

CollideLabs – Eine modulare Plattform zur Bereitstellung von Virtual Labs

Christian Peters*, Andreas Horst, Alexander Becker*, Plamen Dimitrov, Dennis Dittmann, Simon Erpenbach, Katerina Milina, Philipp Niehues*, Malte Wessel, Farnoush Zohourian, Tilman Göhnert**, Per Verheyen**, H. Ulrich Hoppe**

Abteilung für Informatik und Angewandte Kognitionswissenschaft
Universität Duisburg-Essen
Lotharstr. 63/65
47048 Duisburg
(vorname.nachname)@stud.uni-due.de
* (vorname.nachname)@uni-due.de
** (nachname)@collide.info

Abstract: Im Bereich der universitären Informatik-Ausbildung werden oftmals abstrakte Konzepte vermittelt. Der Ansatz der Virtual Labs bietet eine aktuelle Möglichkeit, computergestützte Simulationen verschiedener Konzepte leicht zugreifbar zur Verfügung zu stellen. Im Sinne des Prinzips „Lernen durch Handeln“ können Lernende die in den Labs repräsentierten Konzepte explorieren und verstehen. Der vorliegende Beitrag präsentiert mit Labs zu den Themenbereichen „Kognitive Modellierung mit Produktionsregelsystemen“ und „Wissensdiagnose mit Bayes-Netzen“ zwei beispielhafte Umsetzungen von Virtual Labs aus dem Bereich der Modellierung interaktiver Lehr- und Lernsysteme. Als Basis für die Implementierung dieser Labs dient dabei die CollideLabs Plattform, die einerseits einen Rahmen zur Präsentation von Virtual Labs darstellt und andererseits flexible Unterstützungsfunktionen für die Implementierung solcher Labs zur Verfügung stellt.

1 Einleitung

Im Bereich der universitären Informatik-Ausbildung werden oftmals abstrakte Konzepte vermittelt. Die Vermittlung solcher Konzepte erfolgt im Bereich der Naturwissenschaften bereits häufig durch das Experimentieren in Remote Labs, die einen Zugriff auf entfernte Versuchsapparaturen erlauben, oder in Virtual Labs, die das Experimentieren mit computergestützten Simulationen ermöglichen. Zahlreiche Plattformen [Con13b] fördern das aktive und selbstgesteuerte Lernen sowie das Aneignen komplexer Konzepte, indem sie praktische Erfahrungen mittels „Lernen durch Handeln“ in Virtual und Remote Labs ermöglichen. Im Rahmen von Projekten wie dem durch die EU geförderten Go-Lab¹ Projekt

¹ Go-Lab – <http://go-lab-project.eu>

wird die Relevanz dieser Ansätze [GCV⁺13] und deren Integration mit bestehenden didaktischen Konzepten [Con13a] deutlich.

In diesem Beitrag präsentieren wir die Lernplattform CollideLabs, die zum einen das Ziel verfolgt, Entwickler solcher Laboratorien in ihrer Arbeit zu unterstützen, und es zum anderen ermöglicht, Labs über eine gemeinsame webbasierte Nutzerschnittstelle anzubieten. Als Beispiele für im Rahmen dieser Plattform entwickelte Virtual Labs dienen dabei zwei Labs, die entwickelt wurden, um die Vorlesung „Gestaltung interaktiver Lehr-/Lernsysteme“ (GILLS) an der Universität Duisburg-Essen zu unterstützen.

Diese Veranstaltung richtet sich an Studierende der Masterstudiengänge Angewandte Kognitions- und Medienwissenschaft, Angewandte Informatik, Computer Engineering und Lehramt Informatik und vermittelt unter anderem Inhalte aus den Bereichen der kognitiven Modellierung, der intelligenten Tutorsystemen sowie der Wissens- und Fehlerdiagnose. Für das Sommersemester 2013 wurde diese Veranstaltung im Rahmen eines Blended Learning Ansatzes durch einen „Small Private Online Course“ (SPOC) [Fox13] ergänzt, der Microlernelemente und -aktivitäten, die sonst typisch für „Massive Open Online Courses“ (MOOCs) sind, aufgegriffen und durch Gruppenarbeit ergänzt hat. Zusätzlich zu der bereits vorher bestehenden Verwendung des Lernmanagement-Systems Moodle² zur Bereitstellung von Vorlesungsunterlagen und Übungsaufgaben wurden weitere Lernressourcen wie Videos, Selbst-Tests und Wikis integriert, wobei der Fokus der studentischen Gruppenarbeit auf der Erstellung von Wiki-Artikeln zu vorlesungsrelevanten Inhalten lag [HZH14]. An der konstruktivistischen Lerntheorie anknüpfend erhielten die Studierenden hierdurch die Möglichkeit des aktiven und selbst-gesteuerten Lernens im problemorientierten Kontext [Mai04].

Im Folgenden werden zunächst Grundlagen und verwandte Arbeiten präsentiert. Anschließend werden im Rahmen dieses Beitrags Virtual Labs zu den Themenbereichen „Kognitive Modellierung mit Produktionsregelsysteme“ und „Wissensdiagnose mit Bayes-Netzen“ sowie die CollideLabs Plattform vorgestellt, mit der Studierenden in Zukunft weitere Möglichkeiten angeboten werden, abstrakte Konzepte durch aktives und selbstgesteuertes Lernen zu erfahren.

