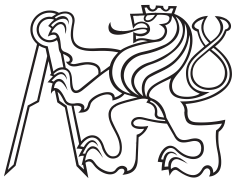
Diplomová práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačů

Adaptivní učení v softwarovém nástroji I3T pro výuku geometrických transformací

Bc. Jaroslav Holeček

Vedoucí: Ing. Petr Felkel, Ph.D.
Obor: Umělá inteligence
Leden 2023

Poděkování

Děkuji ČVUT a fakultě Elektrotechnické za možnost studovat a za předání nespočtu zajímavých informací, znalostí a dovedností. Dále bych rád poděkoval vedoucímu práce, panu Ing. Petru Felkelovi Ph.D. za přívětivé jednání a důležité připomínky k mé práci. Pan Ing. Petr Felkel Ph.D. má také mé poděkování za ochotu, se kterou se ujal této závěrečné práce v době mé nepříznivé studijní situace. V neposlední řadě patří poděkování členům mé rodiny, včetně Kláry Hrebikové, v průběhu času již inženýrky. Poděkování jim patří za podporu, mnohdy nejen psychickou, kterou mi poskytovali po celou dobu mého studia. Na závěr bych nerad zapomenul na přátele, kteří měli pochopení pro mé časové vytížení.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 10. ledna 2023

Abstrakt

Diplomová práce je věnována tématu adaptivního učení. Systémy adaptivního učení, tzv. Inteligentní výukové systémy (*Intelligent tutor systems*), nabývají na významu s rostoucím množstvím interaktivního, elektronického, nebo přímo online výukového obsahu. Inteligentní výukové systémy umožňují výrazně vyšší individualizaci výuky oproti klasickému modelu třída - učitel. Použití systému přináší efektivnější využití času studenta, pro kterého jsou systémem individuálně vybrány takové činnosti, které efektivně vedou k požadovaným cílům. Učitel při využití systému získá více času na řešení individuálních problémů studentů.
V práci je poskytnut rámcový přehled možností inteligentních výukových systémů a jejich možných cílů. Dále také přehled metod využívaných v těchto systémech. Podrobně je popsána možnost využití zpětnovazebního učení (*reinforcement learning*), konkrétně algoritmu Dvojitého Q-učení (*Double Q-learning*). V práci je představeno schéma činnosti systému a všechny jeho potřebné části.
Funkčnost byla ověřena využitím simulovaných dat, a to z důvodu nedostatku dat reálných. Pro simulaci dat je v práci navržen model studenta, úlohy a pokusu o řešení, umožňující zahrnutí minimálního času potřebného pro úspěšné vyřešení úlohy. Spuštěním ITS na simulovaných datech byla ověřena funkčnost systému. Při plánovaném nastavení systému bylo pozorováno neočekávané chování, které upozornilo na důležitost důkladně zvolené funkce odměny. Po vhodnějším nastavení funkce odměny vykazoval systém zlepšení výkonu simulovaných studentů o přibližně 25% při velikosti tréninkových dat v nízkých tisících studentů a téměř 50% zlepšení při velikosti tréninkových dat v nízkých desetitisících studentů.

Klíčová slova: Q, učení, inteligentní, výukový, systém, i3t, geometrické, transformace

Vedoucí: Ing. Petr Felkel, Ph.D.
Katedra počítačové grafiky a interakce, FEL, ČVUT v Praze

Abstract

The thesis is dedicated to the topic of adaptive learning. Adaptive learning systems called Intelligent tutor systems are becoming increasingly important with the growing amount of interactive, electronic and even online learning content. Intelligent tutoring systems allow for significantly more individualised learning compared to the classroom-teacher model. The use of the system results in a more efficient use of the student's time, for whom activities are individually selected by the system, that effectively lead to the desired goals. Using the system gives the teacher more time to solve individual students issues. Other information, which this thesis is only very marginally concerned with obtaining, may be information about student performance and activity and information about the tasks, e.g. their difficulty, warnings about inappropriately worded tasks, etc.

This paper provides a framework overview of the capabilities of Intelligent tutoring systems and their possible goals. It also provides an overview of the methods used in these systems. The possibility of using reinforcement learning, specifically the Double Q-learning algorithm, is described in detail. A flowchart of the system and all its necessary parts are presented. Simulated data is used to verify the functionality, due to the lack of real data.

To simulate the data, a model of the student, the task and the attempted solution is proposed in the thesis, allowing the inclusion of the minimum time needed to successfully solve the task. By running the ITS on simulated data, the functionality of the system was verified. Unexpected behavior was observed during the scheduled system setup, which highlighted the importance of a carefully chosen reward function. After adjusting the reward function more appropriately, the system showed approximately 25% improvement in simulated student performance for training data sizes in the low thousands of students and nearly 50% improvement for training data sizes in the low tens of thousands of students.

Keywords: Q, learning, inteligent, tutor, system, i3t, geometrical, transformation

Title translation: Adaptive learning in I3T software for education of geometric transformations

Obsah

Obrázky

Tabulky

Kapitola 1

Úvod

Tématem diplomové práce je vytvoření tzv. Inteligentního výukového systému (*Intelligent tutor system*, ITS). Podstatou tohoto systému je adaptivně, na základě interakce s uživatelem, obvykle studentem, v reálném čase vyhodnocovat úspěšnost studenta. Dle úspěšnosti studenta jsou systémem vybírány takové úlohy k vyřešení, které nejlépe splní zadaný cíl ITS. Jedná se tedy o počítačové adaptivní procvičování (*computerized adaptive practice*), kdy je obvykle cílem zlepšit co nejefektivněji znalosti studenta. Podobné, avšak netriviálně odlišné téma je počítačové adaptivní testování (*computerized adaptive testing*), kterým se tato práce nezabývá. [Pap15]. ITS zpracovává velké množství dat o interakcích studentů, s výukovým systémem, ke kterému je ITS připojen. ITS umožňuje učiteli eliminovat opakující se úkony, např. opravování úkolů. Dále může poskytovat informace o problematických úlohách či studentech a pomoci tak zaměřit učitelovu pozornost na místa, kde může nejefektivněji studentům pomoci. V některých systémech, jako např. Duolingo[1] je ITS využíván jako úplný, nebo přinejmenším hlavní nástroj pro vedení studenta. [Řih17] Svým efektem podobný systém je tématem této práce. Při jeho využití by mělo být možné automatizovat některé části učitelovi práce a zefektivnit tak využití času nejen učitele, ale i studentů.
Jak zmiňuje také autor [Pap15], pozitivní vliv podobných systémů na výkon studentů ukázali ve svých pracích např. [BDK13] či [KB08]. Již pouhá interaktivita systému a poskytování rychlé zpětné vazby a porovnání výkonu s ostatními studenty má pozitivní vliv [BKRI08].

[1] https://www.duolingo.com/

1.1 Terminologie

Pro snazší porozumění textu práce jsou v této kapitole vysvětleny pojmy, které jsou napříč prací používány:

- **Student** je uživatel, který řeší úlohy v I3T u kterých je zaznamenáváno vyřešení úlohy, případně další údaje jako např. čas potřebný pro vyřešení.
- **Učitel** je uživatel, který analyzuje výsledky studentů v I3T, případně vytváří úlohy, které studenti řeší.
- **Téma** (*knowledge component*, KC) je oblast znalostí nebo dovedností, se kterou ITS pracuje. Student má přiřazenu hodnotu jeho dovedností v tématu. Úloha má přiřazenu hodnotu obtížnosti v tématu. Příkladem může být „sčítání“.
- **Úloha** (*item*) je úkol, určený k vyřešení studentem. Např. „$7 + 3 =?$“
- **Otázka** (*question*) je konkrétní realizace úlohy představená studentovi. Příklad úlohy výše je vlastně také otázkou - úloha je idea sečtení čísla sedm a čísla tři. Na tuto úlohu se lze zeptat mnoha různými otázkami např. formou matematického zápisu „$7 + 3 =?$“, nebo slovně „Kolik je sedm plus tři?“. Odpověď je možné požadovat např. číselnou hodnotu, výběrem z možných odpovědí, správným rozmístěním obrázků a pod. Různé možnosti zadání odpovědi jsou také způsob jak předložit jednu úlohu pomocí více různých otázek.
- **Pokus** (*attempt*) je proces od zobrazení otázky studentovi po její úspěšné či neúspěšné vyřešení, tedy odeslání odpovědi, případně zanechání řešení.
- **Nápověda** je připravená informace, která je ve zvolený moment zobrazena studentovi s cílem pomoci mu k vyřešení otázky.

Poznámka: „Úloha“ a „otázka“ mohou být zaměňovány a obvykle tato záměna nemá vliv na význam sdělení. V místech, kde je nutné tyto pojmy rozlišovat je tato informace explicitně zmíněna.

