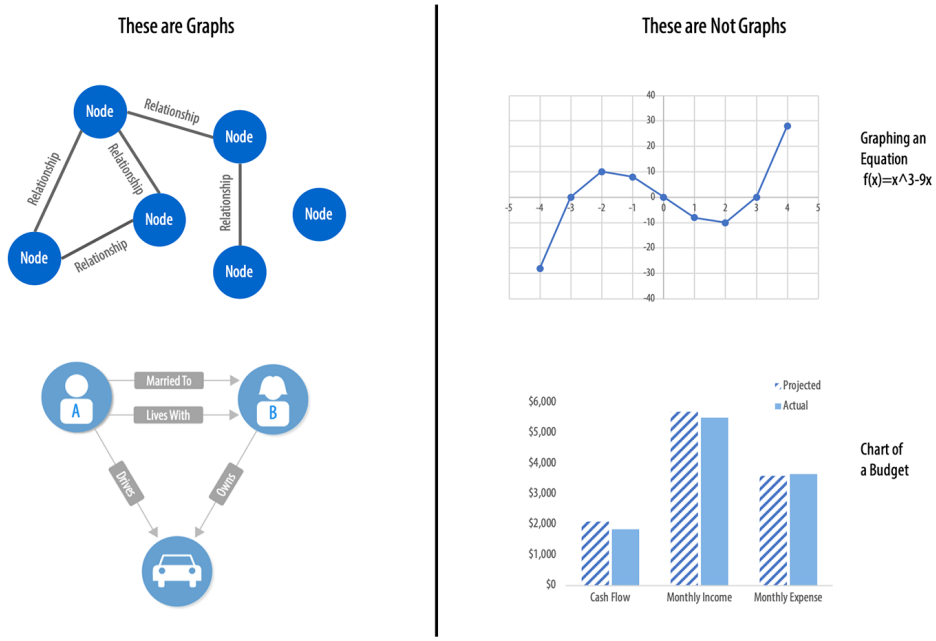


Abbildung 2.1: Übliche Darstellung von Graphen. Links: Knoten sind Kreise, Kanten Linien dazwischen, Rechts: keine Graphen im Sinne der Definition in dieser Arbeit (Hodler 2019).

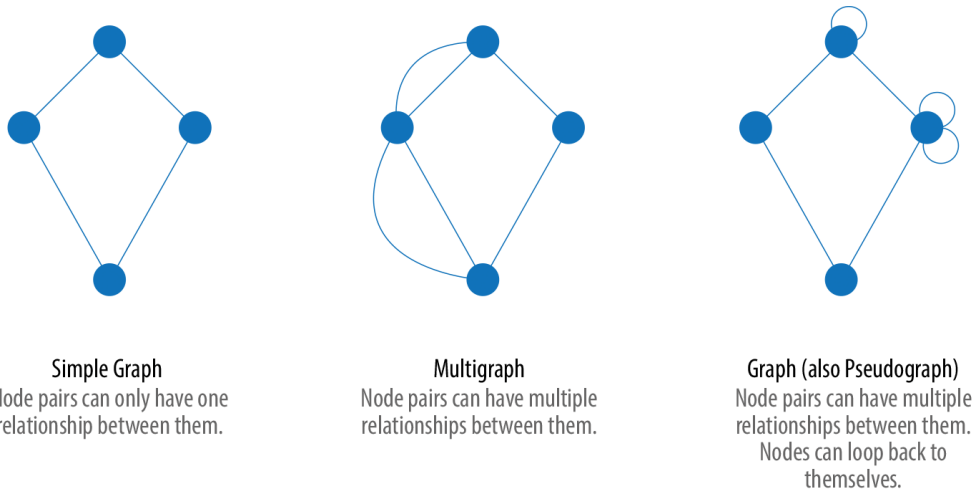


Abbildung 2.2: Links: In einfachen Graphen kann jedes Knotenpaar nur eine Richtung haben. Mitte: In Multigrahpen können Knotenpaare durch mehrere Kanten verbunden sein. Rechts: In Pseudographen können Knoten auch Kanten zu sich selbst haben (Hodler 2019).

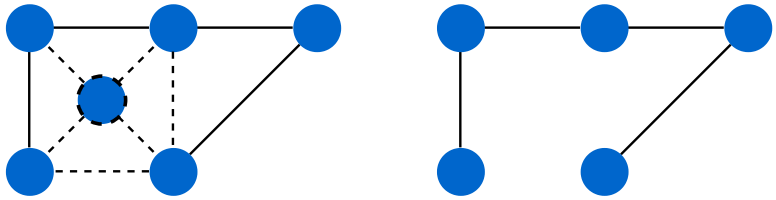


Abbildung 2.3: Beispiel für Teilgraphen, rechter Graph ist isomorph zu allen nicht gestrichelt gezeichneten Objekten im linken Graph. Dadurch ist der rechte Graph ein Teilgraph des Linken.

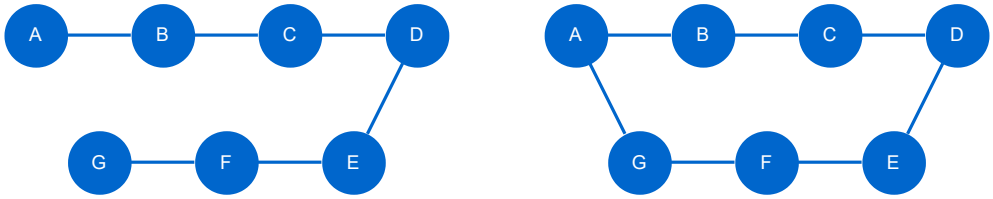


Abbildung 2.4: Links: Ein Pfad entlang der Knoten $A - G$. Rechts: Ein Kreis mit 6 Knoten

Zielknoten, also $V = \{v_0, \ldots, v_n = v_0\}$ und die Kantenmenge $E = \{v_0v_1, \ldots, v_{n-1}v_0\}$ (Diestel 2017), wie in Abbildung 2.4.

Weiterhin kann man sich mit Eigenschaften dieser Graphen auseinandersetzten. Wenn bei einer Kante zwischen Start- und Zielknoten unterscheiden werden soll, bezeichnet man dies einen gerichteten Graph, wie in Abbildung 2.5 rechts. In ungerichteten Graphen muss man beachten, dass Start- und Zielknoten keine festgelegte Reihenfolge haben, also auch vertauscht sein könnten. In Graphen, in denen bei Nichtbeachtung der Kantenrichtung zwischen jedem Knotenpaar ein Pfad existiert, nennt man zusammenhängend (Hodler 2019).

In Graphen, in denen bei Beachtung der Kantenrichtung ein Kreis, also ein Weg mit gleichem Start und Zielknoten, existiert, nennt man zyklisch, sonst azyklisch (Hodler 2019).

In gewichteten Graphen haben Knoten oder Kanten eine oder mehrere Gewichtsfunktionen $f : S \to K, S \subseteq E \cup V$ (Hodler 2019). Diese werden oft durch Zahlen an der Kante dargestellt, wie in Abbildung 2.6 zu sehen.

2.2 Graph Analytics

Graph Analytics ist ein Sammelbegriff für verschiedene Möglichkeiten zur Analyse und Beobachtungen von Graphen und deren Eigenschaften und Strukturen. Beispielsweise kann man grundlegende statistische Verfahren auf Graphen anwenden, eine Visualisierung eines Graphen direkt Beobachten oder Algorithmen auf die Daten des Graphen anwenden (Hodler 2019).

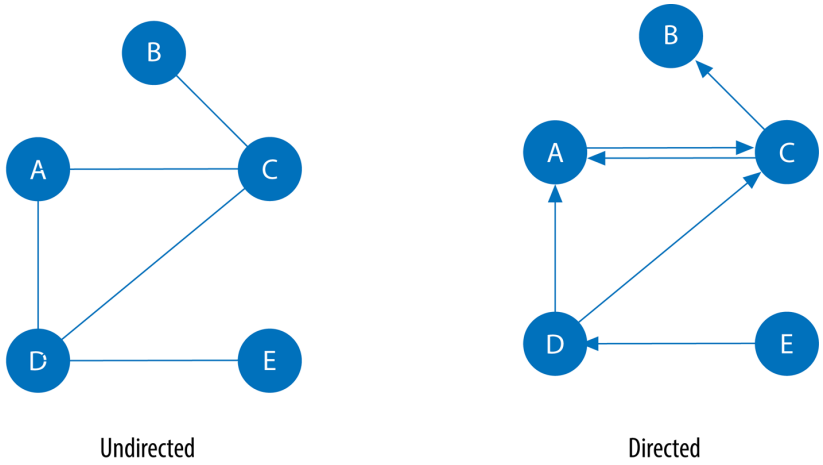


Abbildung 2.5: Links: in ungerichteten Graphen ist keine Richtung definiert, die Kanten werden als bidirektional angesehen. Rechts: In gerichteten Graphen haben die Kanten eine Richtung, dargestellt durch einen Pfeil (Hodler 2019).

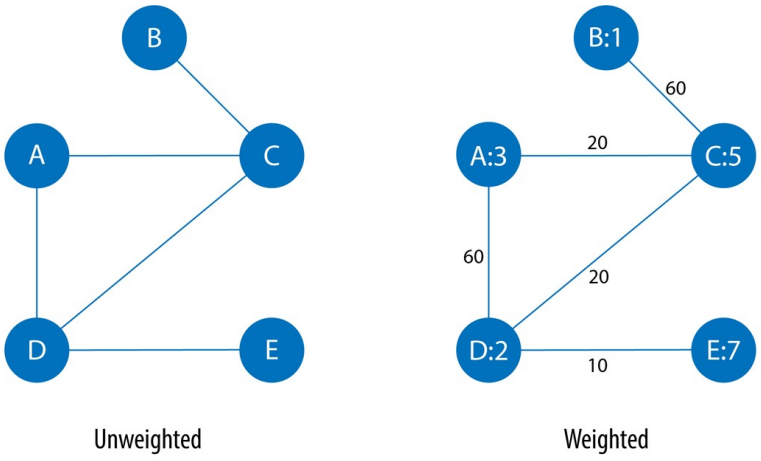


Abbildung 2.6: Links: ungewichteter Graph, falls ein Gewicht benötigt wird, wird dieses Häufig auf 1 gesetzt. Rechts: gewichteter Graph, an den Kanten sind Zahlen, die das Gewicht darstellen (Hodler 2019).

Da Graphen bereits in vielen Anwendungen genutzt werden, existieren einige Graphalgorithmen. Für diese Arbeit relevant ist es vor allem kürzeste Wege in Graphen und zentrale Knoten zu finden.

2.2.1 Kürzeste-Wege-Algorithmen

Das Kürzeste-Wege Problem besteht darin, zwischen gegebenen Start- und Zielknoten den kürzesten Pfad als Teilgraph zu finden (Hodler 2019). Die Breitensuche ist die Basis für viele Algorithmen, die dieses Problem lösen. Dabei werden ausgehend vom Startknoten im ersten Schritt die direkten Nachbarn besucht, dann die neuen Nachbarn aller im Schritt vorher besuchten Knoten. Dieser Algorithmus kann in ungewichteten Graphen den kürzesten Weg finden. Um einen kürzesten Pfad in einem gewichteten Graph zu finden, hat Edsger Dijkstra eine modifizierte Breitensuche vorgestellt, in dem unbesuchte Knoten mit geringstem Gewicht priorisiert werden, unabhängig davon, wie viele Knoten dazwischen liegen (Hodler 2019). Ein weiterer Algorithmus ist Yen's k-Shortest-Path Algorithmus (Yen 1971), der die k kürzesten Pfade, also den kürzesten Pfad und bis zu $k-1$ nächst-kürzeren Abweichungen davon in einem Graphen finden kann, sowie A*, der mittels einer Heuristik den Zeitaufwand minimieren kann (Hodler 2019).

2.2.2 Algorithmen zur Bestimmung der Zentralität

Zentralitätsalgorithmen werden verwendet, um die Rollen einzelner Knoten und deren Einfluss in einem Netzwerk zu finden (Hodler 2019).

Der Knotengrad ist eine Metrik, um den direkten Einfluss auf die Nachbarn der Knoten zu messen. Mit dem Algorithmus Closeness Centrality können Knoten gefunden werden, die den kürzesten Weg zu allen anderen haben (Hodler 2019). Mittels Betweenness Centrality können Knoten gefunden werden, die am meisten Kontrolle über den Fluss im Netzwerk haben, da die meisten kürzesten Pfade durch diese verlaufen (Hodler 2019). PageRank sucht ähnlich zum Knotengrad Knoten mit hohem Einfluss, gewichtet aber die Nachbarn mit deren Einfluss auf das Netzwerk (Hodler 2019).

Die Ziele dieser Algorithmen sind nochmals in Abbildung 2.7 verdeutlicht.

2.3 Graphdatenbank Neo4j

Neo4j ist eine NoSQL Graphdatenbank, das heißt, es existiert kein fixes Relationsschema, stattdessen werden Informationen wie in Graphen durch Knoten (engl. node) und Relationen zwischen Knoten gespeichert (Hodler 2019). Zusatzinformationen können durch Eigenschaften

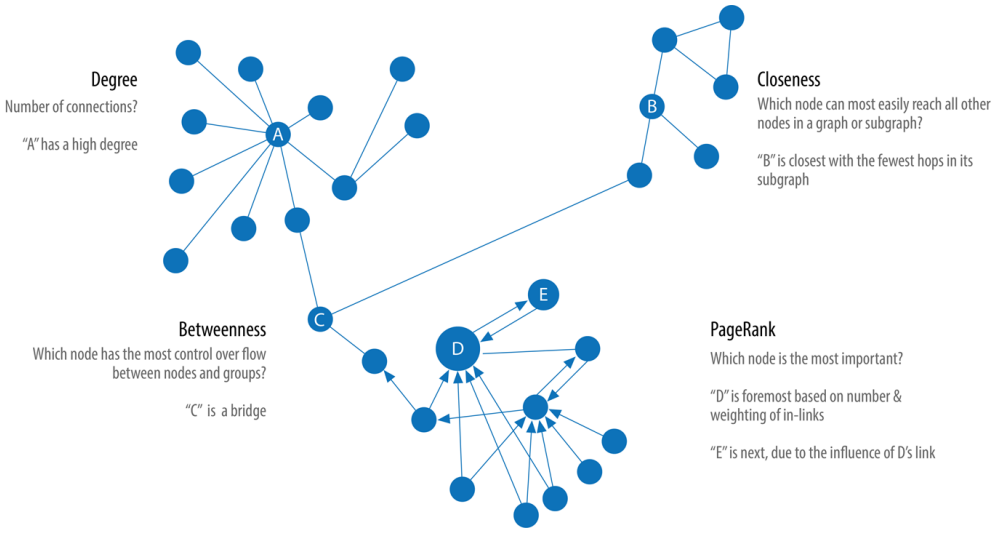


Abbildung 2.7: Eine Visualisierung der Verschiedenen Arten von Zentralität. *A* ist ein Knoten mit höchstem Knotengrad. *B* ist eine Brücke, daher hat sie die höchste Betweenness Centrality. *C* ist im Teilgraph oben rechts der Knoten mit dem kürzesten Weg zu allen anderen, daher hat er die größte Betweenness Centrality. *D* hat gewichtet mit dem Einfluss seiner Nachbarn den höchsten Einfluss nach dem PageRank Algorithmus.

von Knoten und Kanten gespeichert werden, dies ist analog zu einem gewichteten Graph. In neo4j kann man auch mehrere Eigenschaften am gleichen Knoten oder an der gleichen Kante speichern und es werden Zahlen und Strings unterstützt. Graphalgorithmen werden durch eine zusätzliche Bibliothek direkt in der für Graphen optimierten Datenbankabfragesprache *CYPHER* zur Verfügung gestellt (Hodler 2019).