2 Grundlagen und verwandte Arbeiten

Webbasierte Labs können gemäß Govaerts et al. [GCV⁺13] in Remote Labs und Virtual Labs untergliedert werden, die über das Internet verfügbar sind. Während Remote Labs den Zugriff auf entfernte physikalische Versuchsaapparaturen ermöglichen, können Virtual Labs die realistische Simulation dieser und abstrakter Konzepte ermöglichen, wie sie auch in der Modellierung interaktiver Lehr- und Lernsysteme Verwendung finden. De Jong und Kollegen schlussfolgern, dass Remote Labs und Virtual Labs nicht nur einen gleichwertigen Ersatz für physikalische Versuchsaapparaturen bilden, sondern dass mit Virtual Labs auch die Möglichkeit existiert, Lernenden zusätzliche Kontextinformationen zur Verfü-

² Moodle – <http://moodle.de>

gung zu stellen und die Salienz wichtiger Zusammenhänge zu erhöhen [dJLZ13]. Ein Beispiel für die erfolgreiche Umsetzung eines Virtual Labs repräsentiert Optilab [HBBP07], mit dem Lernende unter anderem das Konzept der Lichtbrechung erlernen können.

Zusätzlich zu der Manipulation von Gegenständen und Lichtquellen werden schematische Kontextinformationen zur derzeitigen Anordnung visualisiert. Aktives Handeln und das Explorieren des Lerngegenstands steigert gemäß der Selbstbestimmungstheorie der Motivation nach Edward L. Deci und Richard M. Ryan den durch den Lernenden wahrgenommenen Grad der Selbstbestimmung [DR85]. Das resultierende Gefühl der Autonomie und die Kompetenzerfahrung stehen in engem Zusammenhang mit der Entstehung lernförderlicher intrinsischer Motivation. Lernen durch Handeln kann sich so aufgrund einer Steigerung der Lernmotivation positiv auf den Lernerfolg auswirken.

Die Vorlesung GILLS hat mit der Adaption des Lehrkonzeptes im vergangenen Jahr bereits Maßnahmen zur Einbindung der Teilnehmenden getroffen. So wurden im Rahmen der bereitgestellten Moodle Plattform Werkzeuge angeboten, um Wissen mit Hilfe von Wikis zu strukturieren und zu teilen. Dabei werden typische Elemente von xMOOCs und cMOOCs im Rahmen eines Blended Learning Ansatzes integriert. Während Studierende in xMOOCs vorwiegend vorgegebene Lerninhalte erhalten, sind cMOOCs dadurch geprägt, dass Lernende aktiv Lerninhalte produzieren und miteinander teilen [HZH14]. In Anlehnung an Brunning et al. wird so der traditionelle Frontalunterricht vermieden und dem Lernenden eine aktive Rolle zur Mitgestaltung der Vorlesung übertragen [BSRG99]. Das im Rahmen dieses Beitrags vorgestellte CollideLabs soll in Zukunft von der bestehenden Moodle Lernplattform aus erreichbar sein und den Studierenden die Möglichkeit bieten, einige der in der Vorlesung betrachteten abstrakten Konzepte experimentell zu erfahren.

3 CollideLabs

CollideLabs³ ist eine Lernplattform, die zum einen das Ziel verfolgt, im Sinne eines technischen Frameworks Entwickler von Virtual Labs in ihrer Arbeit zu unterstützen, und es zum anderen ermöglicht, verschiedene erstellte Labs über eine gemeinsame webbasierte Nutzerschnittstelle anzubieten. Im Folgenden werden die beiden implementierten Virtual Labs zu den Themen „Kognitive Modellierung mit Produktionsregelsystemen“ und „Wissensdiagnose mit Bayes-Netzen“ und die zur Bereitstellung verwendete modulare Systemarchitektur im Detail vorgestellt.

3.1 Kognitive Modellierung mit Produktionsregelsystemen

Die Modellierung mit Produktionsregeln stellt für die Studierenden aufgrund der hohen Komplexität und Abstraktion eine große Herausforderung dar. Um Produktionsregelsysteme für die Studierenden praktisch und interaktiv erfahrbar zu machen, wurde ein Lab zu diesem Thema entwickelt.

³ CollideLabs – <http://collidelabs.collide.info>

Die „Kognitive Modellierung mit Produktionsregelsystemen“ stützt sich auf die Cognitive Complexity Theory (CCT), einen nicht-empirischen Evaluierungsansatz von Kieras & Polson [KP85] zur Untersuchung der kognitiven Komplexität auf Benutzerseite bei der Durchführung einer Tätigkeit mit einem Gerät.

CCT unterscheidet die zwei Komponenten Benutzersimulation und Gerätesimulation. Diese haben jeweils einen eigenen Interpretier. Die Benutzersimulation setzt sich aus dem Arbeitsgedächtnis und den Produktionsregeln zusammen. Das Arbeitsgedächtnis beinhaltet eine im Verlauf der Simulation variable Anzahl an Informationseinheiten. Die Produktionsregeln repräsentieren das prozedurale Wissen des Menschen über die Aufgabe und somit ein mentales Modell des Menschen über die interne Struktur und Funktionsweise des zu verwendenden Gerätes. Die zweite Komponente, die Gerätesimulation, besteht aus dem formalen Modell des Gerätes, welches das Gerät korrekt und vollständig beschreiben soll. Das formale und das mentale Gerätemodell sind zu trennen, da das mentale Modell unvollständig sein kann. Das formale Gerätemodell kann in Form eines generalisierten Zustands-Übergangsgraphen dargestellt werden [KP83]. Als letztes Element fließt die Aufgabenspezifikation in die Benutzersimulation mit ein. Diese kann zum Beispiel als Startbedingung im Arbeitsgedächtnis abgebildet werden [BKP90].