1.2 Co je to tutor systém

Inteligentní tutor systém (*Intelligent tutoring system*, ITS) je počítačový program, který pracuje v reálném čase s daty z výukového systému (VS). Výukový

systém je prostředí, ve kterém jsou studentům nabízeny úlohy pro vyřešení. Cílem použití ITS je přizpůsobení VS se zaměřením na individualizaci výuky a procesu učení studenta. Pro studenta použití ITS obvykle znamená řešení úloh odpovídajících jeho znalostem a dovednostem, bez nutnosti takové úlohy vyhledávat. Přiměřenost obtížnosti úloh je dle [Jar07] důležitá pro motivaci studentů. Pro učitele systém přináší možnost analýzy výsledků studentů, odhalení těch, kteří potřebují s tématem pomoci, případně identifikaci cvičení, ve kterých studenti často chybují, nebo vykazují neobvyklé chování. Na základě dat z ITS mohou tvůrci výukových kurzů upravovat strukturu kurzů či evaluaci studentů [Jar07].
Stejně jako existuje mnoho výukových systémů pro nejrůznější témata např. algebra, geometrie, zeměpis, anatomie, algoritmizace, ..., liší se různé ITS vstupními daty, která zpracovávají. Dále se liší informace, které poskytují učiteli či studentovi. A také mají různé cíle, které by, v případě úspěšného nasazení ITS, měly být dosaženy. ITS je obvykle úzce svázán s tématem, kterým se zabývá VS ke kterému je připojen. Představení možnosti jak vytvořit ITS, který je více obecný a lze jej použít v různých VS, je jedním z výsledků této práce.

V rámci ITS jsou obvykle informace či úlohy pevně strukturovány např. učitelem nebo expertem v oboru a studentovi jsou předkládány na základě jeho výsledků na předchozích úlohách buď v rychlejším nebo volnějším tempu. Výsledkem této práce je vytvoření jiného typu ITS, než je ITS s pevnou strukturou úloh. Praktickou částí této práce je ITS, ve kterém je vhodné pořadí úloh generováno systémem na základě výsledků předchozích studentů na těchto úlohách. Výhoda tohoto přístupu spočívá např. v možnosti snadného přidání nové úlohy, kdy ji učitel pouze vytvoří a přidá do seznamu všech úloh. Zařazení úlohy na vhodné místo proběhne automaticky v rámci běhu systému. Dále tento přístup eliminuje potenciální učitelovo zkreslení při organizaci jednotlivých úloh.

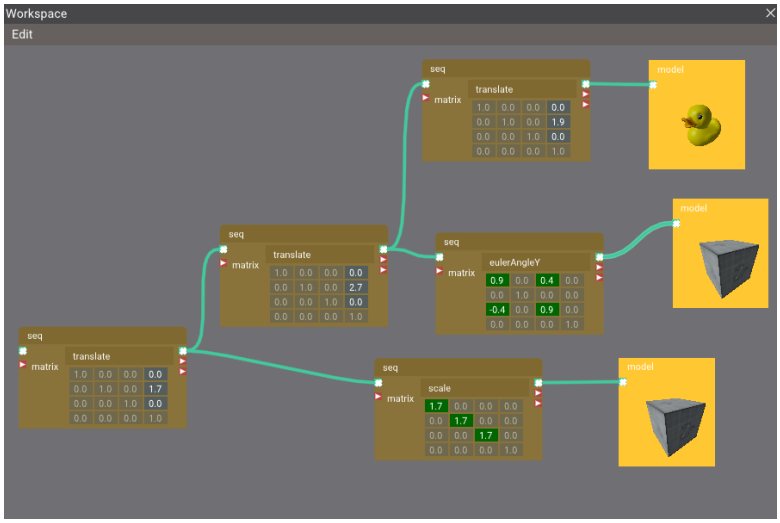
ITS sestává ze dvou programových smyček. Vnitřní smyčky (*Inner loop*) a vnější smyčky (*Outer loop*) [Van06]. Vnitřní smyčka je určena k napomáhání studentovi při řešení aktuální úlohy. V rámci vnitřní smyčky může být studentovi zobrazován jeho postup úlohou a splněné dílčí úlohy. Také může být studentovi zobrazována nápověda. Vnitřní smyčce není v této práci věnována pozornost. Vnější smyčka je určena k výběru úlohy, kterou bude student řešit. Vnější smyčka je hlavním předmětem této práce.

1.3 Stručné představení výukového nástroje I3T

Interaktivní nástroj pro výuku transformací (*Interactive Tool for Teaching Transformations*, I3T) je softwarový nástroj pro výuku geometrických transformací spolu s jejich maticovou reprezentací.
Hlavní část I3T je tvořena dvěma okny - Oknem 3D scény a oknem Pracovní plochy. V okně Pracovní plochy vytváří uživatel graf scény 1.1. Ğraf scény je grafická reprezentace výpočetního grafu typu DAG[2], ve které jsou vrcholy tvořeny moduly pro zobrazované objekty a pro maticové reprezentace geometrických transformací. Hrany poté reprezentují přesun hodnot mezi moduly. Uživatel může v Pracovní ploše měnit polohu těchto modulů a vzájemně je v rámci přípustných operací propojovat. Tím na objekty aplikuje geometrické transformace. Ve složitějších scénách lze využít moduly s operátory pro maticové operace jako sčítání, inverzní matice apod. Dále také moduly reprezentující kameru, automatický generátor hodnot a další.
V okně 3D scény je zobrazen obraz objektu v 3D prostoru. Objekt má pozici a tvar dané aplikovanými geometrickými transformacemi z grafu scény na Pracovní ploše. Interakce uživatele je možná jak s oknem Pracovní plochy, tak s oknem 3D scény. Výsledek této interakce je mezi okny v reálném čase oboustranně přenášen a student tak okamžitě vidí efekt operace, kterou provedl.
Na Pracovní ploše uživatel interaktivně mění graf scény. Propojuje či rozpojuje jednotlivé moduly, a tím určuje, jaké geometrické transformace jsou aplikovány. Hodnoty v maticovém vyjádření geometrických transformací může v jednotlivých modulech plynule měnit. Těmito akcemi je upravována výsledná geometrická transformace aplikovaná na objekt, který v reálném čase mění v okně 3D scény svou pozici, polohu, případně tvar.
V okně 3D scény má uživatel možnost interaktivně měnit pozici scény, respektive hodnoty parametrů hlavní kamery. Obraz hlavní kamery je obsahem okna 3D scény. Změnou hodnot parametrů této kamery tedy umožňuje sledovat scénu z různých úhlů a vzdáleností. Dále je také možné využít tzv. manipulátorů, při jejichž zapnutí je možné graficky „tahat“ za části vybraného objektu a tím v reálném čase měnit plynule jeho pozici, polohu nebo tvar. Takto měněné vlastnosti objektu jsou, opět v reálném čase, promítnuty do hodnot v maticové reprezentaci grafu scény v Pracovní ploše. [FMF+18]

Součástí I3T je také tzv. *Logger* , který umožňuje zaznamenávat činnost uživatele. Takovými činnostmi může být například vyplňování hodnot nebo přidávání či mazání modulů, ale také pohyb a klikání myši. Další zaznamenatelnou událostí je spuštění a dále splnění úlohy v tutoriálu, respektive samostatně spuštěné úlohy. Záznamy událostí týkající se úspěšného či neú-

[2]Orientovaný acyklický graf (*Directed acyclic graph*, DAG) má orientované hrany, které netvoří cyklus, pokud je brána v úvahu jejich orientace, ale mohou tvořit cyklus, pokud orientace hran v úvahu brána není.

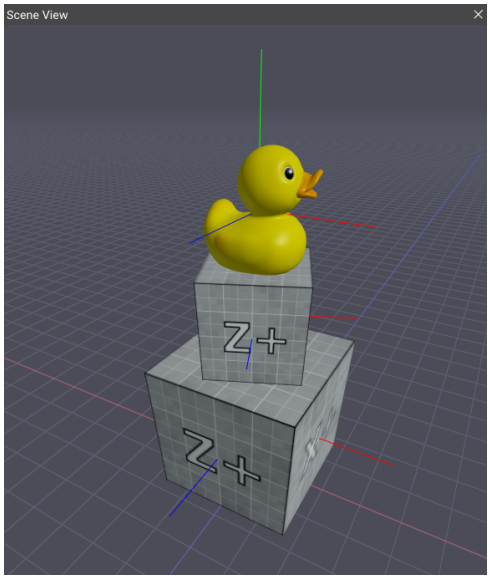


Obrázek 1.1: Okno Pracovní plochy (*Workspace*) v I3T. Obrázek zobrazuje jednoduchý graf scény se třemi objekty, respektive modely (ve žlutých uzlech) a pěti transformacemi (ve hnědých uzlech). Zobrazení tohoto grafu v okně 3D scény můžeme vidět na Obrázku 1.2 Spodní krychle (žlutý modul vpravo dole) je transformována zvětšením (hnědý modul uprostřed dole) a posunutím (hnědý modul vlevo). Prostřední krychle (prostřední žlutý modul) je oproti spodní krychli posunutá (druhý hnědý modul zleva) a také rotovaná (hnědý modul vpravo, uprostřed na výšku). Model kachničky (žlutý modul nahoře) je oproti prostřední krychli posunut (vrchní hnědý modul). Popsané návaznosti jsou reprezentovány modrozelenými spojnicemi modulů.

spěšného plnění úloh studentem a času, který student úloze věnoval, jsou zpracovávány doplňkem vytvořeným v této práci. Na základě informací o splnění úlohy a strávené době je výsledek studenta ohodnocen. Cílem ITS je předložit studentovi úlohy v takovém pořadí, při kterém dosáhne student co nejlepšího celkového hodnocení pro všechny úlohy.