2.4 Learning Management Systems

Learning Management Systems (LMSs) stellen eine virtuelle Plattform zum Lernen bereit, in der Nutzer zusätzlich zum reinen Abrufen von statisch bereitgestellten, multimodalen Lerninhalten angemeldete Nutzer diese individuell managen können. Außerdem stehen weitere Funktionen zum Testen, Tracken und Kommunikation mit anderen Nutzern (Cavus 2015) bereit.

Moodle (Modular Object Oriented Dynamic Learning Environment) ist ein weit verbreitetes LMS (Cavus 2015). Es erlaubt verschiedene Lernaktivitäten wie Lektionen, interaktive Lerneinheiten, Prüfungen, Aufgaben und Umfragen, die in Kursen gegliedert sind. Weiterhin bietet es Austauschforen, individuelles Verfolgen des Lernfortschritts und die Möglichkeit zum synchronen Unterricht an (Cavus 2015).

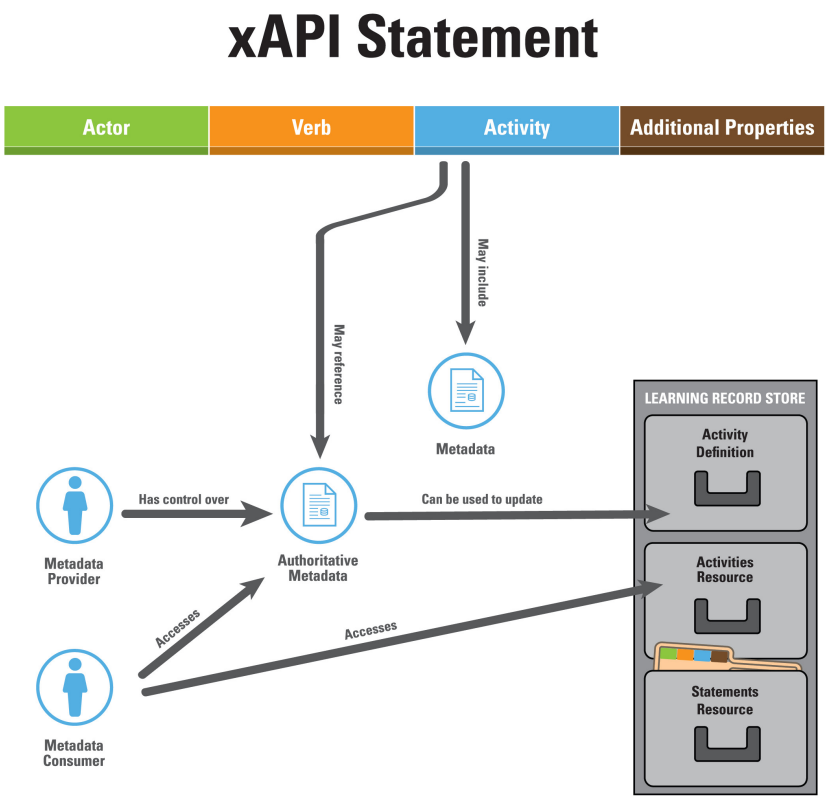


Abbildung 2.8: Schema von xAPI Statements und deren Verwendung (*xAPI-Spec* 2021). Ein Statement besteht aus einem Actor, einem Verb und einem Object, beispielsweise einer Activity, und weiteren optionalen Eigenschaften. Die Statements können dann in einem Learning Record Store gesammelt werden und von autorisierten Anwendungen abgefragt werden.

2.5 Interoperabilität

Um Lernerfahrungen möglichst unabhängig vom Lernsystem und von verschiedenen Lernsystemen zu sammeln, können Standards zur Interoperabilität genutzt werden. Für Beobachtungsdaten ist dies Insbesondere xAPI, für Daten zu Kursstrukturen ist dies Common Cartridge.

2.5.1 Experience API

Die xAPI (*xAPI-Spec* 2021) ist wie SCORM eine technische Spezifikation zum Übertragen und Speichern von Lernerfahrungen. Sie hat sich zum Ziel gesetzt, einfach verständlich zu sein, Vergleichbarkeit von Lernerfahrungen zu gewährleisten und Lernerfahrungen aus allen

```
{
  "actor": {
    "name": "Sally Glider",
    "mbox": "mailto:sally@example.com"
  },
  "verb": {
    "id": "http://adlnet.gov/expapi/verbs/experienced",
    "display": { "en-US": "experienced" }
  },
  "object": {
    "id": "http://example.com/activities/solo-hang-gliding",
    "definition": {
      "name": { "en-US": "Solo Hang Gliding" }
    }
  }
}
```

Listing 2.1: Ein Beispiel für ein xAPI Statement, der Actor ”Sally Glider” hat die Aktivität ”Solo Hang Gliding” ”erlebt” (*xAPI-Spec* 2021).

Kontexten zu erlauben.

Das Format baut auf den JSON-basierten *W3C Activity Streams* (*Activity Streams 2.0* 2017) auf. Die Statements bilden eine Tripelstruktur aus *Actor*, *Verb* und *Object*, die analog zu einem Satz aus Subjekt, Prädikat und Objekt beschreiben, wer etwas tut, was diese Person tut und welche Aktion ausgeführt wird. Ein Statement kann also einfach in einen Satz umgewandelt werden, beispielsweise *Bob schreibt einen Text*. Das Objekt kann dabei wieder ein Statement sein, z.B. *Alice beobachtet, wie Bob einen Text schreibt*. Außerdem sind optionale Felder für ein Ergebnis (Result), den Kontext und weitere Metadaten vorgesehen. Ein Beispiel ist in Listing 2.1 zu finden.

Learning Records bezeichnen die beschriebenen xAPI-Statements mit weiteren Metadaten. Ein Learning Record Provider ist ein als vertrauenswürdig eingestufter Tracker, der als Komponente in Lernsoftware, z.B. moodle (*Experience API (xAPI) - MoodleDocs* 2022) oder auch Lernspielen integriert wird. Die Learning Records können dann in einem Learning Record Store (LRS) gespeichert und abgerufen werden (*xAPI-Spec* 2021).

2.5.2 Common Cartridge

Common Cartride ist eine Erweiterung des SCORM Standards und dient der Organisation und Verteilung von digitalen Lerninhalten durch sogenannte Learning Objects (Gonzalez-Barbone u. a. 2010). Dies wird durch ein Format zum Austausch, Veröffentlichung, Verteilung und Suche

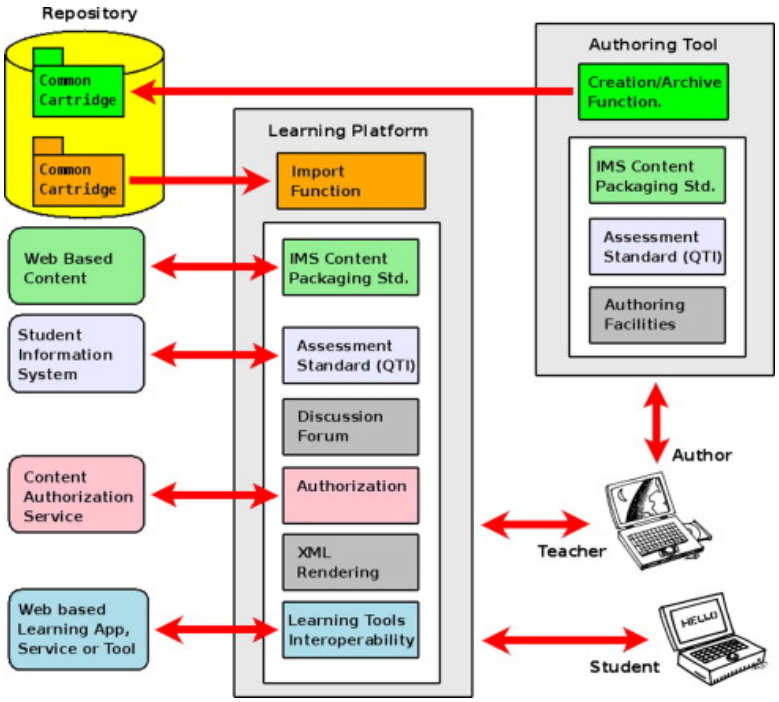


Abbildung 2.9: Beispielhafte Nutzung von Common Cartrige mit Autorenwerkzeug, Repository und LMS. Die Lerninhalte können dabei auf externe Inhalte verweisen (Gonzalez-Barbone u. a. 2010).

von Lerninhalten erreicht (Gonzalez-Barbone u. a. 2010). Common Cartride erlaubt weitere Learning Objects wie Tests, Referenzen auf Webinhalte, Interaktionen mit externen Tools und Austauschforen (Gonzalez-Barbone u. a. 2010). Das Ziel davon ist, dass Lerninhalte besser verfügbar sind, da sie einfach zwischen verschiedenen LMS-Anbietern austauschbar sind

Abbildung 2.9 zeigt, wie Common Cartride von einem Autorenwerkzeug über einen Speicher in ein LMS importiert werden kann und stellt auch Interaktionen von verschiedenen Lerninhalten mit externen Inhalten dar (Gonzalez-Barbone u. a. 2010).

2.6 Adaptivität

Im Umfeld von Intelligenten Tutoring Systemen (ITS) bezeichnet Adaptivität die Möglichkeit, die Lerninhalte den Bedürfnissen des Lerners anzupassen. Ein ITS soll dabei insbesondere in der Lage sein, Nutzeraktivitäten zu beobachten, diese zu interpretieren, um daraus Anforderungen und Präferenzen des Nutzers zu erkennen und darauf basierend den Lernprozess für den Nutzer zu vereinfachen (Despotović-Zrakić u. a. 2012). Als Beispiele werden von Despotović-Zrakić

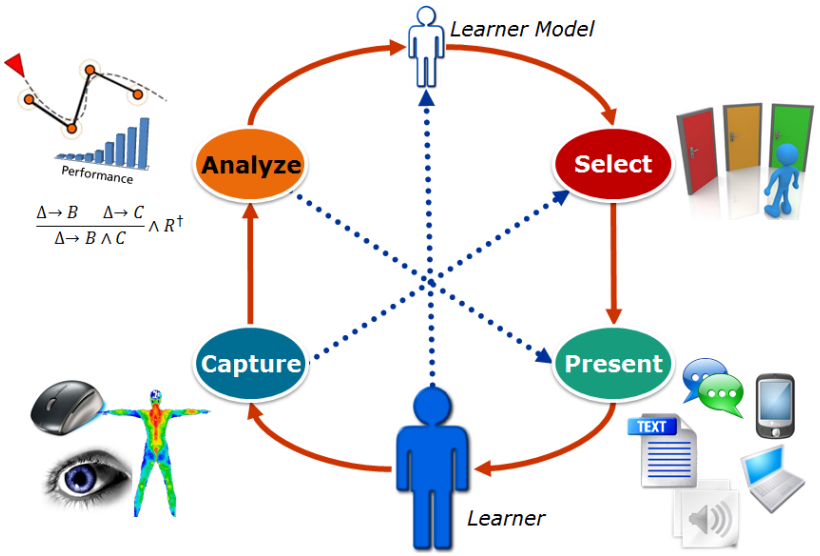


Abbildung 2.10: Vier-Phasen Adaptivitätszyklus (Streicher und Smeddinck 2016, basierend auf V. Shute u. a. 2012

u. a. 2012 adaptive Quizze, intelligente Lösungsanalysen, adaptives Monitoring von ganzen Klassen und adaptive kollaborative Systeme genannt, die ihre jeweiligen Aufgaben effizienter ausführen. (Sottilare u. a. 2012) unterscheidet dabei zwischen zwei wesentlichen Ausprägungen, der Makro- und der Mikroadaptivität.

Makroadaptivität zeichnet sich in Sottilare u. a. 2012 dadurch aus, dass Informationen über den Lerner, die vor der ersten Interaktion mit dem Lernsystem verfügbar sind, verwendet werden, um dem Nutzer off-line, also im Voraus berechnete, angepasste Inhalte zu präsentieren.

Mikroadaptivität wird in (Sottilare u. a. 2012) dagegen als nahezu in Echtzeit vorgenommene Anpassungen der präsentierten Lerninhalte bezeichnet. Als Beispiel wird „Scaffolding" genannt, was eine mikroadaptive Strategie ist, die mit steigender Kompetenz des Lerners weniger Unterstützung bietet.

Valerie J. Shute u. a. 2012 hat den in 2.10 gezeigten Adaptivitätszyklus mit den vier Phasen Capture, Analysis, Select und Present vorgeschlagen. Um Adaptivität zu erreichen, muss man also erst beobachten, was der Nutzer tut und daraus analysieren, wie der Nutzer lernt. Damit kann man geeignete Anpassungen auswählen und in adäquater Form präsentieren.

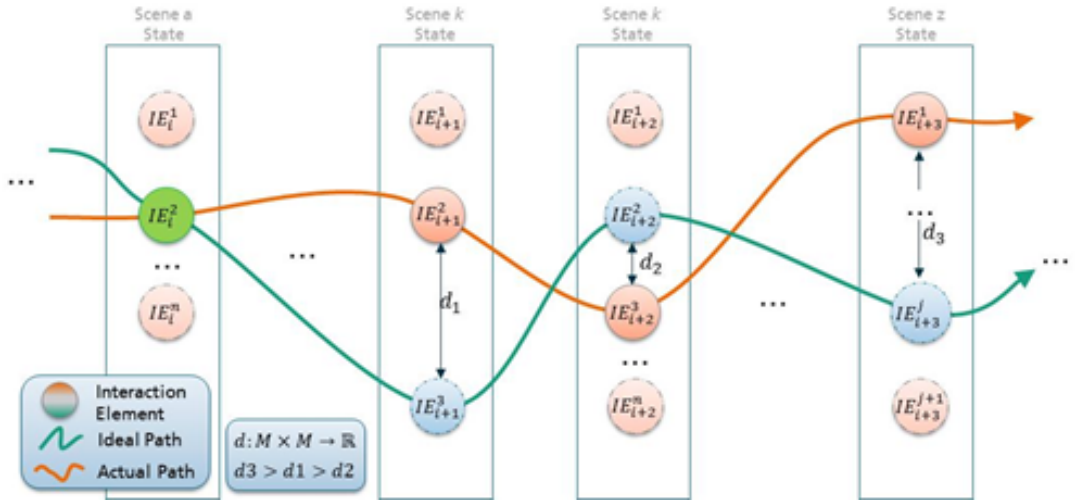


Abbildung 2.11: Beispiel für ein IPM mit Szenen, möglichen Interaktionen, einem IP und einem Lernpfad eines Nutzers (Streicher, Leidig u. a. 2018)

2.7 Ideal Paths Models (IPMs)

Eine Möglichkeit zum Erreichen von Adaptivität sind Ideal Path Models (IPMs) (Streicher, Schönbein u. a. 2021). IPMs modellieren alle nötigen Schritte zum Erreichen eines Lernziels, also eine zielführende Sequenz an Lernobjekten in einer virtuellen Umgebung (Streicher, Leidig u. a. 2018). Ein Ideal Path (IP) ist beispielsweise ein idealer Durchlauf eines Experten in einem Serious Game (Streicher, Leidig u. a. 2018).