Durch diesen Aufbau ist es möglich, ein CCT Modell zu simulieren und dadurch statistische Kennwerte zu erheben, um Vorhersagen über Gedächtnisbelastung (Memory Load), Ausführungszeit (Completion Time) oder Aufgabenkomplexität (Difficulty) abzuleiten. Diese beziehen sich auf die kognitive Last, die bei einem virtuellen Benutzer auftritt und mit deren Hilfe Vorhersagen über die kognitive Komplexität einer Aufgabe gebildet werden können. Die Gedächtnisbelastung wird abgeschätzt durch die Anzahl der Einträge im Arbeitsgedächtnis. Die Ausführungszeit wird abgeschätzt durch die Anzahl der Schritte, die nötig sind, um eine Aufgabe abzuschließen. Dabei können einzelne Regeln auch mehrmals ausgeführt werden. Die Aufgabenkomplexität wird abgeschätzt durch die Anzahl aller unterschiedlichen Regeln, die zur Lösung einer Aufgabe benötigt werden.

Diese Ergebnisse können dann beispielsweise in Verbindung mit Millers The Magical Number Seven, Plus or Minus Two-Theorie [Mil56] interpretiert werden. Millers Theorie zeigt auf, dass der Mensch nur etwa 7 (± 2) Informationseinheiten (Chunks) in seinem Kurzzeitgedächtnis speichern kann. Eine Aufnahme einer weiteren Informationseinheit würde das Vergessen einer bestehenden erfordern und damit zu einer höheren Fehlerquote führen.

3.1.1 Beschreibung des Virtual Lab

Durch die Möglichkeit, eigene Nutzer- und Gerätemodelle zu erstellen und zu simulieren beziehungsweise auf bestehende zurückzugreifen, bietet das CCT Lab die Möglichkeit, mit Produktionsregelsystemen interaktiv zu experimentieren und greift damit das Konzept „Lernen durch Handeln“ auf. Die Nutzerschnittstelle (siehe Abbildung 1) zeigt im oberen Bereich die Statistik-Komponente und darunter in der linken Spalte die Produktionsregeln. Die rechte Spalte zeigt das Arbeitsgedächtnis, das Gerätemodell und den Gerätekonfigurator.

Die Benutzersimulation-Komponente besteht aus dem Arbeitsgedächtnis, den Produktionsregeln und einem im Hintergrund arbeitenden Interpreter für die Produktionsregeln.