1.4 Zaměření inteligentního výukového systému v programu I3T

Inteligentní výukový systém (*Intelligent tutor system*, ITS) vyvíjený v rámci této práce má dva cíle.
Hlavním cílem je individualizace práce studenta a zefektivnění práce studenta i učitele s I3T. Student, u kterého je předpoklad neznalosti tématu geometrických transformací a všech možností I3T, bude díky doplňku schopen s I3T pracovat samostatně. Samostatně pracovat znamená řešit připravené úlohy v efektivním pořadí, bez nutnosti výběru úloh učitelem či expertem. Význam pojmu „efektivní pořadí“ je závislý na zvoleném kriteriu 2.2.1. V této práci



Obrázek 1.2: Okno 3D scény v I3T. Na obrázku vidíme výsledek grafu scény zobrazeného na Obrázku 1.1. Spodní krychle je posunuta a zvětšena. Prostřední krychle je posunuta oproti spodní krychli a rotována. Kachnička je posunuta oproti prostřední krychli.

bylo zvoleno kriterium „**Nalézt pro každého studenta na základě jeho dosavadních odpovědí takové pořadí úloh, které povede k úspěšnému splnění všech úloh v co nejkratším čase**“. Z pohledu učitele doplněk umožňuje ponechat každého studenta pracovat samostatně. Učitel tak získá prostor pro pomoc studentům s pochopením individuálních konkrétních problémů, na které studenti při řešení úloh narazí. Při vytvoření nové úlohy učitelem je tato úloha v ITS automaticky zařazena mezi ostatní úlohy, bez nutnosti specifikovat téma a obtížnost úlohy. Zařazení úlohy probíhá na základě výsledků studentů v této úloze. Učitel tedy nemusí zastávat expertní roli a zařazovat úlohu mezi již existující. Tato část ITS se nazývá „vnější smyčka“[3].

Zmíněný cíl je také formulován v hlavní hypotéze práce a to ve znění: **Student se učí nové dovednosti a znalosti rychleji, než při procházení úloh dle svého uvážení nebo procházením na základě pevně stanoveného průchodu.**

Vedlejšími efekty, které jsou v práci popsány jen povrchně jsou: Možná analýza tématu geometrických transformací. Doplněk umožňuje zobrazit, jaké pořadí dílčích témat (translace, rotace, změny velikosti, a pod.) je pro uživatele nejvhodnější. Dále analýza nabízených úloh, u kterých lze odhalit, která dílčí témata jsou úlohami (ne)dostatečně pokryta, případně odhalit nevhodně či

[3]V rámci ITS existuje také „vnitřní smyčka“, které není v této práci věnována pozornost a která souvisí se sledováním činnosti studenta při řešení úlohy a např. poskytnutí nápovědy. Vzhledem ke komplexnosti většiny úloh v I3T, množství jednotlivých mezikroků, které je potřeba pro splnění úlohy vykonat a velké šíři tématu geometrických transformací, by mohla být vnitřní smyčka přínosným tématem navazující práce zaměřené na ITS v I3T

chybně formulovanou úlohu, u které studenti vykazují neobvyklé výsledky. Dále je také možná analýza studentů, u kterých lze modelovat úroveň dosažené znalosti či dovednosti ve vztahu k jednotlivým dílčím tématům.

1.5 Přizpůsobení zaměření práce okolnostem

Při návrhu zadání této práce nebyl program I3T[4] hotov. Na tvorbě nového I3T však pracovala poměrně velká, šestičlenná skupina studentů a studentek a byl tedy oprávněný předpoklad, že se program podaří dokončit dostatečně rychle. V případě včasného dokončení by byl program I3T intenzivně využíván v rámci výuky na Katedře počítačové grafiky a interakce. V takovou chvíli by existovala skupina studentů o velikosti řádově vyšších desítek až nižších stovek, která by I3T pravidelně využívala a bylo by tedy možné naměřit značné množství dat.

Bohužel, skupina pracující na vývoji I3T se v průběhu práce „rozpadla“ a vývoj I3T se tím velmi zpomalil. S vidinou dokončení alespoň základní funkcionality před začátkem semestru, se pozornost autora této práce, po domluvě s vedoucím práce, přesunula na vývoj GUI programu I3T. Výsledkem bylo vytvoření knihovny DIWNE popsané v příloze A.2 této práce. Implementace knihovny DIWNE byl pro autora náročný proces, viz Přílohu A.2, který zabral mnoho času. Funkční Editor grafů je však nezbytně nutnou součástí programu I3T a proto bylo nutné tento editor vytvořit. Bez editoru by nebylo možné, ani teoreticky naměřit data při využívání I3T studenty, neboť bez editoru by nebylo možné I3T využívat. Bohužel, jednotlivé části programu I3T jsou vzájemně úzce provázány a zprovoznění základní funkcionality se ukázalo jako obdobně náročné, jako zprovoznění celého programu. Pro smysluplné uživatelské využití I3T, při které by bylo možné provádět měření potřebná pro tuto práci, musí být funkční všechny části programu. Dokončit I3T včas se proto nepodařilo a tím se také uzavřela možnost získat měřením reálná data z používání I3T studenty.

V návaznosti na zmíněné události byla práce, po domluvě s vedoucím práce, rozšířena o vytvoření modelu umožňujícího simulovat data, která by mohla být naměřena při skutečném používání I3T. Výsledkem je model popsaný v Části 7. Autor si je vědom skutečnosti, že při využití simulovaných dat není možné splnit původní cíl práce stanovený v zadání ve znění *Výsledkem doplňku by měla být efektivnější práce studentů s nástrojem I3T - student se učí nové dovednosti a znalosti rychleji, než při procházení úloh dle svého uvážení, nebo*

[4]Respektive jeho nová verze, která umožňuje další rozšíření a lepší správu.

procházením na základě pevně stanoveného průchodu. Ověřte tuto hypotézu na třech skupinách studentů: První používá doplněk I3T, druhá volí úlohy dle svého uvážení, třetí se drží pevně stanoveného pořadí úloh, neboť v čistě simulovaném prostředí bez opory reálných dat, není možné věrohodně napodobit chování reálných studentů, měřit úroveň dovednosti reálných studentů ani vytvořit skupinu studentů, která „volí úlohy dle svého uvážení“. Jako cíl ITS v rámci simulovaných dat bylo zvoleno zmíněné kriterium: *Nalézt pro každého studenta na základě jeho dosavadních odpovědí takové pořadí úloh, které povede k úspěšnému splnění všech úloh v co nejkratším čase.*

V kapitole 5 jsou představeny a popsány dosažené výsledky na simulovaných datech. Při simulaci bylo zjištěno, jakým způsobem je potřeba nastavit parametry ITS. Obzvláště důležitá se ukázala funkce pro výpočet odměny za provedenou akci, nebo sekvenci akcí. Při jejím původním, nevhodném nastavení vykazoval algoritmus zcela opačné chování, než bylo očekáváno.

Výsledky práce, po přizpůsobení jejího zaměření pod tlakem zmíněných okolností jsou: Vytvoření přehledu metod používaných v ITS, plynoucí z rešerše problematiky. Tento přehled je sepsán v kapitole 2.7. Vytvoření modelu umožňujícího simulovat činnost studentů v I3T[5], viz kapitolu 7. Vytvoření Inteligentního výukového systému (ITS), tedy doplňku, který navrhuje uživateli vhodné úlohy, popsaného v kapitole 4.1. A také napojení ITS na I3T za využití *loggeru* v rámci kapitoly 4.2. Technická funkčnost ITS je ověřena na simulovaných datech, popsaných v kapitole 7, s cílem *nalézt pro každého studenta na základě jeho dosavadních odpovědí takové pořadí úloh, které povede k úspěšnému splnění všech úloh v co nejkratším čase.* Vedlejším produktem práce je knihovna DIWNE popsaná v příloze A.2, použitelná nejen v I3T, ale i v jiných programech obsahujících „editor grafů“.

[5]Zda vytvořený model odpovídá reálným datům je nutné ověřit poté, co budou reálná data naměřena.

Část I

Přehled tutor systémů

Kapitola 2

Inteligentní výukové systémy

Dle [Pap15] je pozitivní vliv Inteligentního výukového systému (*Intelligent tutor system*, ITS) na studenta dobře prokázaný. Např. [BDK13] ukazuje, že samotestování zlepšuje výkon studenta, avšak studenti mají tendenci přeceňovat své znalosti a na základě těchto zkreslených představ si vybírají úlohy, které nejsou pro jejich studium tak efektivní, jako jiné možné úlohy [KB08]. Přínosné je již pouhé seznámení studenta, zda odpověděl správně či ne ihned po jeho odpovědi [BKRI08].