Ein IPM besteht laut Streicher, Leidig u. a. 2018 aus mehreren Komponenten, die auch in Abbildung 2.11 gezeigt sind:

- **Szene** Die Szene ist ein Zustand des Lernsystems.
- **Interaktionen** In jeder Szene gibt es mögliche Interaktionen, in einem Serious Game können beispielsweise Interaktionen nur an einem bestimmten Ort durchgeführt werden.
- **IPs** Die Idealen Pfade sind effektivsten Durchläufe durch die Szenen und Interaktionen.
- **Lernpfad** Der tatsächliche Lernpfad eines Nutzers ist analog zu den IPs der Durchlauf durch die Szenen und Interaktionen, die ein Nutzer bisher gewählt hat. Für eine on-line-Analyse, also während der Nutzer Interaktionen durchführt, ist dieser nicht vollständig und insbesondere noch nicht am Lernziel.

Weiterhin wurde der Ideal Path Score (IPS) als Metrik mit Werten in $[-1,1]$ für einzelne Interaktionen vorgestellt. Dabei bedeutet ein IPS von 1, dass die letzte Interaktion die best-

mögliche war, 0 bedeutet, dass der Nutzer keinen Frotschritt zum Ziel gemacht hat und −1 bedeutet, dass er sich wieder weit vom Ziel entfernt (Streicher, Leidig u. a. 2018).

2.8 Lern-Performance

Loh u. a. 2014 verwendet die Performance als vom Nutzer erbrachte Leistung, die einen Anfänger von einem Experten unterscheiden kann. Der verwendete Ansatz vergleicht die Performance eines Nutzers mit einem Experten durch den Vergleich der durchgeführten Lernobjekte mittels der Jaccard Similarity:

$$JACC(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Dabei werden für die Mengen A,B Paare von aufeinander folgenden Aktivitäten genutzt, um auch die Reihenfolge der betrachteten Aktivitäten zu berücksichtigen. Die Idee dahinter ist, dass Nutzer besser sind, wenn er die Lernobjekte ähnlich wie ein Experte durchlaufen und schlechter, wenn sich zumindest die Reihenfolge der Lernobjekte oder die genutzten Lernobjekte unterscheiden. Weiterhin wurde gezeigt, dass die Performance nicht mit der benötigten Zeit korreliert (Loh u. a. 2014).

Ein weiterer Ansatz ist die Learning Factor Analysis. Hierbei wird ein Performance Score durch das Können (engl. Ability) α_i eines Lernenden i sowie die Schwierigkeit β_j einer Wissenskomponente (KC, engl. Knowledge Component) j und der durch γ_j gewichteten Addition der Lernhäufigkeit $n_{i,j}$ bestimmt (Pavlik Jr u. a. 2009). Damit ergibt sich die Formel für den Logitwert m und die Performance $p(m)$

$$m(i,j \in KCs,n) = \alpha_i + \sum_{j \in KCs} (\beta_j + \gamma_j n_{i,j})$$

$$p(m) = \frac{1}{1 + e^{-m}}$$

Was damit laut Pavlik Jr u. a. 2009 nicht berücksichtigt wird, ist die erbrachte Leistung (z.B. durch Prüfungen) der einzelnen KCs. Daher wird von Pavlik Jr u. a. 2009 die Performance Factor Analysis eingeführt, die sowohl Erfolge und Misserfolge in einzelnen KCs beachtet. Dazu werden die beobachteten Leistungen, die in der LFA mit $n_{i,j}$ erfasst wurden, in Erfolge $s_{i,j}$ (engl. success) und Misserfolge $f_{i,j}$ (engl. Failure) aufgeteilt. Zusätzlich gibt es für die Misserfolge eine eigene Gewichtung ρ_j analog zur Gewichtung für die Erfolge γ_j (Pavlik Jr u. a. 2009). Außerdem wurde die Konstante für das Können des Nutzers entfernt.

Insgesamt hat sich also nur die Berechnung des Logitwertes m laut Pavlik Jr u. a. 2009

geändert zu

$$m(i,j \in KCs,n) = \sum_{j \in KCs} (\beta_j + \gamma_j s_{i,j} + \rho_j f_{i,j})$$

Eine weitere Möglichkeit zur Performanceberechnung ist das Knowledge Tracing (KT) (Pavlik Jr u. a. 2009). Dabei werden für jede KC 4 Parameter für initiales Lernen, Lernrate, Rateparameter und der Slip Parameter, der die Wahrscheinlichkeit für versehentlich falsche Antworten angibt, verwendet. Dieses Modell hat den Vorteil, dass die Parameter verständlich sind (Pavlik Jr u. a. 2009).

3 State of the Art

In diesem Kapitel wird der Stand der Technik zu Learning Analytics allgemein sowie zu Arbeiten zum Thema Performance und Ideal Path Models (IPMs) zusammengefasst.

3.1 Learning Analytics

Eine weit akzeptierte Definition ist „Learning Analytics ist das Messen, Sammeln, Analysieren und Berichten von Daten über Lernende und ihre Kontexte, um das Lernen und die Lernumgebung zu verstehen und zu optimieren" (Mangaroska u. a. 2019). Bisherige Arbeiten haben gezeigt, dass Techniken aus Learning Analytics angewandt auf historische Daten aus Virtual Learning Environments effektiv die Performance der Studenten vorhersagen. Dabei konnte aus mit KI generierten Regelsets die Performance zu einer hohen Wahrscheinlichkeit vorhergesagt werden. Durch die Verwendung von Regelsets kann auch erklärt werden, wie eine Vorhersage zustande kommt (Alonso u. a. 2019).

Im Gegensatz zu klassischen Vorher-Nachher-Tests, bei denen nur der Wissensstand vor und nach einer Sequenz an Lernobjekten geprüft wird, kann durch Learning Analytics der gesamte Lernprozess beobachtet und damit in Echtzeit das Lernverhalten erkannt werden. Außerdem kann während des Lernprozesses eingegriffen werden und so adaptives Lernen z.B. auch in Serious Games ermöglicht werden. Die Verwendung der xAPI und eines standardisierten Kollektors hilft dabei den Aufwand bei der Implementierung neuer Lernumgebungen geringer zu halten (Alonso-Fernandez u. a. 2017)

3.2 Adaptivität durch Ideal Paths Models

Streicher, Schönbein u. a. 2021 beschreiben ein adaptives Assistenzsystem für ein Lernspiel (engl. Serious Game), das auf einer möglichen Modellierung von IPMs basiert. Die Datengrundlage dafür sind ebenfalls die xAPI-Statements mit der Tripel-Datenstruktur aus Actor, Verb, Object, die in einem Graphen modelliert werden. Die Autoren verwenden den von (Valerie J. Shute u. a. 2012) vorgestellten und in Abbildung 2.10 abgebildeten Adaptivitätszyklus *Capure*, *Analysis*, *Select*, *Present* und modelliert in der *Analysis* Phase ein IPM, aus dem ein IP in der *Select*

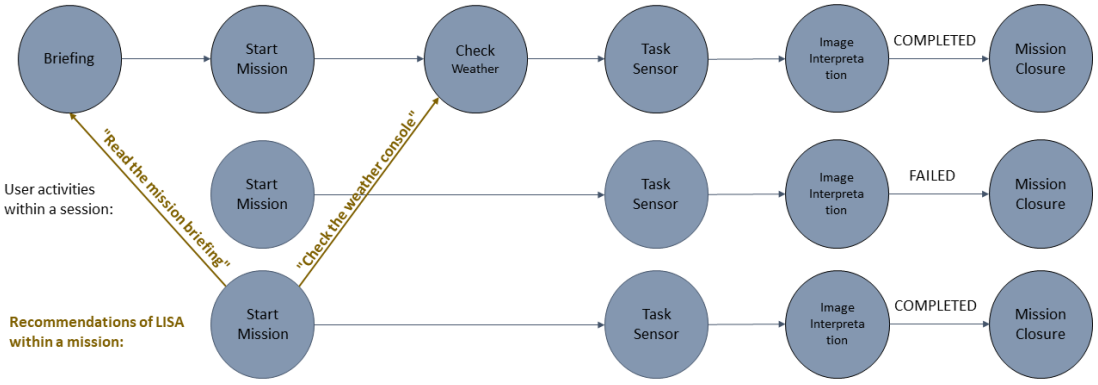


Abbildung 3.1: Modellierung von Nutzungspfaden und Empfehlungen in Grissinger 2020.

Phase ausgewählt wird. Experimentell nachgewiesen wurde dabei, dass die Flexibilität solcher datengetriebenen Modellierungen und die bestehenden Graphalgorithmen für die *Analysis* Phase vorteilhaft sind.

3.3 User-Modeling mit Graph-Analytics

Grissinger 2020 hat gezeigt, wie eine adaptive Berechnung von Hilfestellungen und deren Anzeige im Lernspiel „Lost Earth" möglich sind. Dazu wurde zunächst ein „xAPI-Tracker", also eine Softwarekomponente zum Sammeln von Beobachtungsdaten im xAPI-Format, für das Serious Game entwickelt. Weiter verwendet Grissinger 2020 eine auf Graphen basierte Modellierung von xAPI Daten (Abbildung 3.1), in der für jede Session jedes Nutzers ein Pfad existiert und wendet wie Streicher, Leidig u. a. 2018 darauf Yen's k-Shortest-Path Algorithmus an, um verschiedene IPs zu erhalten. Die visuelle Rückmeldung für Empfehlungen existiert bereits in „Lost Earth" und wurde dazu verwendet. In dieser Arbeit wird der Fokus auf den Vergleich verschiedener Modellierungen in Graphen gelegt.

3.4 Performance Analysen

Pavlik Jr u. a. 2009 hat die in Abschnitt 2.8 erklärte Methode „Performance Factor Analysis" (PFA) vorgestellt, um den Lernfortschritt von Lernenden in intelligenten Tutorensystemen zu bewerten. Dabei haben die Autoren im Gegensatz zur damit verglichenen „Learning Factor Analysis" (LFA) die Performance der Lernenden in die Bewertung von einzelnen Lernobjekten mit einbezogen. Die Berechnung erfolgt auf Grundlage von gesammelten Informationen zur Schwierigkeit eines Wissensbereiches sowie Sensitivität zu korrekten und inkorrekten

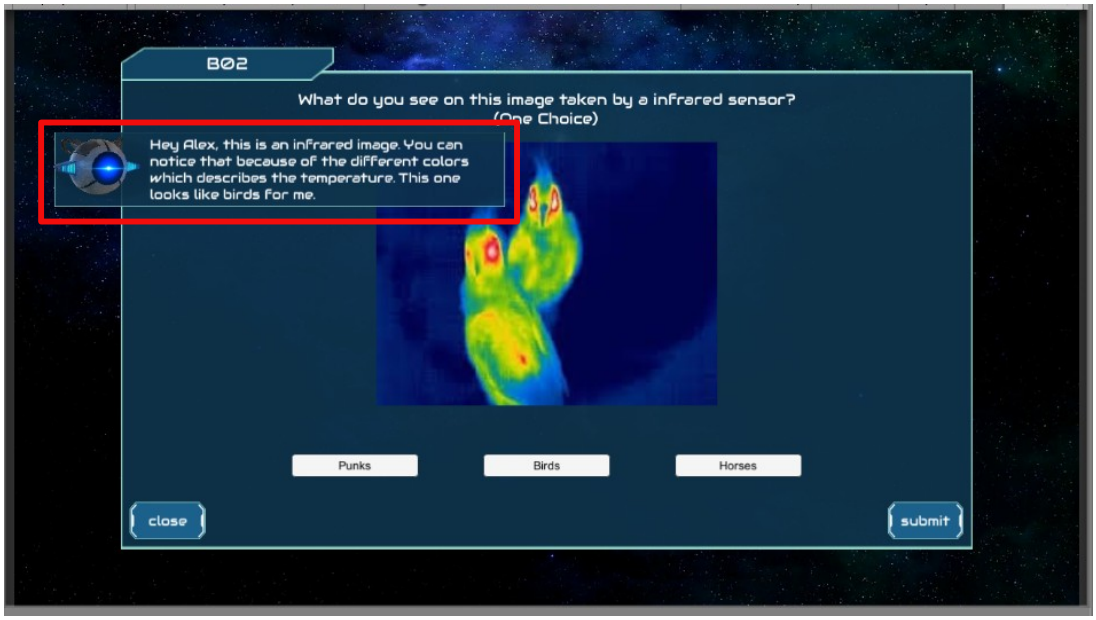


Abbildung 3.2: Ein Beispiel eines adaptiven Hinweises (markiert) im Serious Game „Lost Eart" (Pustovojtovskij 2021).

Antworten in diesem Wissensbereich.

Anhand von vier Datensätzen wurden die Parameter von PFA, LFA und Knowledge Tracing (KT) angepasst. Dabei haben LFA und PFA ein ähnlich gutes Ergebnis in der Vorhersage der Performance erreicht, wogegen KT aufgrund der Annahme, dass das Wissen der Lernenden Bernoulli-verteilt ist, etwas schlechter angepasste Parameter hat (Pavlik Jr u. a. 2009). Weiterhin hat laut Pavlik Jr u. a. 2009 PFA gegenüber KT den Vorteil, dass sich die Kosten des Lernobjekts, typischerweise die Dauer, vorhersagen lassen.

Pustovojtovskij 2021 hat PFA für ein Serious Game implementiert, um damit auch einzelnen Nutzern einen Performance-Score zuzuweisen. Außerdem hat er einen virtuellen Assistenten, der dem Nutzer bei sinkender Performance Hinweise geben kann (Abbildung 3.2). In einer Nutzerstudie wurde nachgewiesen, dass die Performance der Nutzer mit adaptiven Hinweisen besser ist (Pustovojtovskij 2021).

Diese Arbeit beschäftigt sich im Gegensatz dazu mit einer Möglichkeit, einen Performance-Score auf Basis von Graphen zu berechnen.