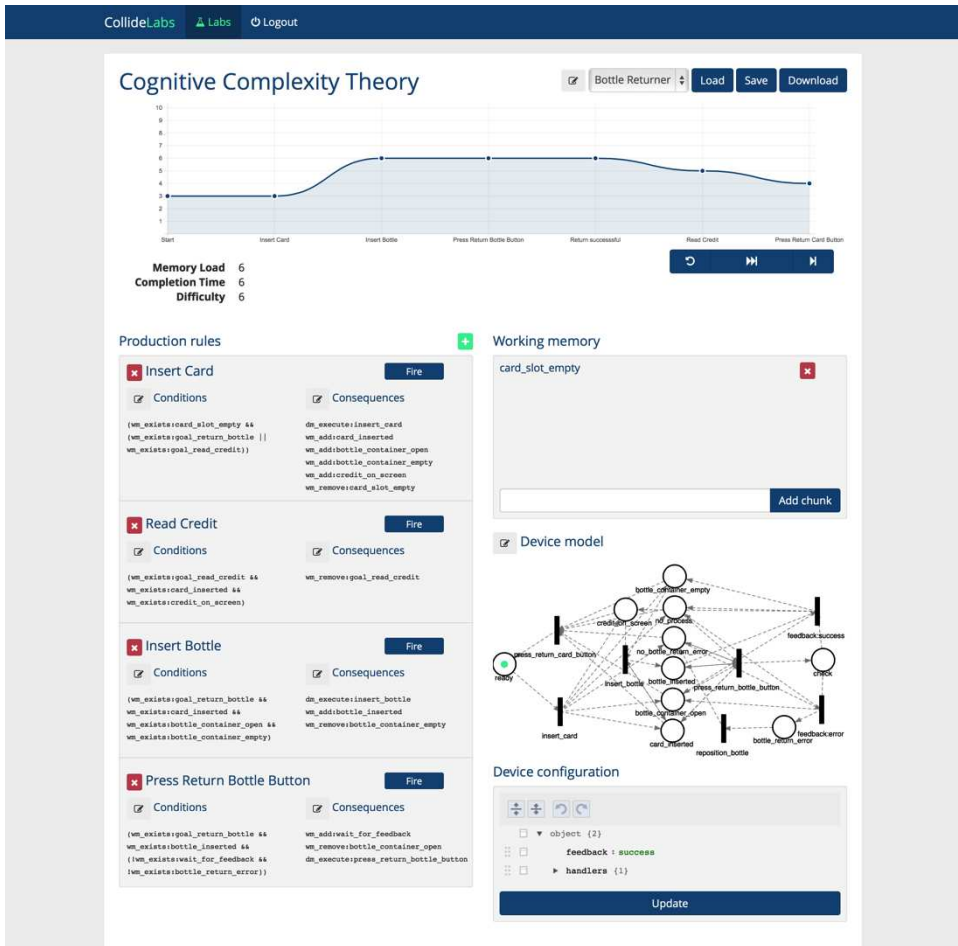


Abbildung 1: Darstellung des CCT Virtual Labs

Die Arbeitsgedächtniseinträge können sowohl vom Anwender des Virtual Labs als auch automatisiert durch die Simulation in Folge des Eintritts einer Produktionsregel hinzugefügt oder entfernt werden. Eine Produktionsregel besteht stets aus einem Bedingungen-Teil und einem Anweisungen-Teil. Im ersten Teil werden die Bedingungen für die Ausführung einer Regel festgelegt, während im zweiten Teil die Anweisungen definiert sind, die bei Erfüllung aller Bedingungen ausgeführt werden. Bei Bedingungen handelt es sich um Arbeitsgedächtniseinträge, welche existieren müssen (z.B. `if (wm_exists:Eintrag_1 && wm_exists:Eintrag_2)`). Bei den Anweisungen kann es sich um das Hinzufügen oder Entfernen eines Eintrags im Arbeitsgedächtnis handeln oder um Änderungen des Gerätezustandes (z.B. `then wm_remove:Eintrag_1`). Bei der automatischen Ausführung prüft der Regel-Interpreter, ob die Bedingungen zur Ausführung einer Regel erfüllt sind. Ist dies der Fall, wird die Änderung im Arbeitsgedächtnis durchgeführt, die feuere Regel für

den Anwender graphisch markiert und die Änderung des Gerätemodells an den Gerätemodell-Interpreter übergeben.

Zur Lösung eines Zustands, in dem multiple Produktionsregeln seitens des Regel-Interpreters simultan ausgeführt werden können, besteht die Option, Produktionsregeln ein Prioritätsmaß hinzuzufügen. Anderenfalls erfolgt die randomisierte Auswahl durch den Regel-Interpreter.

Die Gerätesimulation-Komponente setzt sich aus einem Gerätemodell und einem Gerätemodell-Interpreter zusammen. Das Gerätemodell besteht aus Zuständen und Übergängen, welche das vom simulierten Benutzer verwendete Gerät formal beschreiben. Es wird als Petri-Netz graphisch dargestellt. Anhand dieser Visualisierung kann der Anwender die Simulation aus Geräteperspektive verfolgen. Ein Gerätemodell-Interpreter prüft, ob eine Transition des Petri-Netzes ausgeführt werden kann. Des Weiteren gibt es die Möglichkeit, Gerätekonfigurationen über die Benutzerschnittstelle vorzunehmen. Die Statistik-Komponente gibt die Metriken Gedächtnisauslastung, Ausführungszeit und Aufgabenkomplexität wieder, welche im Verlauf der Simulation aufgezeichnet werden. Der Verlauf der Gedächtnisauslastung wird in einem Graphen dargestellt.

3.1.2 Vermittelter Inhalt und Einsatzszenarien

Die Simulation erfolgt entweder vollautomatisch, schrittweise oder durch den Anwender gesteuert: Bei vollautomatischer Simulation werden die Produktionsregeln in relativ schneller Abfolge auf Ausführbarkeit überprüft und solange durchgeführt, bis alle Möglichkeiten erschöpft sind. Dieser Modus ist besonders für die Vorhersage der kognitiven Komplexität geeignet. Beim schrittweisen Modus wird nach Benutzereingabe die jeweils nächste Regel ausgeführt. Dieser Modus ist besonders geeignet, um eine CCT-Simulation nachzuvollziehen oder eigene, selbstverfasste Szenarien zu überprüfen. Steuert der Anwender die Ausführung der Simulation selbst, hat er die Möglichkeit, zu jeder Zeit jede Regel auszuwählen. Das Virtual Lab überprüft die gewählte Regel auf Ausführbarkeit. Ist die Regel im aktuellen Zustand von Arbeitsgedächtnis und Gerätemodell ausführbar, wird sie ausgeführt, andernfalls wird eine visuelle Rückmeldung gegeben. Auf Grundlage des Prinzips „Lernen durch Handeln“ wird hier durch die starke Interaktion ein im Vergleich höherer Lernerfolg vermutet.

Dem Benutzer stehen initial zwei Szenarien zur Verfügung. Zum Ersten wird die Bedienung eines Flaschenrückgabeautomaten modelliert. Zum Zweiten wird eine Kaffeemaschine modelliert, welche sowohl Kaffee als auch Milchkaffee erstellen kann. Für letzteres sind mehr Schritte (z.B. das Anschließen eines Milchtanks) erforderlich, sodass die unterschiedlichen Komplexitäten beider Aufgaben miteinander verglichen werden können.

Alle Szenarien können mit einem Editor beliebig editiert werden. Darüber hinaus können neue Szenarien erstellt werden, zum Beispiel im Rahmen einer Übungsaufgabe in der Vorlesung. Nutzer können Szenarien serverseitig laden und speichern. Darüber hinaus besteht die Möglichkeit, die Szenarien lokal herunterzuladen und über die Eingabe des Quelltextes in ein dafür vorgesehenes Eingabefeld wieder zu laden. Auch ein Petrinetz für ein Gerätemodell kann über ein Eingabefeld in Quelltextform geladen werden. Eine integrierte

Dokumentationsseite und kontextabhängige Tooltips unterstützen den Anwender bei der Verwendung des Virtual Labs.