V této kapitole jsou zásadní části ITS a možnosti nastavení parametrů těchto částí. Nejprve je popsán rozdíl mezi tzv. vnitřní a vnější smyčkou v části 2.1. Dále jsou v podkapitolách části 2.2 popsány možné cíle ITS 2.2.1, typy otázek a úloh 2.2.2, včetně jejich hierarchie a také typy dat 2.2.5, která lze v ITS měřit. V této podkapitole je také popsán návrh postupu, pomocí kterého se lze vypořádat s velkým množství různých úloh, otázek a dat.
Častým přístupem v ITS je modelování studenta. V části 2.3 je popsánu, co lze od takového modelu očekávat. Obdobně jsou v ITS modelovány úlohy a témata, tedy oblasti znalostí, kterými se ITS zabývá. Jednou z hlavních činností ITS je výběr úlohy pro činnost studenta. Požadavky na tento výběr shrnuje podkapitola 2.6.
Hlavní částí této kapitoly je podkapitola 2.7, ve které najdeme přehled metod využívaných pro práci ITS.

2.1 Vnitřní a vnější smyčka

Vnitřní smyčka ITS je jedna ze dvou [Van06] možností jak pomoci studentovi s procvičováním. Systémy vnitřní smyčky se zabývají nápovědou studentovi, který řeší obvykle komplexnější úlohu. Hlavní otázky pak jsou: kdy, jak často a jak podrobná by nápověda měla být. Další otázka nastává ve chvíli, kdy student nápovědu využije. Má se odpověď studenta započítat jako správná či chybná, nebo ji nepočítat vůbec? [Pap15]

Vnější smyčka je zaměřena především na pořadí úloh, v jakém jsou studentovi předkládány, což zahrnuje také otázku, zda již student dosáhl požadované úrovně v daném tématu či jaký typ úlohy (otevřená úloha, výběr z možností, tvořivá úloha) je v aktuální situaci pro studenta vhodný [Pap15]. Dle [Jar07] je také v rámci vnější smyčky jedním z cílů udržet studenta motivovaného (*flow-state*) k čemuž přispívá několik okolností: srozumitelnost cíle, různé typy úloh a otázek, bezprostřední zpětná vazba (ideálně na každý studentův krok), zobrazení statistiky po dokončení úkolu a porovnání s ostatními, rovnováha mezi úlohami, kde se student ujistí ve svých znalostech a těmi, které jsou pro studenta „výzvou“, adaptivní doporučování, tedy lehčí či těžší úlohy na základě aktuální úspěšnosti. Další možnosti jsou představeny v [Csi97].

Možnost umožnit studentovi ovlivňovat obtížnost úloh, např. pomocí dialogu s výběrem "chci lehčí, vyhovuje, chci těžší"v [Pap15] ukázala, že 1) jsou studenti nejvíce spokojeni (vybírají možnost "vyhovuje") s chybovostí kolem 35% 2) se tím snižuje zájem studentů 3) snížený zájem studentů je částečně kompenzován právě úpravou obtížnosti. Autoři však spekulují, že studenti s vnější motivací (např. zadaná práce ve škole) mají tendenci volit nižší obtížnost i pro jednoduché úlohy, což efektivitu učení snižuje.

Autoři [Jar07, s. 23] také upozorňují, že pro ITS je zásadní vytvořit doporučení také na základě individuálních parametrů každého studenta. Jako špatný příklad uvádí přímočarý přístup např. v IRT v kapitole 2.7.3 a dle autorů i ve vzdělávání obecně. Na základě dat jsou odhadnuty hodnoty parametrů daného modelu a ten je pak přímo použit pro předpovědi pro další studenty.

2.2 Vlastnosti tutor systémů (výukových systémů)

Pojem „tutor systém“ je velmi rozsáhlý a zahrnuje velké množství různých systémů. Tyto systémy se mohou lišit v mnoha ohledech, jako je cíl, ke kterému má použití tutor systému vést; typy úloh, které jsou v tutor systému dostupné; typy otázek, které a jak má student zodpovědět; typy dat, která je možné v tutor systému zaznamenávat; a další vlastnosti podrobněji popsané v odstavcích níže. Jedním z problémů odhadu hodnot parametrů takových systémů např. metodou maximální věrohodnosti u modelu IRT v kapitole 2.7.3 je dlouhá doba výpočtu při větším množství dat. Při práci studenta by takový ITS nemohl aktualizovat své chování na základě aktuálních odpovědí studenta. V takovém systému je možné pouze „offline“ přepočítat hodnoty parametrů, se kterými poté systém pracuje až do dalšího přepočítání. U modelů BKT v kapitole 2.7.4 č i PFA v kapitole 2.7.5 dochází k aktualizaci některých parametrů „online“, obvykle však i tyto modely mají mnoho parametrů, které jsou pevně dané a musí být spočítány „offline“. [Řih17]

Tématu porovnání různých systémů se věnovali autoři [Pel15]. Pro vyčíslení prediktivní síly modelu je dle autorů možné využít jako metriku prosté RMSE.

Situaci komplikuje také to, že obtížnost úloh vyjádřená hodnotou parametrů modelu nemusí být stále stejná pro všechny studenty. Může se relativně měnit např. se změnou studentů, regionu odkud studenti pochází a pod. [Řih17]

Na další problém upozorňují autoři [Jar07]. Při odhadování parametrů modelu se zahrnutým učením studenta je problémem nejednoznačnost. Pozdější úlohy jsou typicky studentem řešeny lépe a obecně není možné rozlišit, zda jde tento lepší výkon způsoben pokrokem studenta, nebo nižší obtížností pozdější úlohy. Tomuto problému lze dle autorů předejít pouze dostatečně obsáhlými daty.

2.2.1 Cíl inteligentního výukového systému

V rámci ITS jsou sbírána data o výkonu studenta při interakci s výukovým systémem. Cílem je přizpůsobit chování výukového systému úrovni a chování daného studenta, např. detekce/zavedení Učení ke zvládnutí viz kapitola 2.7.1, nebo individualizované doporučení, ať už jako nápověda v rámci právě řešené úlohy, či jako doporučení dalšího postupu v rámci dostupných materiálů a činností. Dále zpětná vazba o splnění úkolu a případné porovnání výkonu

s dalšími studenty, nebo modelování znalostí studenta a sledování zájmu studenta. V neposlední řadě také detekce tipování či nesmyslných odpovědí [PE20].

Pro vytvoření a práci s doplňkem, stejně jako pro vyhodnocení práce doplňku a jeho případné porovnání s dalšími možnostmi, je nutné zvolit požadovaný cíl používání doplňku. Dle zvoleného cíle se liší metrika, kterou splnění cíle měříme; požadované chování doplňku; i efekt použití doplňku na studenta. S přihlédnutím k doporučením v části 2.6 je zřejmé, že volba cíle není jednoznačná. Závisí na kontextu, ve kterém je program, respektive doplněk používán. Některé možností představuje následující seznam. Kromě individuálních možností je teoreticky možné požadovat kombinaci těchto možností, tedy např. volit kompromis mezi motivací studenta a přínosem úloh pro jeho dovednosti.

- Doplněk jako testovací nástroj. Při testování je naším cílem zjistit co nejpřesněji úroveň dovedností studenta. Testování obvykle probíhá v omezeném čase. Pro maximální informaci o dovednosti studenta z jeho odpovědi jsou nejlepší úlohy, pro které je šance správného řešení 50% [Řih17]. S přihlédnutím k omezenému času chceme, aby taková úloha trvala studentovi co nejkratší čas tak, aby bylo možné odhad studentovy dovednosti zpřesňovat dalšími úlohami. V I3T lze[1] u většiny úloh kompenzovat nižší dovednost vyšším věnovaným časem (viz vztah (7.3) případně Obrázek 7.4). Taková kompenzace má své limity a při nízkých hodnotách dovednosti by musel být věnovaný čas neúměrně dlouhý. Jak dlouhou dobu je student ochoten věnovat jednotlivým úlohám není autorovi známo, stejně jako parametry, které tuto dobu ovlivňují. Při tomto použití doplňku není prostor pro odhadování vlastností (hodnot parametrů) pro jednotlivé úlohy. Cílem je odhad dovedností studenta na základě jeho výkonu na konkrétních úlohách. Tento výkon však závisí i na hodnotách parametrů úloh a tyto parametry je proto nutné znát předem.
- Doplněk jako nástroj pro udržení pozornosti, motivace a prodloužení času stráveného u programu. Dle [Řih17] jsou vhodné metody pro udržení pozornosti a motivace (*engagement*) především co nejčastější zpětná vazba s aktuálními výsledky studenta a porovnání s výsledky ostatních studentů např. po každé úloze, případně několika málo úlohách. Dále interaktivita prostředí, kdy je „každý krok“ studenta reflektován odezvou programu, např. upozorněním, že krok je správný, a student se díky němu přiblížil ke správnému řešení. Diverzifikace úloh má také pozitivní vliv na pozornost. Vhodné je tedy zobrazovat takové otázky, které byly zobrazeny málo nebo vůbec. Vhodná obtížnost úloh je také faktor, který hraje

[1] Pravděpodobně lze znalost kompenzovat stráveným časem. Tento předpoklad je nutné ověřit z naměřených reálných dat

na pozornost vliv. Výsledky zmíněné také v [Řih17] potvrzují intuitivní představu, že příliš jednoduché úlohy studenta po krátkém čase nudí a příliš obtížné úlohy vedou k jeho frustraci. Obojí vyúsťuje ke ztrátě pozornosti a případně zanechání práce s programem. Autoři ukázali, že vhodná obtížnost úloh pro udržení pozornosti je 70% pravděpodobnost úspěšného vyřešení. V I3T je obvykle možné nedostatek dovednosti studenta, tedy příliš vysokou obtížnost úlohy, kompenzovat delší dobou řešení. Otázkou zůstává jaký vliv má tato kombinace obtížnosti a prodloužené doby řešení na udržení pozornosti a zaujetí studenta.