4 Lernpfadmodellierung mittels Graphen

Dieses Kapitel beschäftigt sich mit der Voraussetzung für eine Analyse, der Konzeption von IPMs aus den Beobachtungsdaten. Dabei werden mit den in der xAPI vorgegebenen Feldern verschiedene Modellierungen vorgestellt und verglichen. Zunächst wird in die unterschiedlichen Bedeutung von Pfaden betrachtet. Weiterhin werden die zu modellierenden Objekte in den Beobachtungsdaten, also den xAPI-Statements identifiziert und Möglichkeiten vorgestellt, wie diese in Graphen dargestellt werden können. Zum Schluss werden noch Eigenschaften und Strukturen der Graphen geklärt.

4.1 Unterscheidung von Pfaden

Ich will im IPM unterschiedliche Abstraktionsebenen verwenden. Das sind als erstes die ungefilterten Nutzungspfade, dann die Lernpfade und zuletzt die Ideal Paths (IPs). Die unterschiedlichen Pfade kann man dabei gleich modellieren, da sie sich vor allem in der Deutung unterscheiden, allesamt jedoch Interaktionssequezen darstellen sollen. Dabei wird für die eigentliche Modellierungen die Lernpfade wie in Streicher, Schönbein u. a. 2021 angenommen, dass die Lernpfade genau so fein auflösend sind, wie die Beobachtungsdaten. Dadurch kann mit der Auswahl an Beobachtungsdaten bestimmt werden, wie klein die für die Adaptivität genutzten Lernobjekte sind. Insbesondere sind dadurch die Nutzungspfade gleich den Lernpfaden.

4.1.1 Nutzungspfade

Ein Nutzungspfad beschreibt auf der Menge aller möglichen Interaktionen mit dem Lernsystem die Sequenz an Interaktionen, die ein Nutzer bisher gewählt hat. Insbesondere werden dabei ungefiltert alle Beobachtungsdaten, die vom Tracker gesammelt wurden, im Modell als Pfad dargestellt. Dies kann allerdings häufig zu kleinschrittig sein, insbesondere in LMS sind dabei beispielsweise Szenen wie Dashboards und Ordneransichten, die der Nutzer zur Navigation betritt, aber keinen Lernfortschritt zum eigentlichen Lernziel macht.

4.1.2 Lernpfade

Lernpfade sind Pfade auf Szenen und Interaktionen, in denen der Nutzer einen Lernfortschritt für das Lernziel erlangt (Streicher und Heberle 2017). Diese sind also ähnlich zu den Nutzungspfaden mit dem Unterschied, dass die Aktivitäten nach Lernfortschritt gefiltert werden. Die Modellierung soll insbesondere zwei Eigenschaften beachten. Als Erstes sollen die Sequenzen der Aktivitäten, also Verb und Object aus den xAPI-Daten einzelner Nutzer nachvollziehbar sein. Außerdem soll die Häufigkeit der Reihenfolge von Aktivitäten dargestellt werden, also in welchem Anteil an Fällen eine Aktivität A vor einer anderen Aktivität B ausgeführt wurde.

4.1.3 Ideale Lernpfade

Als IPs bezeichne ich Lernpfade, die ein Idealer Nutzer befolgen würde, also Lernpfade, die denen von Experten gleichen. Dies bedeutet, dass der Nutzer alle nötigen Aktivitäten durchführt, um sein Lernziel zu erreichen (Streicher, Schönbein u. a. 2021). Um dies in einer datengetriebenen Modellierung umsetzten zu können, soll ideal in dem Zusammenhang bedeuten, dass der Lernpfad ein minimales Gewicht bezüglich einer geeigneten Metrik hat. In einer naiven Form ist solch eine Metrik beispielsweise die Dauer bis zum Erreichen des Lernziels. Allerdings haben frühere Arbeiten gezeigt, dass die Dauer bis zur Vollendung einer Lerneinheit nicht mit dem tatsächlichen Lernfortschritt korreliert (Loh u. a. 2014). Weitere Faktoren könnten z.B. Frustration durch die Lerneinheit sein und sind als pädagogischer Faktor nicht Gegenstand dieser Arbeit. Zur Vereinfachung wird daher in dieser Arbeit die Metrik durch die konstante Funktion $f(v) = 1$ für alle Lerneinheiten verwendet. Das heißt, in den Beobachtungsdaten sollen alle Lerneinheiten gleich effektiv sein.

4.2 Struktur von Graphen der Lernpfadmodelle

Da das Ziel ist, die Daten der xAPI als Graph zu modellieren, wird zunächst geklärt, welche Daten dazu in Frage kommen. Dazu werden folgende Objekte betrachtet, die in der Modellierung relevant sein können, um im nächsten Schritt verschiedene Zuordnungen zu Knoten und Kanten in Graphen zu vergleichen:

- Nutzer (`statement.actor`)
  Da ein Empfehlungssystem für verschiedene Nutzer entwickelt wird, müssen die Nutzer auch in der Modellierung unterscheidbar bleiben. Die Nutzer müssen also in einer Modellierung vorkommen.

- Aktivitäten (`statement.object`)
  Die Aktivitäten sind zusammen mit den Verben die Elemente, die beschreiben, was der Nutzer tatsächlich gelernt oder erfahren hat. Damit sollen sie ebenso notwendigerweise in der Modellierung enthalten sein.

- Verben (`statement.verb`)
  Die Verben stehen in engem Zusammenhang mit den Aktivitäten und geben an, was der Nutzer gerade macht. Daher können sie entweder explizit separat oder zusätzlich zu Aktivitäten am gleichen Objekt im Graphen, also Knoten oder Kante, gespeichert werden.

- Szenen (aus IPM (Streicher, Schönbein u. a. 2021))
  Szenen beschreiben den Zustand des Lernsystems (Streicher, Leidig u. a. 2018), also können diese naheligenderweise als Knoten dargestellt werden. Szenen können sich im IPM nur durch das Ausführen von Aktivitäten ändern, daher können diese auch implizit durch die vorangegangenen Aktivitäten in einer Modellierung dargestellt werden.

- Wissensstand
  Eine weitere Möglichkeit ist es, alle gesammelten Erfahrungen explizit darzustellen. Dies erfordert aber eine exponentielle Anzahl an Objekten in einer Modellierung. Allerdings kann man sehr einfach Nutzer mit gleichem Wissensstand aber einer anderen Reihenfolge an Erfahrungen erkennen. Wissensstände sind Zustände und sollten daher als Knoten modelliert werden.

- Metrik zur Bewertung (`statement.result`)
  Um aussagekräftige Pfade zu finden, benötigt man eine Metrik zur Bewertung, wie „gut" der Nutzer eine Aktivität absolviert hat. Idealerweise wird vom Tracker, der die xAPI-Daten erstellt, direkt ein Ergebnis (`statement.result.score`) mit angegeben. Im verwendeten Beispiel von LMS ist allerdings in vielen Lektionen (z.B. Texte) keine Erfolgskontrolle möglich.

- Zeitpunkt (`statement.timestamp`)
  Der Zeitpunkt einer Aktivität ist wichtig, um eine Reihenfolge an Aktivitäten für einen Nutzer zu erhalten. Diese kann implizit durch gerichtete Kanten realisiert werden.

Insbesondere werden also die Tripel aus den xAPI-Statements benötigt, wie sie in TRIPLEAD-APT vorgesehen (*Innovationswettbewerb INVITE - BMBF* 2022) sind.

In Tabelle 4.1 sind verschiedene Möglichkeiten zur Zuordnung der genannten Objekte auf Knoten und Kanten im Graph genannt. Durch die Möglichkeit von neo4j, Knoten und Kanten

	Aktivitäten	Verben	Szene	Wissensstand	User
1	Kante	an Kante	implizit	Knoten	Knoten & an Kante
2	Kante	Kante	Knoten	implizit	Knoten & an Kante
3	Knoten	Knoten	Knoten	implizit	Knoten & an Kante
4	Knoten	Kante	implizit	implizit	Knoten & an Kante
5	Knoten	Knoten	implizit	implizit	Knoten

Tabelle 4.1: Möglichkeiten zur Zuordnung der Objekte aus xAPI-Daten zu Graph-Objekte

mit Labels zu markieren, können verschiedene Objekte auf Knoten bzw. Kanten zugeordnet werden und trotzdem unterschieden werden. Insbesondere entspricht die 4. Modellierung der Modellierung aus Streicher, Schönbein u. a. 2021. Im folgenden sind einige Anmerkungen zu den Modellierungsmöglichkeiten aus Tabelle 4.1.

1. Durch die xAPI-Statements können sämtliche Lernerfahrungen dokumentiert werden. Durch die Menge aller Lernerfahrungen als Knoten können Nutzern mit dem gleichen Erfahrungsschatz die gleichen Aktivitäten vorgeschlagen werden.

   Wenn man Kanten zu allen Knoten mit Teilmengen jedes Knoten als Kante mit Gewicht 0 einfügt, kann man mit einem Kürzeste-Wege-Algorithmus einen IP finden.

   Die Kanten aller Aktivitäten eines Nutzers bilden einen Walk, da der Nutzer nur durch weitere Aktivitäten neue Erfahrung sammelt.

   Da die Anzahl der benötigten Knoten für n Aktivitäten in $\Theta(2^n)$ wächst, wurde dieser Ansatz nicht weiter verfolgt.

2. Die Szenen sind die Knoten. Jede Aktivität mit zugehörigem Verb wird als Übergang zwischen zwei Szenen angesehen. Dabei sind Aktivitäten, die die Szene nicht wechseln, Kanten mit gleichem Start- und Zielknoten.

   Da die Szenen nicht notwendigerweise im xAPI-Standard erfasst werden muss, ist diese Modellierung für den Anwendungsfall nicht allgemein anwendbar. Auch hier lassen sich die IPs mit kürzesten Wegen finden. Dazu muss dann eine Szene als Zielknoten angegeben werden, die nur bei erfolgreichem Erreichen des Lernziels (beispielsweise eine Prüfung) von Nutzern erreichbar ist.

3. Analog zum vorherigen Fall sind auch hier die Szenen Knoten, die Informationen an Kanten werden allerdings durch weitere Knoten gespeichert. Dadurch sind sie durch neo4j leichter auffindbar.

Durch das Verwenden von Szenen ist diese Modellierung wie die vorherige auch nicht allgemein anwendbar.

4. Diese Modellierung speichert die Szenen implizit durch die zuletzt ausgeführte Aktivität. Ein Nachteil davon ist allerdings, dass Szenen, die durch unterschiedlichen Aktivitäten erreicht werden können, nicht gleichen Knoten entsprechen. Da die Aktivitäten Knoten sind und aufeinander folgende Aktivitäten eines Nutzers Kanten sind, wird der Pfad von Aktivitäten eines Nutzers direkt als Subgraph dargestellt.

   Kürzeste Wege zwischen Nutzern und dem Zielknoten sind auch hier eine Methode, um IPs zu finden, da diese Pfade eine Vereinigung von Teilpfaden von Nutzern sind.

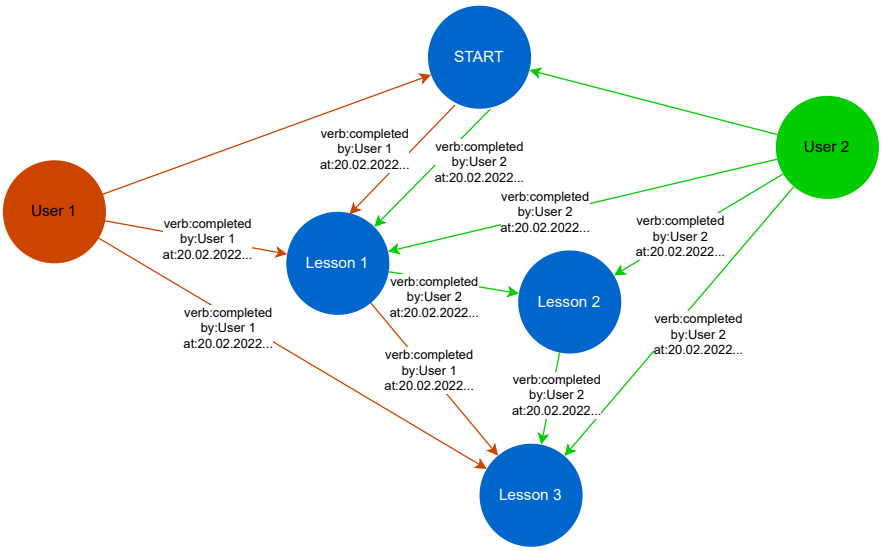
5. Diese Modellierung ist ähnlich zu der vorherigen, nur dass wieder Verben explizit als Knoten dargestellt werden. Da im weiteren Verlauf der Performance-Berechnung keine Verben ohne Aktivitäten verwendet werden, bietet diese Modellierung keine Vorteile zur Vorherigen, benötigt aber mehr Knoten und Kanten.

Die User werden dabei als separate Knotenmenge gespeichert. Um Kanten zwischen Aktivitäten eindeutig zu Pfaden zuzuordnen, kann eine id des Users und der Zeitpunkt des Statements gespeichert werden.

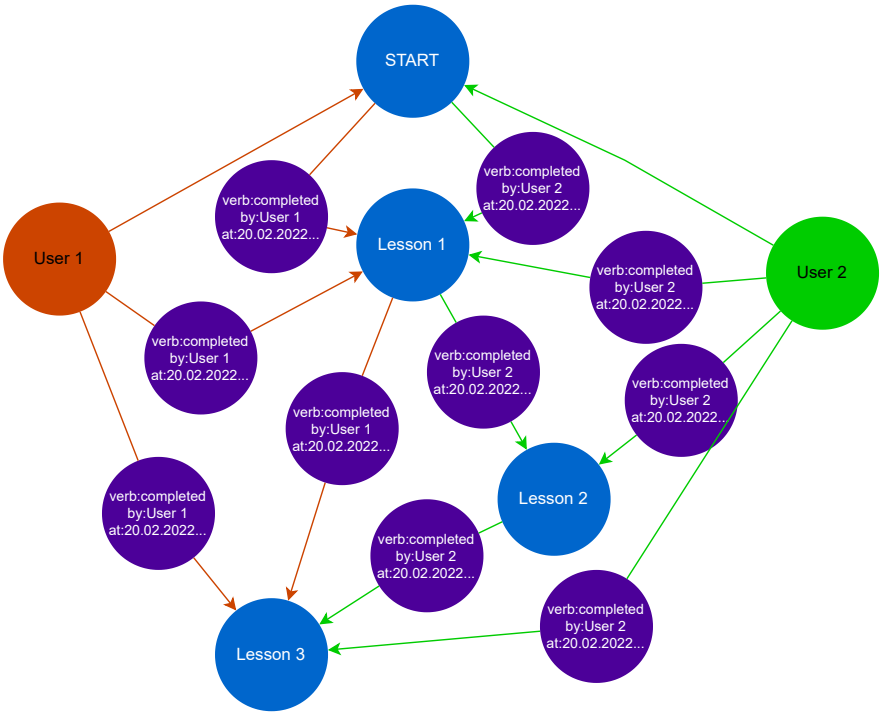
4.3 Verwendete & resultierende Typen von Graphen

Auf der Datengrundlage von xAPI-Daten können einige Beobachtungen zu den in Abschnitt 2.1 beschriebenen Arten und Strukturen der verwendeten Graphen gemacht werden. Kantenrichtungen können verwendet werden, um eine zeitliche Relation zwischen Aktivitäten oder Wissensständen zu modellieren, beispielsweise bedeutet dann $(A,B) \in E$ dass Aktivität A vor B ausgeführt wurde. Außerdem will man beliebige Namen für Aktivitäten, den Zeitpunkt der Durchführung und den Nutzer darstellen. Dies kann in neo4j mit Knoten- und Kanteneigenschaften realisiert werden (Hodler 2019).