3.2 Wissensdiagnose mit Bayes-Netzen

Das zweite Thema der GILLS Veranstaltung, welches in Zukunft mit Hilfe eines Virtual Lab für die Studierenden besser experimentell erfahrbar sein soll, ist die „Wissensdiagnose mit Bayes-Netzen“. Das entsprechende Lab hat die Aufgabe, dem Nutzer die Einsicht in die Arbeit eines intelligenten Tutorsystems (ITS) zu vermitteln, das dynamische Bayes-Netze benutzt [HLP97].

Bayes-Netze sind azyklische Graphen, die dazu dienen, die Wahrscheinlichkeit von voneinander abhängigen Ereignissen zu berechnen. Dynamische Bayes-Netze werden benutzt, um diese Wahrscheinlichkeiten über einen bestimmten Zeitraum zu berechnen und dabei das Netz zu aktualisieren. Dynamische Bayes-Netze bestehen aus zwei Komponenten: Evidenzen und latenten Variablen. Evidenzen sind beobachtbare Variablen, latente Variablen sind nicht direkt beobachtbar. Das Netz ist so aufgebaut, dass Evidenzen an eine latente Variable gekoppelt sind. Im Sinne eines Bayes-Netzes heißt dies, es wird die Wahrscheinlichkeit berechnet, dass eine latente Variable eintritt unter der Bedingung, dass bestimmte Evidenzen vorher positiv oder negativ eingetroffen sind. Da Bayes-Netze somit die Möglichkeit bieten, die latenten Variablen auf Basis der beobachteten Evidenzen abzuschätzen, können sie zur Wissensdiagnose eingesetzt werden. Hierzu werden beobachtbare Informationen als Evidenzen genutzt, wie z. B. die von Lernenden abgegebenen Lösungen und deren Bewertung oder die auf dem Weg zur Lösung gemachten Zwischenschritte, um nicht direkt beobachtbare Variablen abzuschätzen, wie z.B. die Fähigkeiten des Lernenden oder die Wahrscheinlichkeit, bestimmte Fehler in Zukunft zu machen. Ein darauf basierendes ITS arbeitet nach folgenden Schritten: Beobachten der Evidenzen, Berechnen der latenten Variablen und Auswahl und Ausgabe der passenden Hilfestellungen.

3.2.1 Beschreibung des Virtual Lab

Das Virtual Lab soll es dem Nutzer ermöglichen, nachzuvollziehen, wie die Wissensdiagnose mit Bayes-Netzen auf Basis von Lerneraktionen im Rahmen eines intelligenten Tutorsystems durchgeführt werden kann und welche Mechanismen dahinter stehen. Dazu wird im Rahmen des Labs zum einen ein einfaches ITS angeboten, in welchem der Labnutzer die Aktionen eines Lernenden simuliert, und zum anderen werden die sich daraus ergebenden Evidenzen, deren Eingang in das Bayes-Netz und die sich daraus ergebenden Schätzungen der latenten Variablen visualisiert (siehe Abbildung 2).

Als Aufgabentyp für das ITS werden im Rahmen des Labs Aufgaben zu linearen Gleichungen verwendet. Als latente Variablen, also zu diagnostizierendes Wissen, sind das Formelverständnis und das visuelle Verständnis gewählt worden. Der Nutzer des Virtual Labs simuliert in der Aufgabenansicht den Lernenden, indem er die vom ITS gestellten Aufgaben löst. Das Virtual Lab visualisiert die durch das ITS erfassten Aufgabenindikatoren und berechnet die resultierenden beobachtbaren Evidenzen. In der Diagnoseansicht

leitet der Nutzer anschließend auf Basis der Evidenzen die Vorhersage der latenten Variablen ab. Es folgt die Visualisierung des zugrundeliegenden dynamischen Bayes-Netztes und des Vorhersage-Verlaufes.

CollideLabs
Logout

Knowledge Diagnosis with Bayes Nets

Task View
Diagnosis View
Documentation

Task

Given are the points $P_{f1}(1, 3)$ and $P_{f2}(3, 5)$ of the function f . Calculate the slope m_f , intercept b_f , and zero n of f .

Given is also function g with the point $P_{g1}(1, 2)$ and slope $m_g = 2$. Plot the points in the interval $[-4; 4]$ and calculate the intersection of functions f and g .

Solution area

Intersection of functions f and g Ask tutor

Calculate the intersection of functions f and g

Equation for x : $x = (bg - bf) / (mf - mg)$

Insert the values: $x = (1 \cdot 2 - 3) / (3 - 2)$

Simplify: $x = (-1) / (1)$

Value of x : $x = -1$

✓ Fill correct answer
✗ Fill incorrect answer

← Prev step
Transfer solution
Next step →

Observation of evidence variables

Indicator	Variable	Meaning
●	$m_f = 1$	I1 correctness of slope m
●	$b_f = 3$	I2 correctness of y-intercept b
●	$x = -2$	I3 correctness of zero x
●	$s = ?$	I4 correctness of intersection s
Evidence E1: Formula Task Solution:		?
Evidence E2: Formula Tutor Intervention:		False
Indicator	Variable	Meaning
●	15	y-value associated with each x
●	16	each x-value associated with exactly one y-value
●	17	strict positive or negative monotony
●	18	same slope triangle
Evidence E3: Visual Task Solution:		?
Evidence E4: Visual Tutor Intervention:		False