Pozornost a čas strávený studentem u programu také silně ovlivňuje kontext, ve kterém student program využívá. Např. při vyžití programu učitelem při výuce v hodině je čas omezen délkou výuky, respektive řízen učitelem, bez ohledu na osobní preference a stav studenta. Takový kontext je možné zjistit od studenta např. zaškrtnutím možnosti v dialogu před spuštěním samotného doplňku.

- Doplněk jako nástroj pro efektivní učení. Existuje několik kritérií, která mohou být použita k měření efektivity učení [Gus00]:

  1. Výsledky testů, jako jsou známkové hodnocení nebo standardizované testy, mohou poskytnout informace o tom, jak dobře se studenti učí. V rámci I3T lze za „standardizovaný test“ považovat kontrolní úlohy, které jsou stanovené učitelem a nejsou zahrnuty v tréninkových úlohách. Kontrolní úlohy by typicky sestávaly z více kroků a kombinovaly by očekávané/požadované dovednosti studenta. Cílem učení je dosáhnout co nejlepšího výsledku na těchto kontrolních úlohách.
  2. Měření času, který studenti stráví učením, může poskytnout informace o účinnosti učení. Pokud se studenti učí stejný materiál během kratší doby, může to znamenat, že jejich učení je účinnější. V kontextu I3T by taková možnost znamenala postupně předložit studentům všechny úlohy z dané oblasti. Cílem by bylo co nejrychlejší splnění všech požadovaných úloh.
  3. Měření schopnosti studentů aplikovat nové informace v praxi může poskytnout informace o účinnosti učení. Například, pokud jsou studenti schopni využít nové informace při řešení problémů nebo vykonávání konkrétních úkolů, může to znamenat, že jejich učení bylo účinné. Tuto schopnost není možné měřit přímo v rámci I3T a vyžadovalo by externí hodnocení studentů např. učitelem.
  4. Měření změn v chování studentů po učení může poskytnout informace o účinnosti učení [HT07]. Například, pokud jsou studenti po učení více sebevědomí nebo schopni lépe komunikovat nové informace, může to znamenat, že jejich učení bylo účinné. Podobně jako předchozí bod by i tento způsob vyžadoval externí hodnocení studentů např. učitelem.

Výběr další úlohy a otázky obvykle vychází pouze z odhadnutých dovedností studenta a obtížnosti úloh. Autoři [PE20] doplňují, že přínos pro vhodné doporučení může mít i samotná klasifikace odpovědi. Jako příklad, který svou povahou úzce souvisí s typem úloh v I3T, uvádějí (parafrázováno): „*Představme si úlohu, sestávající z více kroků. Student úlohu začal řešit, ale nedokončil ji. Měl by doplněk, po návratu studenta k programu např. další den, předložit tuto úlohu studentovi znovu, nebo raději zvolit jinou? Pokud student řešil úlohu na úrovni I+, pravděpodobně se snažil, ale úlohu nezvládl vyřešit. V takovém případě je lepší předložit studentovi jinou, typicky lehčí, úlohu. Student také mohl úlohu nevyřešit na úrovni I−, kdy pravděpodobně neřešil úlohu zcela soustředěně (např. se na ní pouze podíval na konci vyučování) a v takovém případě je smysluplné předložit ji studentovi znovu.*“

Možností jak měřit efektivitu s učení I3T je více a záleží na tvůrcích systému, čeho chtějí dosáhnout. Některé možnosti jsou:

1. Rychlost splnění všech cvičení. Čím rychleji student splní všechna cvičení, tím lépe.
2. Strávený čas u programu. Čím déle se podaří studenta udržet zaujatého a tedy využívajícího systém, tím lépe.
3. Nejlepší výsledek u kontrolních úloh. Kontrolní úlohu lze vnímat jako (semi)finální test.
4. Co nejpřesnější zjištění úrovně studenta. Tuto informaci lze využít jako vstupní bod při využívání programu.

2.2.2 Typ úloh

Ve výukových systémech lze úlohy rozdělit do několika hlavních kategorií. Jeden z typů je triviálního případ, kdy je úlohou studijní materiál, který slouží pouze k přečtení/shlédnutí a nemá smysl u něj zavádět pojmy jako správně či špatně splněn. Další typ úloh má za cíl zapamatování faktů studentem. Taková fakta obvykle nemají zřejmou logickou návaznost a je nutné si je pamatovat. Příkladem může být slovní zásoba v cizím jazyce; názvy měst, hor či řek a pod. Dalším typem jsou úlohy zaměřené na pochopení principu např. fyzikálního děje či matematického postupu, kde již mohou být jednotlivé dovednosti a znalosti kombinovány. Dalším typem jsou tzv. *puzzle* úlohy. Příkladem může být Sokoban[2] či Kulička[3]. Pro řešení takových úloh nejsou potřeba žádné

[2] https://www.umimeinformatiku.cz/sokoban
[3] https://www.umimeinformatiku.cz/kulicka

konkrétní znalosti. Obdobné úlohy jako zmíněné *puzzle* mohou být úlohy, jejichž řešení také spočívá v řetězení mnoha kroků. Liší se však v tom, že pro jejich řešení jsou také potřebné znalosti. Dalším typem mohou být úlohy pro trénink plynulosti např. výpočtu, které spočívají v drilování známého postupu. Dalšími typu mohou být úlohy zaměřené na aplikaci znalostí ve více či méně reálných scénářích, nebo úlohy pro rozvoj konkrétních dovedností jako je kritické myšlení, komunikace nebo týmová práce [BCK10], [Bru96].

- **Uzavřené úlohy** neumožňují studentovi vytvářet vlastní řešení, ale správnou odpověď pouze vybírá z připravených odpovědí (*multiple-choice question*). U těchto typů otázek nelze vyloučit možnost správné odpovědi pouhým hádáním. Kromě úlohy, pro kterou je otázka studentovi předložena a její obtížnosti, má také výrazný vliv na řešení volba rušivých možností (*distractors*) a jejich počet - tedy možností odpovědi, které jsou chybné a student mezi nimi musí najít odpověď správnou. [Řih17]

- **Otevřené úlohy** vyžadují po studentovi vytvořit část či celé řešení daného problému. Takové řešení obvykle sestává z více navazujících kroků. Co ovlivňuje obtížnost takové úlohy je předmětem výzkumu, ukazuje se však, že značně záleží na hloubce mrtvých stavů (*dead-ends*), možností vracet kroky zpět [Jar07] a minimální šířce (ve smyslu počtu uzlů) grafu na cestě od počátečního stavu do cílového. Výsledky prací zmíněných v [PE20] ukazují, že i při otevřených úlohách je typicky několik málo „běžných“ chybných odpovědí, které použije většina chybujících studentů.

Obsáhlý přehled typů úloh nalezneme v [Piz07]

2.2.3 Typ otázek

Typů otázek na které má student v rámci jedné úlohy odpovídat, a způsobů jak odpovídat, existuje velké množství. Mohou to být například: výběr jedné či více možností z nabízených odpovědí (*multiple-choice*). Extrémním případem je poté výběr z odpovědí Ano/Ne. Dále ruční napsání odpovědi, buď jednoduché např. jednoslovné, nebo komplikované jako např. napsání programu. Možné je také pouze upravit existující odpověď. Odpověď na otázku také může sestávat z více kroků, které na sebe mohou, ale nemusí navazovat. Každý výukový systém může obsahovat pouze jeden typ úloh či otázek, ale také jejich libovolné kombinace.

2.2.4 Hierarchie úloh

Dalším aspektem týkajícím se úloh je, zda jsou sestaveny do hierarchie. V takovém případě ovlivňují dovednosti studenta v jedné kategorii úloh jeho výkon i pro úlohy z jiných kategorii, které jsou společně s první kategorií v jedné nadřazené kategorii. Taková hierarchie může být značně rozsáhlá počtem kategorií a nebo hloubkou hierarchie. Obvykle také jedna úloha spadá do několika kategorií, neboť pro její úspěšné vyřešení je potřeba několik různých dovedností studenta. Jedním rozdílem je, zda jsou nebo nejsou ve výukovém systému úlohy uspořádány hierarchicky. V případě že ano, spočívá další rozdíl v tom, jak je tato hierarchie vytvořena. Úlohy mohou být např. vytvořeny nezávisle a poté expertem roztříděny do kategorií. Hierarchie může být vytvořena automaticky na základě podobnosti výsledků studentů na jednotlivých úlohách [Pel19]. Hierarchie úloh může být také využita při automatické generování otázek pro danou úlohu.