Wenn die zeitliche Relation der Aktivitäten in den näher betrachteten Modellierungen 4 und 5 in Tabelle 4.1 mit gewichteten Kanten dargestellt wird, ist ein einzelner IP ein azyklischer Graph, allerdings könnten Nutzer eine Aktivität mehrfach ausführen und dadurch können in Lernpfaden und Nutzungspfaden Kreise entstehen. Die Graphen sind nicht notwendigerweise zusammenhängend. Insbesondere wenn wenig Daten zur Verfügung stehen kann es Aktivitäten geben, die noch nicht ausgeführt wurden. Um einen IP mittels kürzesten Pfaden für einen Nutzer zu finden, muss allerdings wenigstens ein Pfad zwischen einem Start-Knoten und dem Lernziel existieren. Diese Eigenschaften lassen sich in 4.2 sehen.



(a) Modellierung nach Punkt 4 aus Tabelle 4.1



(b) Modellierung nach Punkt 5 aus Tabelle 4.1

Abbildung 4.1: Modellierung der gleichen Pfade. Wenn man jeweils den Teilgraph ohne Nutzer und Verben zwischen Nutzer und Aktivität betrachtet, bleiben genau die Nutzungspfade der Nutzer auf geteilten Knoten für die Aktivitäten übrig.

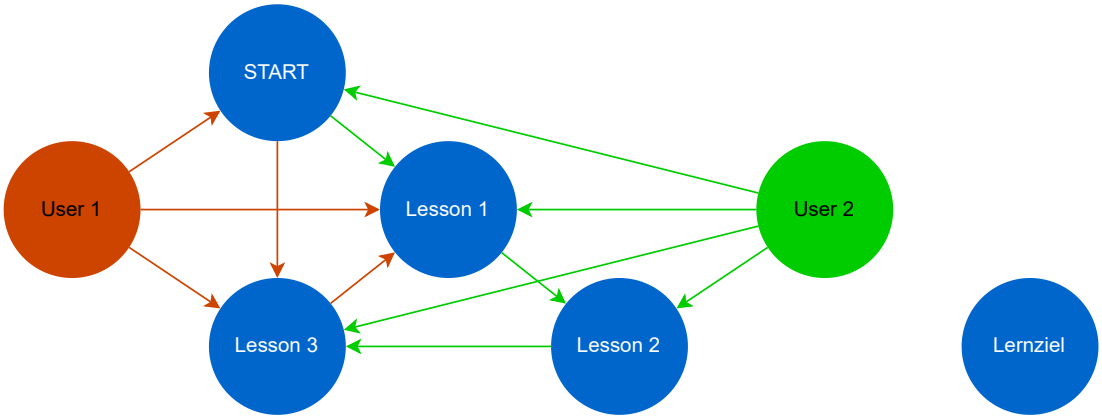


Abbildung 4.2: Der Graph ist nicht zusammenhängend, da noch kein Nutzer das Lernziel erreicht hat. Kanten einzelner Nutzer (farblich unterschieden) bilden eingeschränkt auf die Knoten, die Aktivitäten darstellen, einen Pfad. Beide Nutzerpfade zusammen bilden einen Kreis (*Lesson 1, Lesson 2, Lesson 3*).

5 Algorithmische Analyse der Lernpfadmodelle

In diesem Kapitel werden Ansätze vorgestellt, um Ideal Paths (IPs) und die Progress Performance mit Techniken der Graph Analytics zu bestimmen. Für die IPs wird dafür ein kürzester Wege Algorithmus betrachtet. Außerdem wird die Progress Performance mithilfe von Ideal Paths definiert. Dazu wird die von Loh u. a. 2014 verwendete Jaccard Similarity angepasst, sodass diese nicht mehr den Lernfortschritt sondern nur die Effektivität des Lernens bezüglich Ideal Paths (IPs) berücksichtigt.

5.1 Berechnung von Idealen Pfaden

Aufgrund der theoretischen Betrachtung der verschiedenen Modellierungen aus Tabelle 4.1 wird die Modellierung verwendet, die Aktivitäten und User an Knoten und Verben an Kanten modelliert. Kanten entstehen dabei zwischen Nutzer und Aktivität, um die Tripelstruktur abzubilden und zwischen von einem Nutzer direkt aufeinander folgend ausgeführten Aktivitäten. An der Kante steht das Verb, das zu der Aktivität gehört, an dem die gerichtete Kante endet. Dadurch entsteht auch der Bedarf eines gemeinsamen Startknotens, um das erste Verb im Lernpfad darstellen zu können. Diese Modellierung wurde ebenfalls von Streicher, Schönbein u. a. 2021 verwendet.

Da ein IP ein Pfad vom Startknoten zum Knoten des Lernziels sind, eignen sich Algorithmen für kürzeste Wege zur Suche dieser Pfade. Um verschiedene IPs zu finden, die dem Nutzer besser angepasst sein könnten, verwende ich wie auch Streicher, Schönbein u. a. 2021 Yen's k-Shortest-Paths Algorithmus. Zentralitätsalgorithmen eignen sich, um Lerneinheiten zu finden, die den größten Einfluss auf alle anderen Lerneinheiten haben oder von den meisten Lernpfaden verwendet werden, allerdings finden diese nur einzelne Knoten und keine Pfade.

Zur Evaluation können in synthetischen Nutzungspfaden die IPs mit dieser Methode berechnet werden und dann mit den erwarteten Idealen Pfaden verglichen werden.

5.2 Performance auf Idealen Pfaden

In Loh u. a. 2014 wurde gezeigt, dass die Jaccard Similarity

$$JACC(A,B) = \frac{|A \cap B|}{|A \cup B|} \in [0,1] \subseteq \mathbb{R}$$

eine geeignete Metrik ist, um Experten und Anfänger mit Interaktionssequenz A für Interaktionssequenzen B eines als Experte bekannten Nutzer zu unterscheiden. Da die IPs genau die effektivsten Lernpfade sind, ist eine Interaktionssequenz eines Experten genau entlang eines IP. Dadurch kann man mit der Jaccard Similarity die Performance eines Nutzers mit einem IP vergleichen. Dies gibt an, wie weit der Nutzer zum Lernziel fortgeschritten ist

Für den Anwendungsfall wird eine Metrik für die Progress Performance gesucht, die als eine Art der Fortschrittsgeschwindigkeit angibt, wie effektiv der Nutzer bisher gelernt hat. Nicht mit einbezogen werden soll dabei, wie weit der Nutzer im Lernprozess fortgeschritten ist. Insbesondere soll die Progress Performance dadurch bereits bei unvollständigen Pfaden eine Aussage darüber treffen können, wie ähnlich sich der Nutzungspfad und der bisherige Teil eines IP sind. Beispielsweise sollen Nutzer, die am Anfang eines IP sind, die gleiche Performance erhalten wie Nutzer, die das Lernziel erreicht haben und prozentual gleich viele Umwege genutzt haben. Ziel davon ist es, zu entscheiden ob die vom Nutzer durchgeführten Interaktionen zielführend für ein gegebenes Lernziel sind und dies als numerischen Wert im Intervall $[0,1] \subseteq \mathbb{R}$ anzugeben. Die Jaccard Similarity ist in diesem Fall nicht in der vorgestellten Form anwendbar. Beispielsweise erhält ein Nutzer, dessen einzige Interaktion mit der ersten eines 100 Interaktionen langen IP gleicht trotzdem nur eine Ähnlichkeit von $\frac{1}{100} = 1\%$, obwohl er bislang den IP befolgt.

Daher stelle ich eine Anpassung vor, die darauf beruht, den IP nicht vollständig mit einzubeziehen.

$$\text{Perf}_1^P = \frac{\text{Progress}}{\text{Time}} = \frac{\#(\text{Schritte von Nutzer und IP})}{\#\text{Nutzerschritte}}$$

Dafür benötigt man eine genaue Definition der Nutzerschritte sowie analog die Anzahl der Schritte des IPM. In einem ersten Ansatz sind dies genau die Aktivitäten, also in der Graphmodellierung die Knoten. Dieser Ansatz, der auch in Streicher, Schönbein u. a. 2021 verfolgt wurde, berücksichtigt allerdings nicht die Reihenfolge der Aktivitäten. Ein Nutzer, der einen IP genau rückwärts durchläuft wird nicht die Grundlagen haben, um in seinen ersten Lektionen effektiv lernen zu können.

Daher ist ein zweiter Ansatz, die Kanten in der Modellierung als Nutzerschritte zu zählen, dies

entspricht den Ausführungen von zwei Aktivitäten direkt hintereinander wie im Idealen Pfad und wurde auch von Loh u. a. 2014 verwendet. Hierbei wird ein Nutzer, der nach jeder Aktivität eines IPs eine andere Aktion durchführt keine Lerngeschwindigkeit von Perf_P diagnostiziert bekommen, obwohl er das Lernziel erreichen kann.

Daher ist ein dritter Ansatz, die beiden vorherigen Ansätze gewichtet zu kombinieren, für einen Lernpfad eines Nutzers P_U und einen IP sowie den Gewichtungsparameter $\alpha \in [0,1]$:

$$\text{Perf}_1^P(P_U,P_I) = \frac{\text{Progress}}{\text{Time}} = \alpha\frac{|V(P_U) \cap V(P_I)|}{|V(P_U)|} + (1-\alpha)\frac{|E(P_U) \cap E(P_I)|}{|E(P_U)|}$$

Es gilt $\text{Perf}_1^P \in [0,1]$, da

$$0 = \frac{0}{|V(P_U)|} \leq \frac{|V(P_U) \cap V(P_I)|}{|V(P_U)|} \leq \frac{|V(P_U)|}{|V(P_U)|} = 1$$

$$0 = \frac{0}{|E(P_U)|} \leq \frac{|E(P_U) \cap E(P_I)|}{|E(P_U)|} \leq \frac{|E(P_U)|}{|E(P_U)|} = 1$$

$$0 = \alpha \cdot 0 + (1-\alpha) \cdot 0 \leq \text{Perf}_1^P$$

$$1 = \alpha + 1 - \alpha \geq \alpha \cdot 1 + (1-\alpha) \cdot 1 \geq \text{Perf}_1^P$$

In allen Fällen gibt es mehrere Lernpfade, allerdings soll dem Nutzer ein Lernpfad empfohlen werden, der seinem bisherigen Lernpfad am nächsten ist. Daher wird der IP genutzt, auf dem der Nutzer die höchste Performance hat:

$$\text{Perf}^P(P_U) = \max_{P_I \in IPs} \text{Perf}_1^P(P_U,P_I)$$

In den Beispielen in Abbildung 5.1a bis Abbildung 5.1c sind die durchgezogenen Kanten im Lernpfad des Nutzers U, die anderen Kanten gehören zu zwei Ideal Paths.

Im ersten Beispiel (Abb. 5.1a) hat der Nutzer U bereits alle Lektionen vor dem Lernziel auf dem Idealen Pfad IPM_1^U abgeschlossen, also sollte die Empfehlung sein, das Lernziel zu prüfen. Zur Performanceberechnung werden die Knoten und Kanten auf dem Lernpfad gezählt:

$$|P_U| = |\{L_1, L_\beta, L_2, L_3, L_3^+\}| = 5$$

$$||P_U|| = |\{UL_1, L_1L_\beta, L_\beta L_2, L_2L_3, L_3L_3^+\}| = 5$$

$$|P_U \cap IPM_1| = 3, |P_U \cap IPM_2| = 3$$

$$||P_U \cap IPM_1|| = 1, ||P_U \cap IPM_2|| = 1$$

Mit $\alpha = 0.5$ resultiert daraus die Progress Performance für IPM_1^U und IPM_2^U gleichermaßen:

$$\text{Perf}^P(P_U) = \max_{P_I \in IPs} \frac{1}{2}\frac{|P_U \cap P_I|}{|P_U|} + \frac{1}{2}\frac{||P_U \cap P_I||}{||P_U||} = \frac{2}{5}$$

Im zweiten Beispiel (Abb. 5.1b) gilt für IPM_1

$$|P_U| = |\{L_x, L_1, L_y, L_z\}| = 4$$

$$|P_U \cap IPM_1| = 1, |P_U \cap IPM_2| = 0$$

$$||P_U \cap IPM_1|| = 0, ||P_U \cap IPM_2|| = 0$$

und mit $\alpha = 0.5$ resultiert daraus die Progress Performance

$$\text{Perf}^P(P_U) = \max_{P_I \in IPs} \frac{1}{2}\frac{|P_U \cap P_I|}{|P_U|} + \frac{1}{2}\frac{||P_U \cap P_I||}{||P_U||} =$$

$$\max\{\frac{1}{8} + 0, \frac{0}{8} + 0\} = \frac{1}{8}$$

Insbesondere ist hier der Schnitt mit dem zweiten Idealen Pfad IPM_2 leer und dementsprechend die Progress Performance auf diesem Pfad gleich 0.