Task View
Diagnosis View
Documentation

Estimate of Formula Understanding

Bayes Net

PFU: Previous Formula Understanding: Fair

Evidence E1: Formula Task Solution: Excellent

Evidence E2: Formula Tutor Intervention: False

E1: Formula Task Solution

E2: Formula Tutor Intervention

Previous Page
Next Page

Probability Table

Filter Previous FU
Fair (50%)

Previous FU	Input		Output
	E1	E2	Estimated FU
Fair	Very Poor	True	Very Poor
Fair	Very Poor	False	Poor
Fair	Poor	True	Poor
Fair	Poor	False	Poor
Fair	Fair	True	Fair
Fair	Fair	False	Fair
Fair	Good	True	Fair
Fair	Good	False	Good
Fair	Excellent	True	Fair
Fair	Excellent	False	Good

Please select the estimated FU by clicking the according table row.

P(FU | PFU, E1, E2) = **Good**

Transfer Estimation

Abbildung 2: Zusammengefasste Darstellung des Bayes Virtual Labs

3.2.2 Vermittelter Inhalt und Einsatzszenarien

Das Virtual Lab soll das Verständnis fördern, wie ein Bayes-Netz basiertes Tutorsystem seine Daten ermittelt und damit seine Handlungen ausführt. Es wurde für Studierende der GILLS Vorlesung entwickelt, damit diese den Vorlesungsinhalt praktisch erfahren können. Dabei steht das Eingabe-Ausgabe Verhalten im Vordergrund, also dass der Nutzer sehen kann, welche Auswirkungen die Eingabe von Schülerlösungen, z.B. mit gezielt eingestreuten Fehlern, auf die Ausgabe und damit die Werte des Systems hat. Dies wird dadurch erreicht, dass gleichzeitig ein Tutor und ein Evidenzbereich dargestellt werden, wobei der Evidenzbereich nach jeder Eingabe des Nutzers aktualisiert wird. Hierdurch wird der direkte Zusammenhang zwischen Ein- und Ausgabe deutlich. Der Evidenzbereich stellt den Einfluss von Teil- und Komplettlösungen dar, wodurch die Details der Berechnung gut hervorgehoben werden. Das Virtual Lab kann dazu genutzt werden, dem Nutzer praktische Erfahrungen zu vermitteln. Dadurch, dass der Nutzer selbst die Schülerlösung eingibt, kann er gezielt eigene Fehler einstreuen und damit bestimmte Auswirkungen auf die Inferenz Berechnung erzeugen. Es ist also nicht nur möglich, Beobachtungen über gegebene Szenarien von Schülerlösungen anzustellen, sondern auch eigene Szenarien für verschiedene Schülertypen zu erstellen.

3.2 Systemarchitektur

CollideLabs ist ein flexibles Framework für Virtual Labs, welches als Multi-Agenten-System konzipiert worden ist. Das System differenziert zwischen der webbasierten, anwenderseitigen Nutzerschnittstelle und den ausführenden, serverseitigen Agenten der modularen Virtual Labs. Als Kommunikationsplattform zwischen der Nutzerschnittstelle und den Labs dient ein SQLSpaces Server [Wei12]. Die SQLSpaces sind eine Implementierung des Tuple Space Konzepts [Gel85], die besonderen Wert auf eine einfache Unterstützung sprach-heterogener Multiagentensysteme legen. Dadurch können im Rahmen der CollideLabs die Virtual Lab-Agenten problemspezifisch in der jeweils optimalen Programmiersprache wie Java, Prolog, Python oder C# umgesetzt werden. Einzelne Labs sind voneinander unabhängig, verfügen jedoch zusammen mit dem CollideLabs Framework über eine gemeinsame serverseitige Datenbank zum flexiblen Datenaustausch. Abbildung 3 zeigt den schematischen Aufbau der CollideLabs Architektur, bestehend aus einem Frontend Framework zur Darstellung der System-Schnittstelle und labspezifischer Nutzeroberflächen sowie einen Backend Server zur Bereitstellung von Mediator-Funktionen zur Realisierung der Kommunikation zwischen Frontend und Backend. Die in der Architekturübersicht dargestellten Virtual Labs skizzieren in Anlehnung an die in diesem Beitrag vorgestellten Labs die schematische Labstruktur.

Die CollideLabs Architektur basiert auf bewährten Webtechnologien rund um das serverseitige und Javascript-basierte NodeJS⁴. Als ein Vorteil von NodeJS gilt die modulare Erweiterbarkeit durch frei verfügbare Pakete. Hierzu zählen unter anderem durch CollideLabs verwendete Pakete zur Strukturierung der Applikation, der Verwaltung von Nutzeraccounts und der Bereitstellung von Logging-Schnittstellen.