2.2.5 Typy dat

Některé typy dat je možné měřit pouze ve specifických výukových systémech, případně typech úloh či otázek. Např. počet kroků nelze měřit u otázek typu *multiple-choice*. Jiné typy dat lze využít v libovolném výukovém systému. Např:

- Správnost odpovědi/řešení. Správnost je jednoduchý a přímočarý způsob měření výkonu studenta. Jak zmiňuje [PE20], lze výkon studenta měřit i podrobněji a na základě dalších dat vytvářet přesnější předpovědi
- Doba odpovědi (*response time*) je čas, který student potřeboval ke splnění, případně nesplnění úlohy. Dobu odpovědi lze dobře využít pro předpovědi výsledků tvořivých úloh a úloh spočívajících v řetězení jednoduchých kroků. Jeho použití pro např. výběr z možných odpovědí je omezené. Tento údaj však může být silně zarušený (studenta při řešení vyruší okolní vlivy, může být unavený) a také závisí na nastavení a použití výukového systému (např. na chytrém telefonu trvá obsluha úlohy déle než při použití klávesnice a myši) [PE20]. Doba odpovědi má také multiplikativní charakter, tedy lepší student odpovídá rychleji úměrně k celkovému času, nikoliv v absolutních hodnotách) [vdL06] [vdL09a], což dle [Řih17] ukazuje na log-normální rozdělení doby odpovědi. Proto také navrhují autoři použití logaritmu naměřené hodnoty času. Log-normální rozložení odpovědi potvrzují i výsledky autorů [Jar07], kteří také využívají tohoto rozložení ke generování umělých dat pro evaluační účely - $p(t|\tilde{x}, \tilde{y})$, kde $\tilde{x}$

jsou parametry studenta a $\tilde{y}$ jsou parametry úlohy. Autoři [Řih17] ukázali, že z doby odpovědi lze odhadnout několik dalších informací. Rychleji odpovídající studenti vykazují vyšší vstupní znalost (*prior knowledge*) 0.81 oproti 0.72 a je o něco vyšší šance, že se do systému znovu připojí (18% oproti 10% u pomalejších studentů). Dále lze také z doby odpovědi odhadnout, zda bude student preferovat obtížnější či jednodušší úlohy. Autoři také naměřili, že doba odpovědi koreluje s pravděpodobností správné odpovědi na další otázku. Dále také zmiňují, že využití doby odpovědi umožňuje zlepšit shlukování (*clustering*) úloh a konvergence hodnot jejich modelů je při využití doby odpovědi rychlejší. Autoři [Řih17] také prezentují lepší výsledky při použití metody EMA [Pv17], která počítá s průměrnou hodnotou předchozích dob odpovědí, spočítanou na základě exponenciálního plovoucího průměru (*exponential moving average*).

Pro vypořádání se s ohromnou rozličností úloh a otázek navrhují autoři [PE20] vytvoření rozhraní mezi výukovým systémem a ITS. Toto rozhraní spočívá v rozdělení odpovědí studenta do šesti kategorií. Tři kategorie pro správné odpovědi a tři kategorie pro nesprávné odpovědi. Tento přístup vede ke snížení složitosti systému. Každý typ úloh generuje různý typ dat a bez zmíněného zjednodušení by bylo nutné zpracovávat každý typ dat samostatně. Nevýhodou je ztráta informací, které mohou být potenciálně důležité.

Autoři označují kategorie pro správnou odpověď „ C^+ (velmi rychlá), C^0 (běžná), C^- (např. použití nápovědy) a kategorie pro chybnou odpověď $I+$ (jedna chyba (při složené úloze), rychlá oprava po upozornění na chybnou odpověď), I^0 (běžná), I^- (mnoho chyb, velmi rychlá odpověď)“ Pro různé typy chování studenta a jejich možnou detekci pak uvádí příklady: *Odpověď I^- mezi sekvencí odpovědí C^+ a C^0 je pravděpodobně překlik, např. nechtěné odeslání nedokončeného řešení. Převážně odpovědi I^- s občasnými C^+ pravděpodobně znamenají, že student pouze tipuje a proklikává cvičení. Odpověď I^+ mezi řadou odpovědí C^0 a C^- pravděpodobně znamená, že student narazil na úlohu na hranici svých dovedností a je na místě drobná dopomoc či povzbuzení. Převážně odpovědi I^0 či I^+ pravděpodobně znamenají, že student se snaží, ale není schopen úlohy řešit; v takovém případě je na místě vrátit se k prerekvizitám či jednoduššímu procvičování.*

Takové rozdělení dle autorů [PE20] může mít přínos i pro tvůrce obsahu či celého ITS. Jako příklad uvádějí: *Pro odhad obtížnosti úlohy může být přínosné odfiltrovat odpovědi I^-, které často souvisí se studenty, kteří nepracují se systémem soustředěně a pouze se „porozhlíží“. Další příklad může být vysoký poměr odpovědí I^+, který může naznačovat nevhodně či nejasně formulovanou otázku. Pokud je otázka v pořádku, i tak je vhodné jí věnovat pozornost např. tvorbou nápovědy.* Dále klasifikace umožňuje snazší interpretaci odpovědí pro

studenta (či rodiče), který by nemusel dokázat rozpoznat důležité informace v hrubých datech (např. pouze z doby řešení úlohy). Zpětná vazba je důležitá součást ITS a proto by měla být pro uživatele srozumitelná.

2.3 Model studenta

Model studenta je matematický model, popisující, nebo alespoň zaznamenávající znalosti studenta, obvykle na základě předchozí interakce s VS a ITS. A dále také postup, jakým z těchto historických dat odhadnout hodnotu požadovaných informací. Pro co nejpřesnější odhad hodnot je model studenta klíčový [DB12]. Pro informaci o aktuální znalosti studenta se dle [Pap15] dá dobře využít doba pro vyřešení úlohy - např. správná rychlá odpověď ukazuje na pravděpodobnou vysokou znalost, rychlá chybná odpověď může indikovat chybnou znalost, nebo nezájem studenta o výukový systém či jiné nestandardní chování.

Odhadované informace mohou být:

- Pravděpodobnost správné odpovědi, což je obvykle hodnota, která je cílem předpovědi ITS při úlohách, jejichž řešení lze zcela vyhodnotit jako správně/špatně. Výsledkem předpovědi je poté hodnota pravděpodobnosti s jakou student odpoví správně. Jak zmiňují autoři [Pap15], výsledky výzkumů [YK13], [PWT12] či [LKM14] ukazují, že i malé zlepšení přesnosti odhadu má velký vliv na průběh procvičovaní a výkon studenta.
- Čas řešení úlohy, je hodnota která intuitivně i dle výsledků v práci [Jar07] úzce souvisí s dovedností studenta. Studenti s vyšší úrovní dovednosti řeší danou úlohu v kratším čase. Výsledkem předpovědi je očekávaný čas, který danému studentovi zabere řešení dané úloh. Dle autorů [Řih17] je potřeba interpretovat naměřená data obezřetně, neboť obtížnější úlohy obvykle řeší pouze zdatnější studenti a tedy v naměřených datech často chybí výsledky pro skupinu horších studentů.

V rámci modelu studenta je vhodné pro zlepšení přesnosti zahrnout předchozí znalost studenta (*prior knowledge*), která se projeví nejsilněji v prvních úlohách. Odhad předchozí znalosti je samozřejmě problematický při úplně prvním setkání studenta se systémem, avšak na předchozí znalost lze pohlížet i jako znalost konkrétního tématu a odhad pak provést na základě výsledků

studenta v jiných tématech. Takový odhad se obvykle velmi podobá způsobu, jaký se používá pro testování studenta. Autoři [Pap15] se zaměřují na odhad pravděpodobnosti správné odpovědi v tématu, které student zatím neprocvičoval za předpokladu homogenity studenta i témat - tedy, že lze znalost studenta v dané oblasti popsat jedním parametrem. Pro složitější případy pak navrhují seskupení studentů (např. podle IP adresy v případě procvičování geografických témat) a seskupení témat učitelem či automatizovaně.

Další částí modelu je zachycení změny znalostí studenta. Používáním výukového systému se znalost zvyšuje[4] a stejně může znalost studenta růst i aktivitami mimo systém. Na druhé straně postupem času dochází k zapomínání, kvůli kterému znalosti studenta v průběhu času samovolně klesají. Křivka zapomínání zmíněná autory [Pap15], předpokládá exponenciální snižování znalosti v čase. Pro její použití je potřeba dopočítat parametry této křivky [Pap15].

2.4 Model úlohy

Model úlohy je matematický model reprezentující úlohu či otázku tak, aby bylo možné analyzovat parametry úlohy nebo generovat otázku automaticky. Pokud model úlohy obsahuje informaci o diskriminačním faktoru[5] a náhodnosti, lze tyto údaje využít pro odhalení nevhodných či špatně formulovaných úloh. Dále také dle autorů [Jar07] je výhodnější zprvu studentům předložit méně diskriminační úlohy, abychom získali lepší představu o výkonech studenta.