Im dritten Beispiel (Abb. 5.1c) ist

$$|P_U| = |\{L_x,L_\alpha,L_w,L_\beta,L_y, L_z,L_2\}| = 7$$

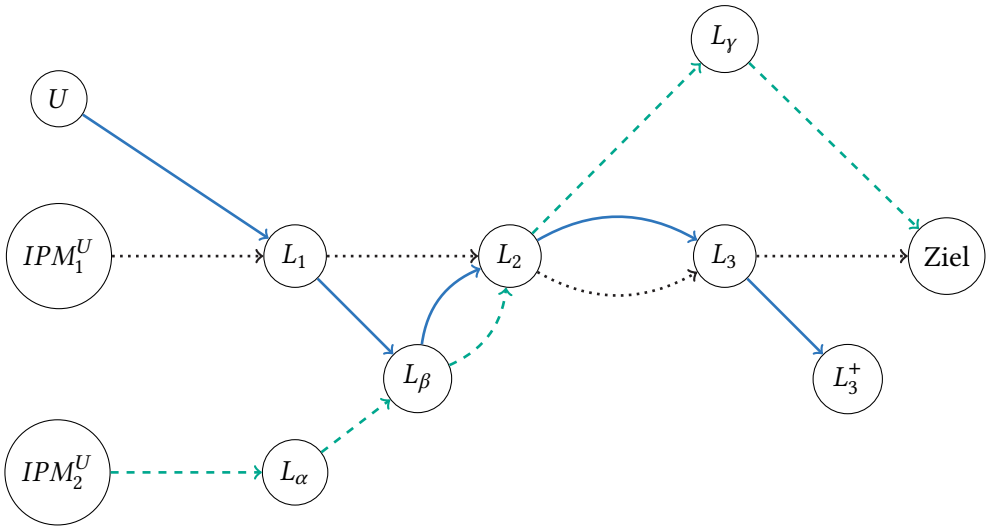
$$|P_U \cap IPM_1| = 1, |P_U \cap IPM_2| = 3$$

$$||P_U \cap IPM_1|| = 0, ||P_U \cap IPM_2|| = 0$$

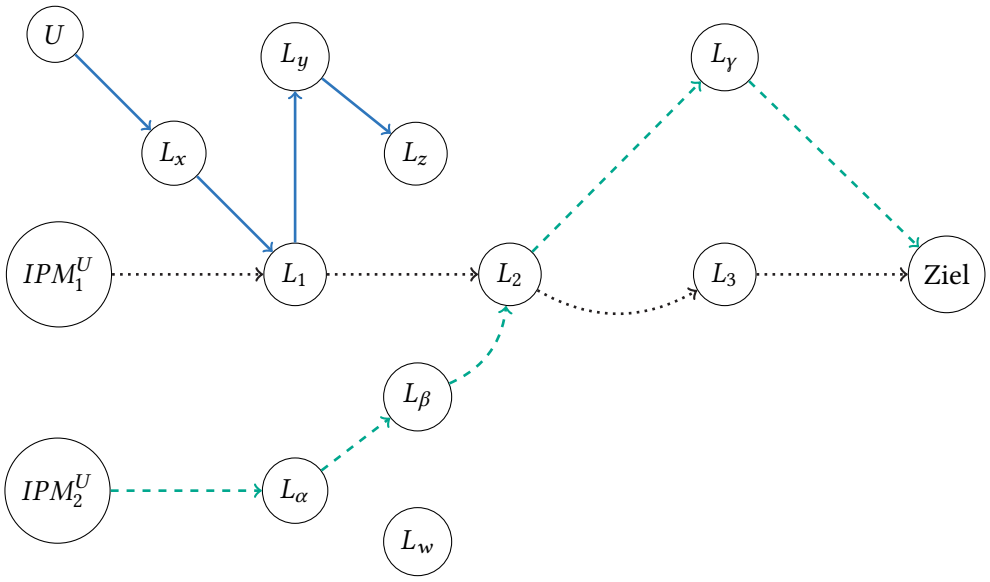
und mit $\alpha = 0.5$ resultiert daraus die Progress Performance für IPM_2

$$\text{Perf}^P(P_U) = \max_{P_I \in IPs} \frac{1}{2}\frac{|P_U \cap P_I|}{|P_U|} + \frac{1}{2}\frac{||P_U \cap P_I||}{||P_U||} =$$

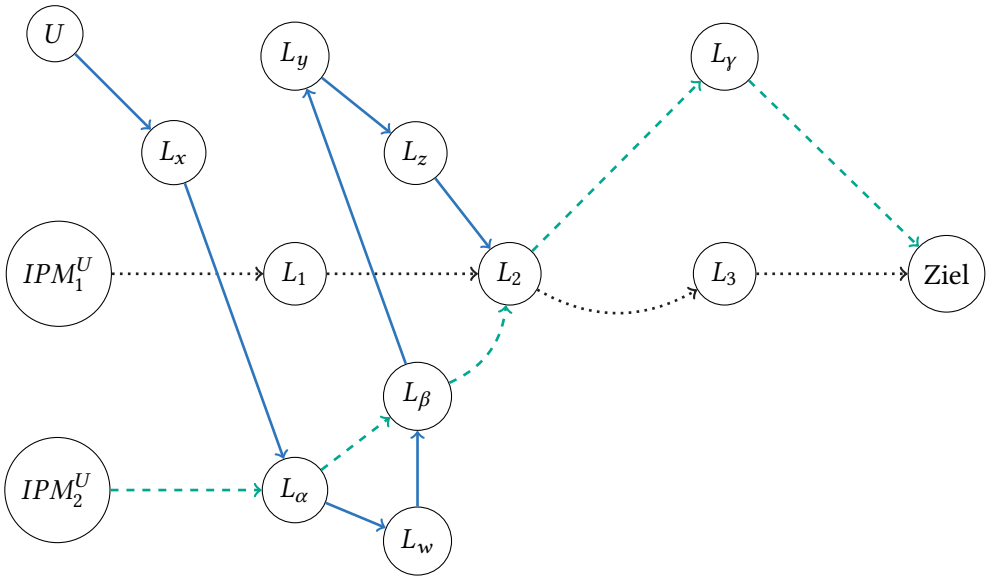
$$\max\{\frac{1}{14} + 0, \frac{3}{14} + 0\} = \frac{3}{14}$$



(a) Nutzungspfad (blau) auf beiden Nutzungspfaden, z.B. Nutzer hat sich nach der ersten Lerneinheit zu schwach eingeschätzt



(b) Nutzungspfad (blau) mit wenigen zielführenden Aktivitäten



(c) Nutzungspfad (blau) mit vielen Unterbrechungen durch nicht-zielführende Pfade, z.B. ein sehr abgelenkter Nutzer

Abbildung 5.1: Drei Nutzungspfade für das Lernziel „Ziel" mit zwei Idealen Pfaden (jeweils grün gestrichelt: ausführlicher für Anfänger, schwarz gepunktet: kürzer für Fortgeschrittene

6 Anwendung & Verifikation

Zur Verifikation wurde ein Prototyp implementiert und aufgrund synthetischer Daten manuelle Tests durchgeführt. Zwei dabei gefundene Probleme der Modellierung werden an Beispielen gezeigt. Dabei handelt es sich im Wesentlichen um Optimierungen des Algorithmus zum Finden der Ideal Paths (IPs). Weiterhin werden die Ergebnisse diskutiert und eingeordnet sowie offene Fragen genannt.

6.1 Implementierung eines Prototyps zur Berechnung von IPs & Performance

Eine testweise Implementierung erfolgte auf Basis des ELAI-DDA (Dynamic Difficulty Adjustment) Service (Streicher, Roller u. a. 2017). Dadurch existiert bereits der Rahmen eines Microservice, dessen interne Logik angepasst werden kann und im Abschnitt 6.3 einfach eingebunden werden kann. Wie in Abschnitt 1.2 angegeben nutzt der Microservice ein Eingabe-Verarbeitung-Ausgabe Schema. Die einzelnen Schritte werden im folgenden erläutert.

6.1.1 Struktur der Eingabedaten

Eine direkte Eingabe erfolgt nur für den Nutzer und das Lernziel, für die ein Performancewert und eine Empfehlung für den Lernpfad des Nutzers zum Erreichen des Lernziels angefragt wird. Diese werden direkt in der URL durch Parameter angegeben, beispielsweise durch „`.../response?user=alice@example.com&goal=http://example.com/Lernziel`".

Zur Berechnung von Performance und Empfehlung sowie dem dafür benötigten IP werden zusätzlich noch Aktivitätsdaten benötigt. Diese werden im Format des xAPI-Standards von einem Learning Record Store (LRS) bereitgestellt und werden aktiv vom GraphPerformance-Service vom konfigurierten LRS abgerufen. Da diese Daten in der Graphdatenbank neo4j zur Analyse gespeichert werden, werden nur seit der letzten Abfrage neu gespeicherte Statements abgefragt.

```
MERGE (user:Agent {userid: $userid})
MERGE (activity:Activity {name: $name, id: $activityid})
WITH user, activity
CREATE (user) -[:VERB {name: $verb, time: $time}]-> (activity)
RETURN id(activity) AS node_id
```

Listing 6.1: Cypher-Anfrage zur Erstellung einer neuen Kante zwischen Nutzer und Objekt des statements für eine neue Interaktion. Wörter, die mit $ beginnen werden durch bereinigte Parameter aus den Statements substituiert.

```
MATCH (user:Agent {userid: $userid})
WITH user
MATCH (activity:Activity) -[verb]- (user)
WITH user, verb, activity ORDER BY verb.time ASC
WITH COLLECT(activity) AS activities
FOREACH (n in RANGE(0, size(activities) - 2) |
FOREACH (prec IN [activities[n]] |
FOREACH (next IN [activities[n + 1]] |
MERGE (prec) -[:NEXT]-> (next))))
RETURN activities
```

Listing 6.2: Cypher-Anfrage zur Erstellung der Lernpfade eines Nutzers. Wörter, die mit $ beginnen werden durch bereinigte Parameter aus den Statements substituiert.

6.1.2 Verarbeitungsschritte zur Berechnung der IPs & Performance

Die Verarbeitung wurde möglichst modular gehalten, sodass die drei Bereiche der Modellierung, Berechnung der IP und die Berechnung der Performance einfach durch andere Objekte austauschbar sind, die die Modellierung oder Berechnung mit einer anderen Strategie durchführen.

Im ersten Schritt müssen die neuen xAPI-Daten in die Modellierung überführt werden. Dazu wird zunächst mit dem CYPHER-Ausdruck aus Listing 6.1 Nutzer und Lernobjekt des Statements (`statement.actor` & `statement.object`) in der Datenbank gefunden oder erstellt und eine Kante mit dem Verb und Zeitstempel (`statement.verb` & `statement.timestamp`) eingefügt. Danach werden die Lernobjekte des Nutzers mit dem CYPHER-Ausdruck aus Listing 6.2 nach dem Zeitstempel sortiert, um die Kanten des eigentlichen Lernpfads in richtiger zeitlicher Reihenfolge einzufügen.

Die Berechnung der IP erfolgt dann durch Graphalgorithmen aus der neo4j Graph-Data-Science Bibliothek (GDS). Diese stellt Yen's k-Shortest-Paths Algorithmus zur Verfügung, sodass die IP-Berechnung mit dieser Methode eine Cypher-Abfrage wie in Listing 6.3 auf einem dafür aus allen Aktivitätsknoten erstellten benannten Graph ist. Die Rückgabe besteht

```
CALL gds.shortestPath.yens.stream($graphname, {
    sourceNode: $source,
    targetNode: $target,
    k: $k
}
YIELD nodeIds, index, totalCost AS distance
RETURN nodeIds, index, distance
```

Listing 6.3: Cypher-Anfrage zur Berechnung der IPs mit Yen's *k*-Shortest-Paths Algorithmus zwischen einem universellen Startknoten und dem angegebenen Lernziel. Wörter, die mit $ beginnen werden durch bereinigte Parameter aus den Statements substituiert.

```
def intersection(a, b):
    """
    Return the intersection of two lists as a set
    """
    return set(a).intersect(set(b))

def user_overlap(user_activities, ipm_activities):
    """
    Overlap part of user activities with ipm activities (user
        activities as all activities)
    """
    return float(len(intersection(user_activities, ipm_activities)))
        / len(user_activities)
```

Listing 6.4: Berechnung der Performance auf einem IP mit der vorgestellten Methode .

aus mehreren Listen von Knoten-IDs, da die Elemente für die Performanceberechnung nur eindeutig sein müssen und nicht alle Metadaten benötigt werden. Außerdem kann dann die Empfehlung für das nächste Lernobjekt wieder aus der ID bestimmt werden.

Der letzte Schritt ist die eigentliche Bestimmung der Performance. Dazu ist die Formel wie in Listing 6.4 zu sehen in Python implementiert worden. Aus allen Pfaden wird der Pfad ausgewählt, der das Ergebnis dieser Funktion maximiert. Die Performance ist dann die Performance des Nutzers auf diesem Pfad und dem Nutzer wird die nächste Aktivität auf diesem Pfad empfohlen, die er noch nicht ausgeführt hat.

Die Schritte der Verarbeitung sind nochmal in Abbildung 6.1 visualisiert.

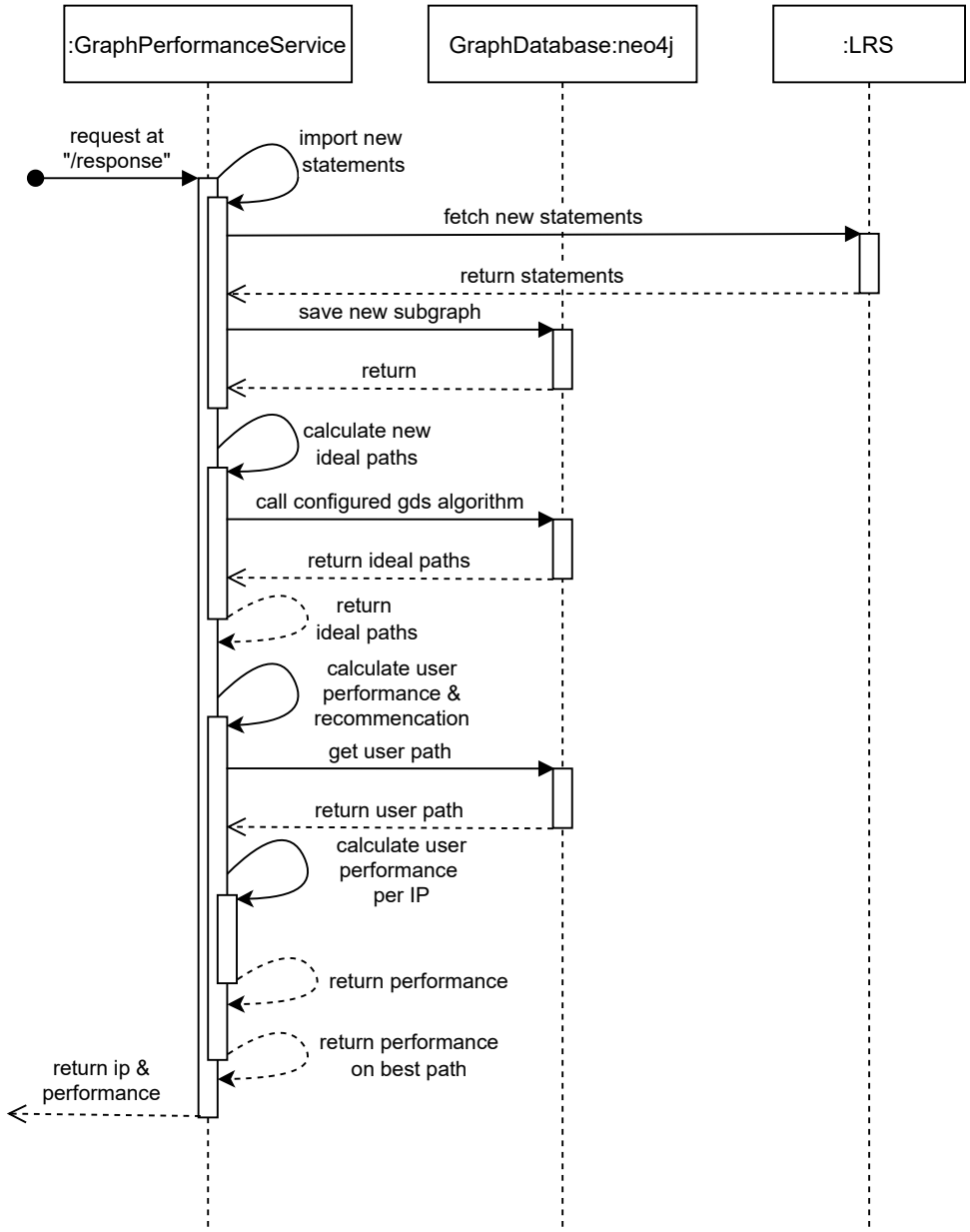


Abbildung 6.1: Ein Sequenzdiagramm der Verarbeitungsschritte des GraphPerformanceService und Kommunikation mit den externen Datenbanken

```
{
    "success": true,
    "performance": 0.5,
    "recommendation": {
        "next_activity": {
            "id": "https://example.com/courseX/Lesson1",
            "name": "https://example.com/definition/lesson"
        }
    }
}
```

Listing 6.5: Beispiel einer Antwort des Graph-Performance-Service im JSON-Format.