⁴ NodeJS – <http://nodejs.org>

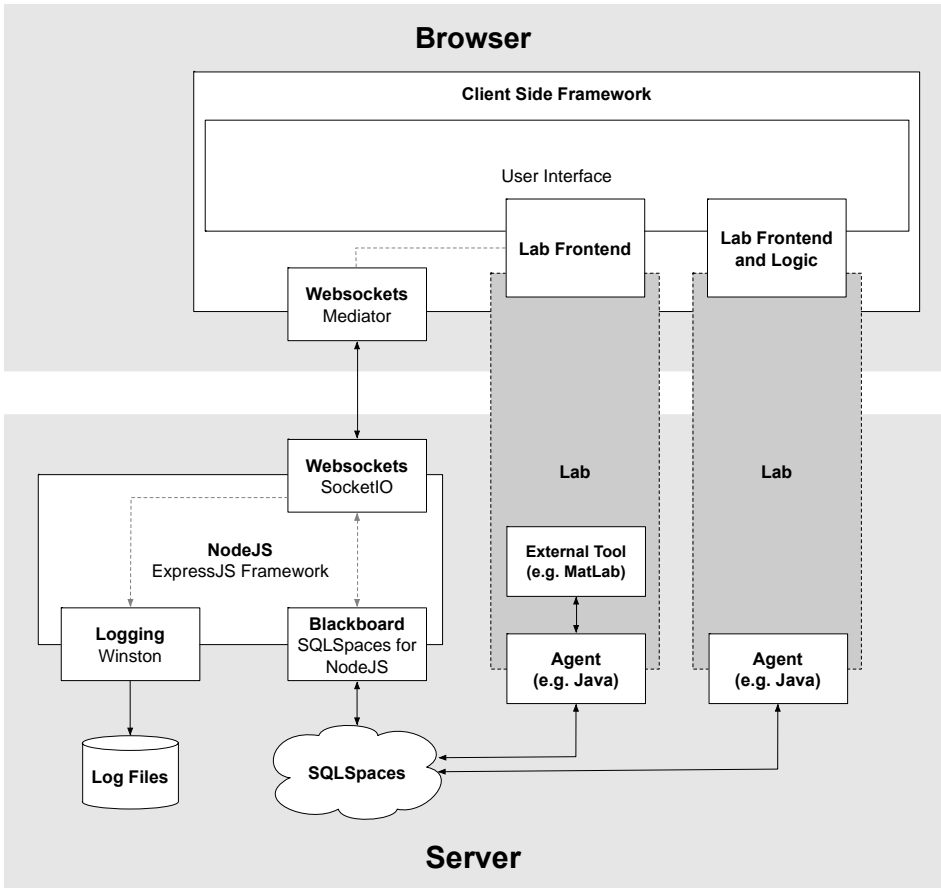


Abbildung 3: Schematische Darstellung der CollideLabs Architektur.

Virtual Labs können in der CollideLabs Architektur als Plug-Ins verstanden werden, die aus zwei Komponenten bestehen. Das Lab Frontend basiert auf Javascript-Bibliotheken zur Realisierung der Synchronität zwischen dem Frontend und dem Backend und erlaubt es, die Funktionalität mittels Javascript zu integrieren. Das optionale Lab Backend kann in Form eines Agenten in einer der zuvor gelisteten, mit SQLSpaces kompatiblen Programmiersprachen entwickelt werden und so optimal zur Problemlösung angepasst werden. Die Erstellung eines neuen Virtual Labs erfordert die Beteiligung eines Entwicklers. Je nach bestehendem Virtual Lab kann eine Erweiterung um zusätzliche Szenarien auch von Dozenten ohne explizite Kenntnisse in der Softwareentwicklung vorgenommen werden.

Im Falle des Labs zur „Kognitiven Modellierung mit Produktionsregelsystemen“, welches dem rechten Lab in der Architekturübersicht entspricht, übernimmt das Frontend nahezu alle Aufgaben des Labs, sowohl die Nutzerschnittstelle als auch die in JavaScript implementierte Logik. Der Agent im Backend dient nur dem serverseitigen Speichern und Laden von Szenarioinformationen. In einer ersten Version des Labs zur „Wissensdiagnose

mit Bayes-Netzen“ dient das Frontend ausschließlich als Nutzerschnittstelle, während die Berechnung des Bayes-Netzes mit MatLab⁵ im Backend erfolgt. Diese Berechnung wurde in der aktuellen Version in das Frontend verlagert.

Die Kommunikation der einzelnen Bestandteile erfolgt mittels einer Schnittstelle (Mediator) auf dem Webserver. Auf Basis der SQLSpaces werden Informationen als Tupel gespeichert, die von allen Einzelkomponenten abrufbar sind und zur Weiterverarbeitung bereit stehen. Der CollideLabs Mediator implementiert drei Strategien zur Kommunikation zwischen dem Lab-Frontend und dem Lab-Backend: Das Anfordern von Informationen aus dem Backend und deren Weiterverarbeitung im Frontend (request/response pattern), das Ausführen von Befehlen im Backend Agenten (command pattern) und das Ausführen von Befehlen im Frontend (event pattern).

Die CollideLabs Systemarchitektur verfügt über die Möglichkeit, System- und Nutzeraktivitäten in Form erweiterbarer Log-Dateien zu speichern. Während Systemlogs lediglich der System- und Fehleranalyse dienen, ermöglichen Nutzerlogs das Festhalten der Interaktion der Lernenden mit dem System. Die Logging-Schnittstelle kann im Frontend über den Mediator und im Backend mittels SQLSpaces angesprochen werden. Aktivitäten werden auf Basis des ActivityStreams-Standards⁶ festgehalten. Die bereits integrierten Virtual Labs halten so Informationen zu ausgeführten Aktionen fest, welche seitens des Systems um Informationen zum Benutzer, dem Virtual Lab und der aktuellen Zeit ergänzt werden.