2.5 Tématický model (*Domain model*)

Autoři [Řih17] uvádějí, že v tématickém modelu jsou jednotlivé úlohy roztříděny do příslušných témat, přičemž jedna úloha může zasahovat do více témat. Tématický model lze využít dvěma způsoby. Za prvé jako vstupní informace pro model studenta, což vede ke zvýšení přesnosti odhadu jeho odpovědi. Za druhé jako informace pro uživatelské rozhraní pro srozumitelnou vizualizaci postupu.

[4] doufejme

[5] Diskriminační faktor je hodnota, která vyjadřuje, nakolik se na konkrétní úloze liší výkon studentů s nízkou a vysokou úrovní dovednosti

2.6 Výběr úlohy či otázky

Dle [Řih17] musíme v této části ITS při výběru úlohy či otázky (*item selection*) brát do úvahy tři aspekty:

1. Udržet studenta motivovaného a zaujatého. Ukazují, že vhodnou hodnotou pravděpodobnosti správné odpovědi je 0.7.
2. V rámci požadovaného výsledku diverzifikovat otázky, tedy zobrazit studentovi spíše otázky, které ještě nebyly zobrazeny vůbec, nebo málo.
3. Neopakovat stejnou úlohu, tedy volit spíše otázky týkající se jiné úlohy, než která byla nedávno procvičována.

Použití modelu pro výběr vhodné úlohy v reálném nasazení spočívá v odhadu parametrů modelu tak, aby co nejlépe odpovídal naměřeným reálným datům. Dále je pro studenta, který aktuálně ITS používá, spočítán modelem odhadovaný čas a pravděpodobnost úspěšného vyřešení všech úloh v systému. Zahrnuty mohou být pouze vybrané úlohy, např. dle tématu. V navazující části ITS je na základě vypočtených hodnot zvolena vhodná úloha s požadovanou očekávanou pravděpodobností úspěšného vyřešení a s požadovaným odhadovaným časem.

Nastavení ITS se může lišit v závislosti na požadovaném výsledku a situaci, např. při testu je cílem v omezeném čase co nejpřesněji zjistit úroveň znalostí studenta, při učení v omezeném čase (např. vyučovací hodina) může být cílem co nejrychlejší získávání nových dovedností, v jiných situacích může být cílem kompromis mezi získáváním nových dovedností a zábavností. Každá z těchto situací vede na jiné požadavky pro hodnoty parametrů úlohy představené studentovi.

2.7 Metody v tutor systémech

Seznam následujících podkapitol poskytuje přehled metod používaných v ITS. Metody se významně liší svou složitostí i přístupem. Část z metod spočívá ve vytvoření matematického modelu a odhadování hodnot tohoto modelu na základě naměřených dat. Doporučení pak spočívá odhadu např. očekávaného času řešení pro každou úlohu a výběr té úlohy, která splňuje určený požadavek.

Další část metod využívá strojové učení, založené na průběžném zpracovávání dat a průběžné přizpůsobování chování systému pro dosažení zvoleného cíle.

2.7.1 Učení ke zvládnutí (*Mastery learning*)

Studentovi jsou předkládány úlohy z vybrané oblasti do doby, než v této oblasti dosáhne požadované (obvykle vysoké, např. 90%) úspěšnosti. Až po dosažení této úspěšnosti je vpuštěn do dalšího tématu. [Pap15] Pro tuto metodu je zásadní volba typu vstupních dat, která hrají zásadnější roli, než model studenta. Kriteriem může být např. jednoduchý plovoucí průměr. [Pv18]. Tato metoda také vyžaduje zařazení úloh do skupin. Nutné je také určení pořadí jednotlivých skupin, ve kterém má student výukovým systémem procházet.

2.7.2 Metoda odstupňovaných intervalů (*Graduated-Interval Method*)

Metoda stupňujících se intervalů (*Graduated-interval recall method, Pimsleur method*), při které je představeno nové téma a úlohy pro něj jsou studentovi předkládány stále častěji (prokládány úlohami na jiné téma či jinými činnostmi). Mírně složitější je Leitnerova metoda (*Leitner System*) , která spočívá ve vytvoření skupin dle frekvence procvičování tématu (např. v intervalu 1 dne - 2 dnů - 4 dnů - . . .). Nové téma je zařazeno do skupiny s nejkratším intervalem procvičování a pokud student odpoví správně, je téma posunuto do skupiny s delším intervalem. Naopak pokud odpoví špatně, je téma posunuto do skupiny s kratším intervalem. První metoda zcela ignoruje vstupní znalost studenta a ani nijak nesleduje jeho výsledky v průběhu učení. Druhá metoda umožňuje téma, které student ovládá, rychle posunout do skupin s dlouhým intervalem a zaměřit se tak intenzivněji na témata, ve kterých student často chybuje. [Pap15]

Obě tyto metody vyžadují buď rozřazení úloh učitelem do témat, nebo mohou pracovat s každou úlohou zcela individuálně. Nelze tedy pomocí nich přímočaře sledovat vzájemnou závislost jednotlivých úloh (např. přínos vyřešení jedné úlohy pro vyřešení jiné úlohy).

Přizpůsobení aktivaci paměti

Metodu Přizpůsobení aktivaci paměti (*Adaptive Character of Thought-Rational*) shrnuje [Jar07] následovně: Pro každé téma je v modelu uložena řada $t_1 \dots t_n$, kde každé t_i určuje, před jakou dobou bylo téma uživatelem procvičováno. S touto řadou můžeme spočítat tzv. aktivaci paměti (*memory activation*) m. Způsoby jak tuto hodnotu spočítat se mírně liší. Jedním z nich je např. [PA05]:

$$d_1 = a \tag{2.1}$$

$$d_i = ce^{m_{i-1}} + a \quad i \in (2, \dots, n) \tag{2.2}$$

$$m_n = \ln(\sum_{i=1}^{n} t_i^{-d_i}) \quad , \tag{2.3}$$

kde a, c : parametry „křivky zapomínání“

d : „parametr zapomínání“

t_i : doba od i-tého procvičení

m_n : hodnota aktivace paměti po n procvičeních

Způsobů jak využít tuto hodnotu je také více, např. [PA08]: Pokud je mezi tématy nejnižší hodnota aktivace mezi „neprocvičeno“ α a „optimální hodnotou“, je vhodné toto téma procvičovat. Pokud je nejnižší hodnota nižší než α, je vhodnější věnovat se studijním materiálům, než procvičování. Nebo pro výpočet pravděpodobnosti správné odpovědi:

$$P = \frac{1}{1 + e^{\frac{\tau - m_n}{s}}} \quad , \tag{2.4}$$

kde P : pravděpodobnost správné odpovědi

τ : práh (obtížnost tématu)

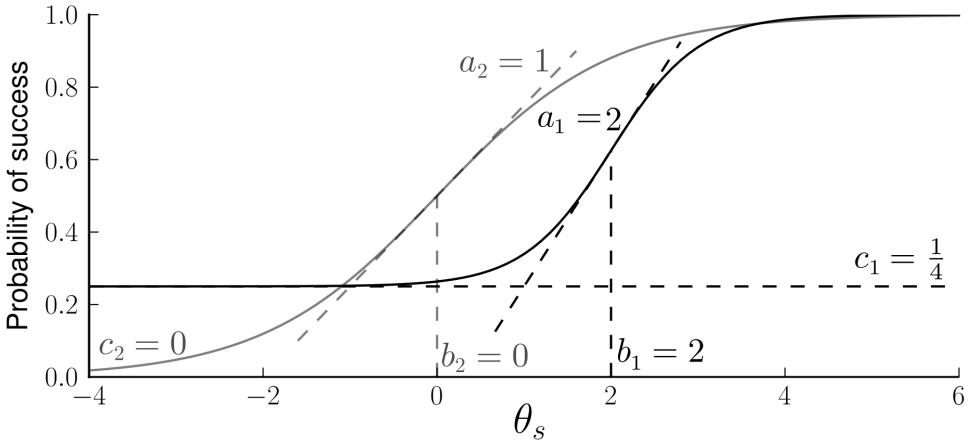
s : citlivost

m_n : hodnota aktivace po n procvičení

2.7.3 Teorie odezvy (*Item Response Theory*)

Teorie odezvy (*Item response theory*, IRT) [Bak01] má původ v psychologii a snahách o automatizované adaptivní testování, tedy co nejpřesnější zjištění úrovně zkoumané dovednosti (v kontextu této práce u studenta) [Pap15] [PE20] [Řih17].