6.1.3 Struktur der Ausgabedaten

Die Ausgabe ist eine Antwort des Servers im JSON-Format. In dieser wird angegeben, ob die Berechnung erfolgreich war. Falls ja, enthält diese auch die Performance als Gleitkommazahl und Name und ID der xAPI-Aktivität. Das genaue Format ist im Beispiel in Listing 6.5 zu sehen.

6.2 Auswertung der Analysetechniken

Zur Auswertung werden synthetische Daten verwendet. Dies sind konstruierte Folgen von xAPI-Statements, sodass bestimmte IPs erwartet werden können. Außerdem können so Probleme an kleinen Beispielen demonstriert werden.

6.2.1 Ideal Path Berechnung

IPs sind nach Definition Lernpfade, auf denen man das Lernziel bestmöglich erreichen kann. Wenn ein Nutzer eine hohe Performance Perf^P hat, soll dies genau bedeuten, dass der Nutzer auf einem guten Pfad zum Erreichen des Lernziels ist. Dies soll ein Pfad sein, der möglichst gut bewertete Lektionen (niedrige positive Zahl) aber keine unnötigen Umwege enthält.

Im ersten Beispiel (Abbildung 6.2a) bestehen die xAPI-Daten aus den zwei Expertenpfaden *'Adding digits Lesson', 'Adding numbers Lesson', 'Subtraction Lesson', 'Multiplication Lesson', 'Simple Math Test Out'* und *'Addition Lesson', 'Subtraction Lesson', 'Digit Multiplication Lesson', 'Number Multiplication Lesson', 'Simple Math Test Out'*, nach Annahme sind alle Lernobjekte darin gleich wertvoll. Die Berechung der IPs lieferte dabei beide Pfade sowie die folgenden zwei weiteren, kürzeren Pfade: *'Addition Lesson', 'Subtraction Lesson', 'Multiplication Lesson', 'Simple Math Test Out'* und *'Adding digits Lesson', 'Adding numbers Lesson', 'Subtraction Lesson', 'Digit Multiplication Lesson', 'Number Multiplication Lesson', 'Simple Math Test Out'*.

Im zweiten Beispiel existieren zur Verdeutlichung eines Problems zwei Expertenpfade zum Erlernen der Berechnung des Binomialkoeffizienten. Ein erfolgreicher Nutzer lernt dabei das Berechnen der Fakultät vor der Division, der andere Nutzer genau umgekehrt. Dadurch entstehen die Expertenpfade *'Math Multiplication', 'Math Factorial', 'Math Division', 'Math Binomial'* und *'Math Multiplication', 'Math Division', 'Math Factorial', 'Math Binomial'*. Die Berechnung der IPs findet allerdings zusätzlich die unvollständigen Pfade *'Math Multiplication', 'Math Division', 'Math Binomial'* und *'Math Multiplication', 'Math Factorial', 'Math Binomial'*, in denen entweder das Lernobjekt zur Division oder zur Fakultät fehlt.

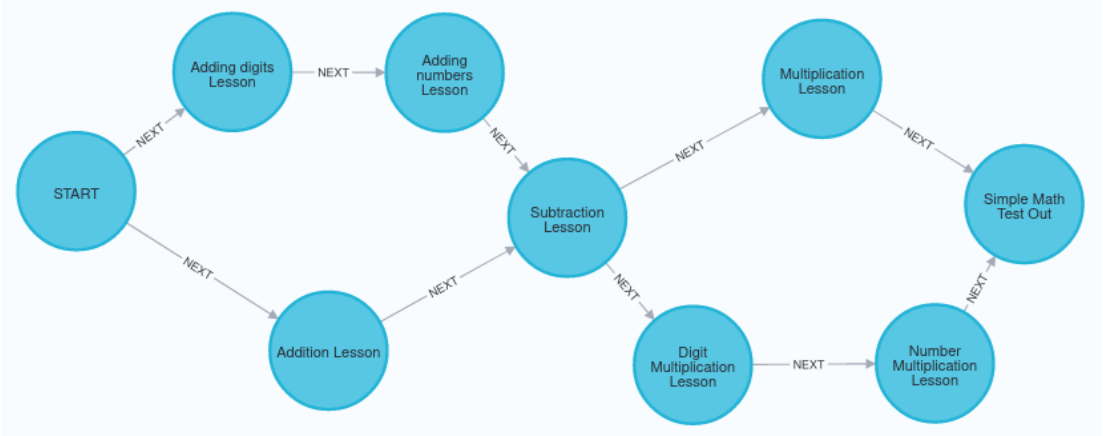
Zudem sind die korrekten Pfade die längeren Pfade, dadurch würden diese bei einem kleineren Parameter k, im Beispiel $k = 2$, in Yen's-k-Shortest-Paths Algorithmus nicht mehr gefunden werden.

6.2.2 Auswertung der Performance auf berechneten Idealen Pfaden

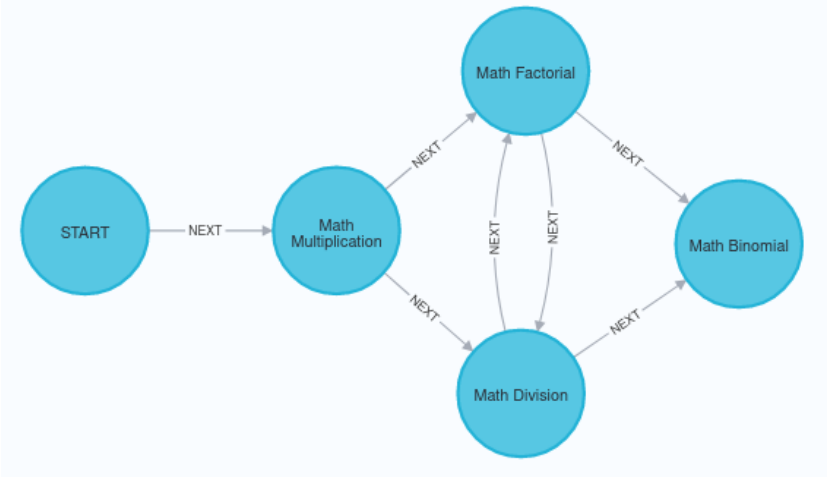
Der Wert Performance ist nach Konstruktion stark von der Auswahl der IPs abhängig. Bei der Verwendung von Yen's k-shortest-Paths Algorithmus ist zu beachten, dass der Parameter k in der getesteten Implementation unabhängig von der Größe und Struktur des Graphen beliebig festzulegen ist. In Kombination mit der Tatsache, dass für die Berechnung der IPs alle Interaktionen, auch die des Nutzers selbst, miteinbezogen werden, kann es passieren, dass dadurch ein IP berechnet wird, der irrelevante Interaktionen enthält. In Abbildung 6.2c sind zwei Expertenpfade in blau dargestellt, mit denen die Grundlagen für einen Arithmetiktest gelernt werden. Ein Nutzer, der nach zwei Lektionen zu dem davon größtenteils unabhängigen Thema Geometrie die erste Interaktion eines Expertenpfades durchführt, erhält den besten Performancewert 1.0, da *'Triangles Lesson', 'Rectangles Lesson', 'Addition Lesson', 'Subtraction Lesson', 'Arithmetics Test'* ein berechneter IP ist. Zusätzlich ist dieser genau so lang wie ein anderer Expertenpfad, also lässt sich dieser Pfad nicht sicher durch einen niedrigeren Parameter k beheben.

6.3 Einordnung im Use-Case

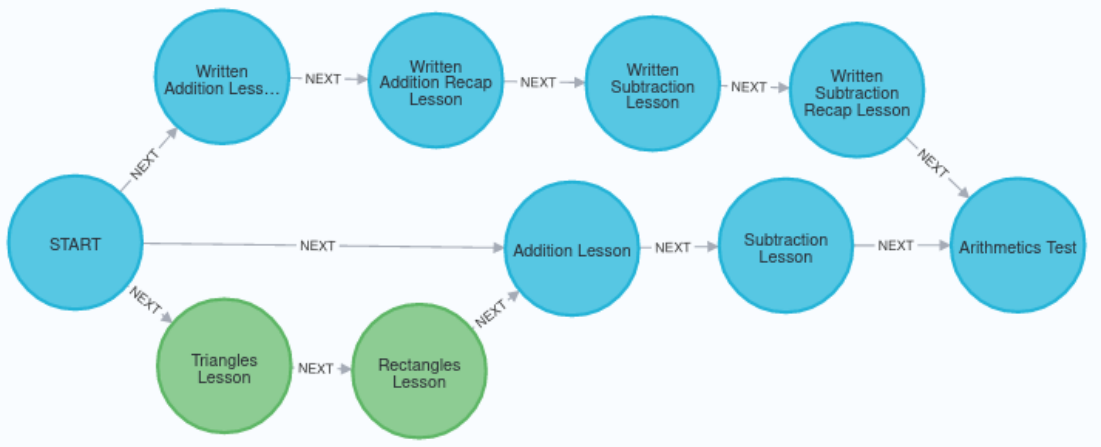
Im Anwendungsfall sollen in der Lernplattform Moodle adaptiv Empfehlungen für nächste Lerneinheiten oder Lernpfade vorgeschlagen und dem Nutzer präsentiert werden. Außerdem soll der Nutzer einen Hinweis bekommen, wie gut seine aktuelle Fortschrittsperformance ist. Die in dieser Arbeit vorgestellten Methoden können für die Berechnung des Vorschlags für die nächste Lerneinheit und die Performance auf dem nächsten IP verwendet werden. Der Nutzer soll in einem LMS die Möglichkeit haben, seine Performance und eine adaptive



(a) Zwei Expertenpfade (obere bzw. untere Knotenreihe). Offensichtlich gibt es noch zwei weitere Pfade zum Lernziel *'Simple Math Test Out'*



(b) Zwei Expertenpfade (obere bzw. untere Knotenreihe). Offensichtlich gibt es noch zwei weitere Pfade zum Lernziel *'Simple Math Test Out'*



(c) Zwei Expertenpfade (obere bzw. untere Knotenreihe). Offensichtlich gibt es noch zwei weitere Pfade zum Lernziel *'Simple Math Test Out'*

Abbildung 6.2: Graph-basierte Modellierung von xAPI-Statements

Empfehlung anzufragen. Wie in Abbildung 6.3 gezeigt, schickt das LMS dann einen request an einen externen Service, der den Graph Performance Service aufruft und einen Wert für die Performance und eine adaptive Empfehlung für den Nutzer zurück gibt. Der externe Service bereitet diese Antwort visuell auf und liefert ein im LMS einzubettendes Objekt zurück. Das LMS kann dieses dann einbetten und dem Nutzer anzeigen.

6.4 Diskussion

Als Bestandteil eines Assistenzsystems wurden mit den Ideal Path Models (IPMs) (Streicher, Schönbein u. a. 2021) verschiedene Modellierungen von beobachteten Verhaltensdaten entwickelt und analysiert. Dabei gab es im Wesentlichen zwei unterschiedliche mit xAPI-Daten umsetzbare Ansätze zur Modellierung von IPs, die mit den aus dem xAPI-Standard zur Verfügung stehenden Daten sinnvolle Graphen zu modellieren. Einer davon wurde allerdings wegen einem exponentiellen Speicherbedarf nicht weiter verfolgt. Zusätzlich wurden verschiedene Varianten vorgestellt, wie weitere Daten integriert werden können. Die Anwendung einer nicht weiter modifizierten Berechnung kürzester Pfade auf einer dieser Modellierungen hat gezeigt, dass die Modellierung tatsächlich die gesuchten IPs finden kann. Allerdings wurden durch weitere Beispiele auch Probleme mit diesem Ansatz aufgezeigt. Zum einen können durch simple kürzeste Pfade zentrale Lernobjekte ignoriert werden, wenn diese in unterschiedlicher Reihenfolge von unterschiedlichen erfolgreichen Nutzern erlernt werden. Zum anderen werden bei Berücksichtigung aller Pfade in einigen Randfällen IPs berechnet, die thematisch irrelevante

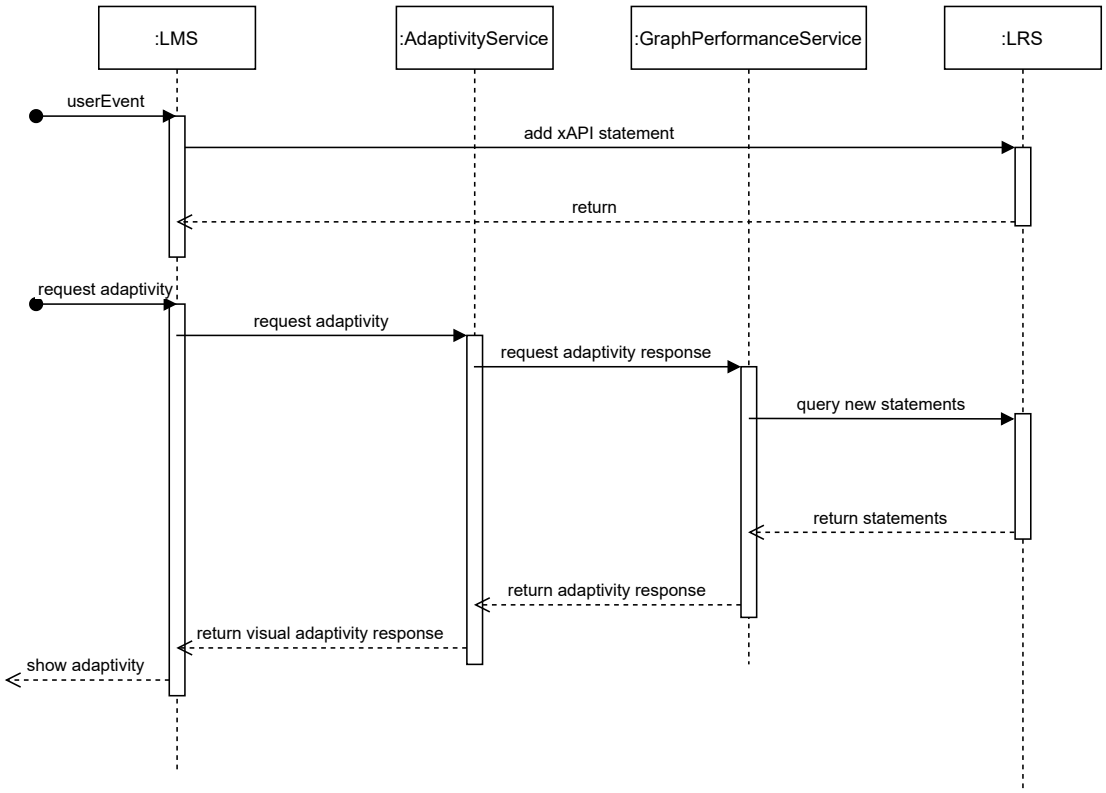


Abbildung 6.3: Aufrufhierarchie im Anwendungsfall. Der Nutzer der Lernsoftware kann nach dem Absolvieren von Lerneinheiten eine adaptivity response anfragen. Das LMS lädt dann eine visuelle Repräsentation der adaptivity response vom GraphPerformanceService durch einen dazwischenliegenden Service, der optional auch weitere Funktionen anbieten kann.