4 Zusammenfassung und Ausblick

CollideLabs ist eine modulare Lernplattform, die es dem Lernenden auf Basis des „Lernen durch Handeln“ ermöglicht, aktiv Konzepte aus dem Bereich der Modellierung interaktiver Lehr- und Lernsysteme zu erlernen. Die entstandene Architektur ist aufgrund der Modularität des Virtual Lab Ansatzes auf beliebige Lerngegenstände erweiterbar und ermöglicht durch den Einsatz der SQLSpace Kommunikationssystem die zukünftige Entwicklung weiterer Systemkomponenten. Die integrierte Schnittstelle zum Loggen von Nutzerdaten kann zukünftig als Basis dafür dienen, weiterführende Analysen des Lernerhaltens in der Plattform durchzuführen. Insbesondere die Integration bestehender Analyse-systeme wie der Analytics Workbench [GZVH14] kann CollideLabs um sinnvolle Funktionen zur Analyse ergänzen.

Im Bereich der vorgestellten Labs sind insbesondere Erweiterungen für das Bayes Virtual Lab denkbar. Beispielsweise könnten weitere Szenario-Parameter wie das Bayes-Netz und die Aufgaben vom Nutzer variiert werden, um mehr Möglichkeiten zum Experimentieren zu geben. CollideLabs und die integrierten Virtual Labs kommen im Rahmen der Vorlesung „Gestaltung interaktiver Lehr-/Lernsysteme“ an der Universität Duisburg-Essen im Sommersemester 2014 erstmalig zum Einsatz. Erste Analysen der andauernden Evaluation deuten auf eine hohe Akzeptanz und häufige Interaktion mit den Virtual Labs der CollideLabs Plattform durch Studierende hin.

⁵ MatLab – <http://mathworks.de/products/matlab>

⁶ Activity Streams – <http://activitystrea.ms>

Literaturverzeichnis

- [BKP90] S. Bovair, D. E. Kieras und P. G. Polson. The acquisition and performance of text-editing skill: A cognitive complexity analysis. *Human-Computer Interaction*, 5(1), 1-48, März 1990.
- [BSRG99] R. H. Bruning, G. J. Schraw, R. R. Ronning und J. A. Glover. *Cognitive Psychology and Instruction*. Merrill Upper Saddle River, NJ, 1999.
- [Con13a] The Go-Lab Consortium. D1.2 Curriculum analyses (M12). Deliverable, Go-Lab Project (FP7/2012-2016 grant agreement no. 317601), 2013.
- [Con13b] The Go-Lab Consortium. D2.1 Appendix 1: Analysis of Existing Repositories and Federations of Online Labs (M12). Deliverable, Go-Lab Project (FP7/2012-2016 grant agreement no. 317601), 2013.
- [djLZ13] T. de Jong, M. C. Linn und Z. C. Zacharia. Physical and virtual laboratories in science and engineering education. *Science*, 340(6130), 305–308, 2013.
- [DR85] E. L. Deci und R. M. Ryan. *Intrinsic motivation and self-determination in human behavior*. Plenum, 1985.
- [Fox13] A. Fox. From MOOCs to SPOCs. *Communications of the ACM*, 56(12), 38–40, Dezember 2013.
- [GCV+13] S. Govaerts, Y. Cao, A. Vozniuk, A. Holzer, D. G. Zutin, E. S. C. Ruiz, L. Bollen, S. Manske, N. Faltin, C. Salzmänn et al. Towards an Online Lab Portal for Inquiry-Based STEM Learning at School. In *Advances in Web-Based Learning—ICWL 2013*, 244–253. Springer, 2013.
- [Gel85] D. Gelernter. Generative communication in Linda. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 7(1), 80-112, Januar 1985.
- [GZVH14] T. Göhnert, S. Ziebarth, P. Verheyen und H. U. Hoppe. Integration of a flexible analytics workbench with a learning platform for medical specialty training. In *Proceedings of the Workshop „Computational Approaches to Connecting Levels of Analysis in Networked Learning Communities“* co-located with 4th International Conference on Learning Analytics and Knowledge (LAK 2014), Indianapolis, Indiana, USA, März 2014.
- [HBBP07] E. Hatzikraniotis, G. Bisdikian, A. Barbas und D. Psillos. Optilab: Design and development of an integrated virtual laboratory for teaching optics. In *Proceedings of the 7th International Conference on Computer Based Learning in Science*. Crete: Technological Educational Institute of Crete, 2007.
- [HLP97] M. Helander, T. K. Landauer und P. Prabhu. *Handbook of Human-Computer Interaction*. Elsevier Science B. V., second, completely revised edition, 1997.
- [HZH14] T. Hecking, S. Ziebarth und H. U. Hoppe. Analysis of Dynamic Resource Access Patterns in a Blended Learning Course. In *LAK'14: Proceedings of the Fourth International Conference on Learning Analytics and Knowledge*, New York, NY, USA, 2014. ACM.
- [KP83] D. Kieras und P. G. Polson. A Generalized Transition Network Representation for Interactive Systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '83*, Seiten 103–106, New York, NY, USA, 1983. ACM.
- [KP85] D. Kieras und P. G. Polson. An approach to the formal analysis of user complexity. *International Journal of Man-Machine Studies*, 22(4), 365 – 394, 1985.
- [Mai04] D. Mair. *E-Learning - das Drehbuch: Handbuch für Medienautoren und Projektleiter*. X. media. press Series. Springer, 2004.
- [Mil56] G. A. Miller. The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. *Psychological Review*, 63, 81–97, 1956.
- [Wei12] S. Weinbrenner. *SQLSpaces - a Platform for Flexible Language-Heterogeneous Multi-Agent Systems*. Dr. Hut, 2012.