Lernobjekte enthalten.

In einem weiteren Schritt wurde auf Basis von Loh u. a. 2014 die Progress Performance durch eine Einschränkung der IPs auf die vom Nutzer absolvierten Lernobjekte zur Operationalisierung eingeführt.

Diese Arbeit hat sich mit den theoretischen Modellierungen und deren Verwendung befasst und daher zwar Tests mit synthetische Daten, nicht aber durch Nutzerstudien durchgeführt. Die Behebung der Probleme der IP-Berechnung sind ebenfalls nicht Teil der Arbeit. Vermutungen zur Behebung sind zum einen die Einbeziehung aller Aktivitäten, die alle erfolgreichen Nutzer von Teilen eines IP durchgeführt haben. Zum anderen könnte der Ausschluss von Pfaden nicht oder noch nicht erfolgreicher Nutzer das weitere Problem in Kombination mit der Performanceberechnung beheben. Ein weiterer offener Punkt ist die Gestaltung einer Metrik, die die Effektivität einzelner Lerneinheiten unabhängig vom Ziel misst. Es ist nicht ausgeschlossen, dass dafür eine manuelle Einschätzung des Erstellers nötig ist, möglicherweise kann dies auch automatisiert durch Metadaten der Durchführungen anderer Nutzer erfolgen.

Ein weiteres in dieser Arbeit nicht verfolgtes Problem mit echten Nutzerdaten kann sein, dass Nutzer mit mehr undokumentiertem Vorwissen als erwartet notwendige Lernobjekte überspringen. Damit werden kürzere IPs berechnet, als im IPM als Referenzmodell vorgesehen sind.

7 Fazit und Ausblick

Das übergeordnete Projektziel ist es, die Adaptivität von Lernsystemen zu verbessern. In dieser Arbeit wurden dazu das Konzept von Ideal Path Models (IPMs) betrachtet, das Beobachtungsdaten von Lernsystemen zur Analyse als Pfade in Graphen modelliert. Weiterhin wurde mithilfe der Ideal Paths (IPs) aus einer Modellierung für ein IPM und Graph Analytics ein Performancewert berechnet.

In einem ersten Schritt wurden verschiedene Modellierungen entwickelt. Dazu wurden die Beobachtungsdaten beschrieben und geklärt, welche Daten davon modelliert werden sollen. Weiterhin wurden diese dann in verschiedenen Varianten auf Knoten und Kanten in Graphen abgebildet. Die Varianten der Modellierung wurden dann von einem theoretischen Standpunkt analysiert, um die sinnvollste Modellierung zur praktischen Umsetzung zu finden. Diese verwendet Knoten für Nutzer und Aktivitäten und Kanten zwischen Nutzer und Aktivität sowie zwischen aufeinander folgenden Aktivitäten, um den Lernpfad als Pfad im Graphen darzustellen. Außerdem wurde beobachtet, dass dabei ein gerichteter, gewichteter Graph entsteht, der die Lernpfade der Nutzer enthält, aber nicht notwendigerweise azyklisch ist.

Zur Analyse der Beobachtungsdaten wurde weiterhin festgestellt, dass in dieser Modellierung beispielsweise mit Yen's k-Shortest-Path Algorithmus Pfade gefunden werden, die potentielle IPs sind. Außerdem wurde als Abwandlung der Jaccard Similarity und mithilfe von IPs eine Metrik zur Messung der Progress Performance entwickelt. Diese gibt an, wie effektiv ein Nutzer lernt, also wie viel seiner Lernobjekte prozentual keine unnötigen Umwege sind. Für einen einzelnen IP wird die Performance berechnet, indem der Anteil der Knoten und der Anteil der Kanten, die sich der Lernpfad mit dem IP teilt, gewichtet kombiniert wird. Der Nutzer erhält dann den IP mit der besten Performance als Empfehlung zusammen mit dieser Performance.

Zur Verifikation dieser Ergebnisse wurde ein Prototyp implementiert, der die genannte Modellierung in der Graphdatenbank neo4j umsetzt. Mit synthetischen Eingabedaten wurde festgestellt, dass die Verwendung von Yen's k-Shortest-Path Algorithmus die erwarteten IP finden kann. Probleme gibt es allerdings, wenn Lernpfade Aktivitäten in unterschiedlicher Reihenfolge durchlaufen. Dabei kann es passieren, dass ein notwendiges Lernobjekt im IP ausgelassen wird. Da die Daten aller Nutzer modelliert werden, werden auch Pfade von Nutzern gefunden, die das Lernziel noch nicht erreicht haben oder keine zielführenden Lernobjekte

nutzen. Daher kann es passieren, dass Nutzer den eigenen Lernpfad als Anfang des IP empfohlen bekommen und auf diesem Pfad eine sehr hohe Performance haben.

In weiterführender Forschung kann untersucht werden, ob die aufgezeigten Probleme durch Filtern der Eingabedaten oder der IPs behoben werden können. Die Verwendung anderer oder angepassterer Algorithmen kann ebenso noch untersucht werden. Außerdem wurde in dieser Arbeit angenommen, dass alle Lerneinheiten in ihrem Thema didaktisch gleich sinnvoll sind, in der Praxis können sich diese allein schon im Umfang stark unterscheiden. Daher wurde vorgeschlagen zu untersuchen, ob sich dieses Problem durch eine auf einzelnen Lerneinheiten definierte Metrik zu beheben lässt.

Literatur

Activity Streams 2.0 (Mai 2017). en. URL: https://www.w3.org/TR/2017/REC-activitystreams-core-20170523/ (besucht am 07. 02. 2022).

Alonso, José M. und Gabriella Casalino (2019). „Explainable Artificial Intelligence for Human-Centric Data Analysis in Virtual Learning Environments". en. In: *Higher Education Learning Methodologies and Technologies Online: First International Workshop, HELMeTO 2019, Novedrate, CO, Italy, June 6-7, 2019, Revised Selected Papers*. Hrsg. von Daniel Burgos u. a. Bd. 1091. Communications in Computer and Information Science. Cham: Springer International Publishing, S. 125–138. URL: http://link.springer.com/10.1007/978-3-030-31284-8 (besucht am 22. 11. 2021).

Alonso-Fernandez, Cristina u. a. (Apr. 2017). „Systematizing game learning analytics for serious games". In: *2017 IEEE Global Engineering Education Conference (EDUCON)*. ISSN: 2165-9567, S. 1111–1118.

Cavus, Nadire (Juni 2015). „Distance Learning and Learning Management Systems". en. In: *Procedia - Social and Behavioral Sciences*. The Proceedings of 6th World Conference on educational Sciences 191, S. 872–877. URL: https://www.sciencedirect.com/science/article/pii/S1877042815028712 (besucht am 02. 03. 2022).

Despotović-Zrakić, Marijana u. a. (2012). „Providing Adaptivity in Moodle LMS Courses". In: *Journal of Educational Technology & Society* 15.1. Publisher: International Forum of Educational Technology & Society, S. 326–338. URL: https://www.jstor.org/stable/jeductechsoci.15.1.326 (besucht am 08. 03. 2022).

Diestel, Reinhard (2017). *Graph theory*. New York, NY: Springer Berlin Heidelberg.

Experience API (xAPI) - MoodleDocs (2022). URL: https://docs.moodle.org/dev/Experience_API_(xAPI) (besucht am 07. 02. 2022).

Gonzalez-Barbone, Victor und Luis Anido-Rifon (Jan. 2010). „From SCORM to Common Cartridge: A step forward". en. In: *Computers & Education* 54.1, S. 88–102. URL: https://www.sciencedirect.com/science/article/pii/S0360131509001869 (besucht am 02. 03. 2022).

Green, Kathryn R. und Haynes L. Chewning (Mai 2020). „The Fault in our Systems: LMS as a Vehicle for Critical Pedagogy“. en. In: *TechTrends* 64.3, S. 423–431. URL: https://doi.org/10.1007/s11528-020-00480-w (besucht am 08. 03. 2022).

Grissinger, Alexander (Okt. 2020). „User Mo deling using Graph Analytics for Adaptivity in E-Learning Systems“. en. Magisterarb. Heidelberg: SRH University Heidelberg. URL: https://websites.fraunhofer.de/alexanderstreicher/wp-content/uploads/2020/12/MscGrissinger2020_GraphAnalyticsAdaptivity.pdf.

Hodler, Amy E. (2019). *O'Reilly Graph Algorithms Book*. en. URL: https://neo4j.com/graph-algorithms-book/ (besucht am 15. 11. 2021).

Innovationswettbewerb INVITE - BMBF (2022). de. URL: https://www.bmbf.de/bmbf/de/bildung/berufliche-bildung/foerderinitiativen-und-program-ur-staerkung-der-berufsbildung/innovationswettbewerb-invite/innovationswettbewerb-invite_node.html (besucht am 07. 03. 2022).

Loh, Christian Sebastian und Yanyan Sheng (Okt. 2014). „Maximum Similarity Index (MSI): A metric to differentiate the performance of novices vs. multiple-experts in serious games“. en. In: *Computers in Human Behavior* 39, S. 322–330. URL: https://www.sciencedirect.com/science/article/pii/S0747563214003963 (besucht am 03. 03. 2022).

Mangaroska, Katerina und Michail Giannakos (Okt. 2019). „Learning Analytics for Learning Design: A Systematic Literature Review of Analytics-Driven Design to Enhance Learning“. In: *IEEE Transactions on Learning Technologies* 12.4. Conference Name: IEEE Transactions on Learning Technologies, S. 516–534.

Pavlik Jr, Philip I, Hao Cen und Kenneth R Koedinger (2009). „Performance Factors Analysis – A New Alternative to Knowledge Tracing“. en. In: S. 8.

Pustovojtovskij, German (Feb. 2021). „Anwendung von Regressionsmodellen für adaptive digitale Lernspiele in der Bildauswertung“. de. Bachelor. Karlsruhe: Karlsruher Institut für Technologie (KIT). URL: https://websites.fraunhofer.de/alexanderstreicher/wp-content/uploads/2021/02/BScPustovojtovskij2021-PFA.pdf (besucht am 03. 10. 2022).

Raza, Syed A. u. a. (Apr. 2021). „Social Isolation and Acceptance of the Learning Management System (LMS) in the time of COVID-19 Pandemic: An Expansion of the UTAUT Model“. en. In: *Journal of Educational Computing Research* 59.2. Publisher: SAGE Publications Inc, S. 183–208. URL: https://doi.org/10.1177/0735633120960421 (besucht am 06. 03. 2022).

Shute, Valerie J. und Diego Zapata-Rivera (2012). „Adaptive Educational Systems“. In: *Adaptive Technologies for Training and Education*. Hrsg. von Paula J. Durlach und Alan M. Lesgold.

Cambridge: Cambridge University Press, S. 7–27. URL: https://www.cambridge.org/core/product/identifier/9781139049580%23c76903-1-1/type/book_part (besucht am 07. 03. 2022).

Shute, VJ und D Zapata-Rivera (2012). *Adaptive educational systems. Adaptive technologies for training and education, P. DURLACH, Ed.*

Sottilare, Robert A u. a. (2012). „A Modular Framework to Support the Authoring and Assessment of Adaptive Computer-Based Tutoring Systems (CBTS)". en. In: S. 13.

Streicher, Alexander, Julius Busch und Wolfgang Roller (2021). „Dynamic Cognitive Modeling for Adaptive Serious Games". In: *Adaptive Instructional Systems. Adaptation Strategies and Methods*. Hrsg. von Robert A. Sottilare und Jessica Schwarz. Lecture Notes in Computer Science. Cham: Springer International Publishing, S. 167–184.

Streicher, Alexander und Florian Heberle (2017). „Learning Progress and Learning Pathways". In: *Computer-Driven Instructional Design with INTUITEL*. Hrsg. von Kevin Fuchs und Peter A. Henning. River Publishers Series, S. 37–55. URL: https://www.riverpublishers.com/research_details.php?book_id=431.

Streicher, Alexander, Sebastian Leidig und Wolfgang Roller (2018). „Eye-Tracking for User Attention Evaluation in Adaptive Serious Games". In: *13th European Conference on Technology Enhanced Learning, EC-TEL 2018*. Leeds, UK: Springer, S. 583–586. URL: http://link.springer.com/10.1007/978-3-319-98572-5_50 (besucht am 21. 09. 2018).

Streicher, Alexander, Wolfgang Roller und Christian Biegemeier (2017). „Application of Adaptive Game-Based Learning in Image Interpretation". In: *11th European Conference on Game-Based Learning ECGBL 2017*. Graz, Austria: Academic Conferences and Publishing International Limited, S. 975–978.

Streicher, Alexander, Rainer Schönbein und Stefan Pickl (2021). „Graph-Based Modeling for Adaptive Control in Assistance Systems". In: *Advances in Artificial Intelligence, Software and Systems Engineering*. Hrsg. von Tareq Z. Ahram, Waldemar Karwowski und Jay Kalra. Lecture Notes in Networks and Systems. Cham: Springer International Publishing, S. 39–46.

Streicher, Alexander und Jan D Smeddinck (2016). „Personalized and Adaptive Serious Games". In: *Entertainment Computing and Serious Games: International GI-Dagstuhl Seminar 15283, Dagstuhl Castle, Germany, July 5-10, 2015, Revised Selected Papers*. Hrsg. von R. Dörner et al. Cham: Springer International Publishing, S. 332–377. URL: http://dx.doi.org/10.1007/978-3-319-46152-6_14.

xAPI-Spec (Dez. 2021). original-date: 2012-12-19T16:46:59Z. URL: https://github.com/adlnet/xAPI-Spec/blob/11196cf0d7afee50059ed7a9a2cbe6c72624fa3c/xAPI-About.md (besucht am 12. 12. 2021).

Yen, Jin Y. (Juli 1971). „Finding the K Shortest Loopless Paths in a Network“. In: *Management Science* 17.11. Publisher: INFORMS, S. 712–716. URL: https://pubsonline.informs.org/doi/abs/10.1287/mnsc.17.11.712 (besucht am 06. 03. 2022).

Abbildungsverzeichnis