Principem je logistická funkce (v různých modelech nalezneme různý počet



Obrázek 2.1: Parametry logistické funkce v IRT. Obrázek zobrazuje vliv parametrů a, b, c, tedy diskriminačního faktoru, obtížnosti a koeficientu hádání na tvar křivky. Křivka zobrazuje vztah mezi dovedností studenta Θ_s a pravděpodobností správné odpovědi (svislá osa). Obrázek převzatý z [Řih17].

parametrů) ve tvaru (viz také obrázek 2.1) [Pap15], [Jar07]:

$$P(corr|a,b,c,\Theta) = c + (1-c)\frac{e^{a(\Theta-b)}}{1+e^{a(\Theta-b)}} \quad , \tag{2.5}$$

kde $P(corr)$: pravděpodobnost správné odpovědi
a : diskriminační faktor
určuje je úloha rozděluje studenty dle úrovně znalosti
b : obtížnost úlohy
c : parametr hádání je pravděpodobnost,
s jakou lze úlohu vyřešit pouhým hádáním
Θ : dovednost studenta značí úroveň studenta v tématu,
které úloha procvičuje

Rozšíření IRT modelu je možné. Obzvláště pro účely ITS je vhodná možnost pracovat s více dovednostmi studenta a více složkami obtížnosti úlohy [Ack94]. V rovnici (2.5) je v takovém případě např. $\vec{a}(\vec{\Theta} - \vec{b})$ místo $a\Theta - b$.

Pro použití IRT je zásadní odhad parametrů logistické funkce, k čemuž je využita např. metoda maximální věrohodnosti, kdy je nejprve pro odhad parametrů úlohy využit pevně daný parametr studenta. Poté je naopak parametr studenta odhadnut na základě pevně daných parametrů úlohy a tyto dva kroky jsou opakovány dokud se hodnoty parametrů mění [Bak01], [Pap15]. Další přístup je založen na ELO viz kapitola 2.7.5 systému [Elo78] původně použitém pro hodnocení hráčů šachu. Tento přístup pohlíží na řešení úlohy jako „zápas“ mezi studentem a úlohou. Pokud student úlohu vyřeší,

klesne obtížnost úlohy (v šachu rating hráče, který prohrál) a stoupne odhad znalosti studenta. Opačně stoupne obtížnost úlohy a klesne odhad znalosti studenta, pokud student úlohu nevyřeší. [Řih17]

Jak připomíná [Jar07], parametry IRT nejsou jednoznačné (např. lze k parametrům Θ a b přičíst libovolnou konstantu a získáme totožný model). To je obvykle řešeno normalizací (např. $E(\Theta) = 0$ a $\sigma(\Theta) = 1$). Výhodou je možnost odhadnou parametry IRT i z odpovědí studentů, kteří nejsou reprezentativním vzorkem [Bak01], takzvaná skupinová invariance (*group-invariance*).

Podobný přístup nabízí i modelování doby odpovědi, respektive logaritmu doby odpovědi, dle [Řih17]:

$$\tau_{il} \sim \mathcal{N}(d_i - a_i\tilde{\Theta}_l, c_i) \quad , \tag{2.6}$$

kde τ_{il} : logaritmus odhadu doby
d_i : obtížnost i-té úlohy (logaritmus doby průměrného studenta)
a_i : diskriminační faktor určuje, jak se liší doba řešení pro studenty s různou úrovní dovednosti
c_i : parametr hádání/náhodnosti úlohy
$\tilde{\Theta}_l$: parametr studenta - úroveň znalosti studenta v tématu, které úloha procvičuje
i : číslo úlohy ,

který také umožňuje rozšíření jako: započítání více dovedností, učící či křivka zapomínání.

2.7.4 Sledování znalosti (*Knowledge Tracing, Bayesian Knowledge Tracing*)

Model předpokládá, že student je vystaven úloze na sledované téma vícekrát v průběhu času a odpovídá na otázku zda (s jakou pravděpodobností) se student již téma naučil. Např. [CA95] *odpovědi studenta* $0, 0, 0, 1, 0, 1, 1, 1$ *v čase (*0 *značí chybnou odpověď,* 1 *správnou odpověď) znamenají, že student téma umí?* Na rozdíl od modelu IRT tento model (*Bayesian Knowledge Tracing*, BKT) implicitně pracuje s učením [Řih17]. V modelu jsou v nejjednodušším případě zachyceny čtyři pravděpodobnosti s předpokladem, že každá úloha (či krok v úloze) odpovídá jedné znalosti [Jar07] [Pap15]:

- Pravděpodobnost, že student téma již umí

- Pravděpodobnost, že se student téma naučí aktuální činností
- Pravděpodobnost správné odpovědi, pokud student téma neumí (tedy hádá)
- Pravděpodobnost chybné odpovědi, pokud student téma umí (omyl, překliknutí)

Student je poté v jednom ze dvou stavů Umí/Neumí s tím, že výše zmíněné pravděpodobnosti udávají očekávaný výsledek řešení úlohy a změnu stavu studenta.

Model předpokládá, že vždy, když je student vystaven úloze zahrnující dané téma, má šanci se toto téma naučit. Pro přechod mezi stavy i předpovědi výsledků jsou použity podmíněné pravděpodobnosti na základě pozorovaných výsledků studenta. BKT je možné rozšířit např. o zapomínání, vstupní dovednost studenta či zahrnutí prerekvizit. Pro odhad parametrů z naměřených dat existuje několik možností viz [Řih17].

2.7.5 Doporučovací systém na základě shlukování (*Recommender System based on collaborative filtering*)

Jak zmiňují také autoři [Jar07]. Doporučovací systémy založené na shlukování (*collaborative filtering*), jsou v současné době velmi rozšířené na nejrůznějších platformách, od nabídky knih [LSY03], přes zobrazování zpráv [DDGR07], po doporučování filmů a seriálů [BKV07], [Tak08] Autoři [Jar07] připomínají, že pro využití této metody v ITS je potřeba vytvořit doporučovací systém adaptivní pro každého studenta zvlášť, neboť každý student je rozdílný v rámci znalostí jednotlivých témat. Autoři dávají jako protipříklad IRT, s přímočarým postupem nasbírání dat, kalibrování modelu a jeho použití.

Collaborative filtering zahrnuje tři metody:

Odchylka od průměru. Jedná se o metodu, která využívá *stochastic gradient descent*.

$$\hat{r}_{ui} = \mu + b_u + b_i \tag{2.7}$$

$$min \sum_{u,i} \quad (r_{ui} - \mu - b_u - b_i)^2 + \lambda(\sum_u b_u^2 + \sum_i b_i^2) \quad ,$$

kde μ : průměrný rating (v aplikaci ITS např. čas)
b_u : odchylka uživatele od průměru
b_i : odchylka úlohy od průměru
$\hat{r}_{ui}$: odhadnutý rating, resp. jak je úloha vhodná pro studenta)
λ : koeficient pro (předejití) přeučení modelu

Faktorizace matic (*Matrix factorization*). Při které je studentovi a úloze přiřazen vektor skrytých proměnných a očekávaný rating (vhodnost úlohy) je poté vyjádřen:

$$\hat{r}_{ui} = \mu + b_u + b_i + \vec{q_i}^T \vec{p_u} \tag{2.8}$$

$$min \sum_{u,i} \quad (r_{ui} - \mu - b_u - b_i - \vec{q_i}^T \vec{p_u})^2 + \lambda(b_u^2 + b_i^2 + ||\vec{q_i}||^2 + ||\vec{p_u}||^2), \tag{2.9}$$

kde μ : průměrný rating (v aplikaci ITS např. čas)
b_u : odchylka uživatele od průměru
b_i : odchylka úlohy od průměru
$\hat{r}_{ui}$: odhadnutý rating)
λ : koeficient pro (předejití) přeučení modelu
$\vec{q_i}$: vektor skrytých parametrů pro úlohu)
$\vec{p_u}$: vektor skrytých parametrů pro studenta

Metoda sousedů. Je metoda založena na identifikaci k nejpodobnějších úloh. Podobnost je obvykle měřena Pearsonovým koeficientem, další možnosti a jejich srovnání v [Pel19]) a poté odhadnutí ratingu (v ITS vhodnosti další úlohy) jako průměrů těchto sousedících úloh:

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j \in S^k} s_{ij}(r_{uj} - b_{uj})}{\sum_{j \in S^k} s_{ij}} \quad , \tag{2.10}$$

kde $\hat{r}_{ui}$: odhadnutý rating (v ITS např. čas) pro uživatele u a úlohy i
b_{ui} : odchylka uživatele a úlohy
s_{ij} : vzdálenost úlohy i od úlohy j
r_{uj} : rating úlohy (v ITS např. čas) úlohy u uživatelem j
S^k : Množina všech úloh, které již byly hodnoceny (řešeny)

Dle [Pel19] není podobnost úloh dobře definovaný pojem a není zcela zřejmé, co má taková podobnost úloh označovat. Navíc je dle autorů značené odvislá od konkrétní aplikace či tématu. Autoři doplňují, že výběr vhodné metriky [či měření] je zásadnější než výběr konkrétního clusterovacího algoritmu a upozorňuji, že některé clusterovací algoritmy vyžadují dodržení specifických vlastností měření vzdálenosti.

Výsledky práce [Řih17] naznačují, že v reálném případě, kdy studenti nevyplní všechny úlohy, jednotlivé dovednosti nejsou ortogonální a témata úloh nejsou nezávislá, je pro dobré rozdělení úloh potřeba až miliony odpovědí a deseti tisíce studentů.

Elo

Jak zmiňují i autoři [Pap15] [Řih17]. Elo systém byl navržen a je používán pro srovnávání hráčů šachu [Elo78]. Každý hráč má přiřazenu úroveň (reálné číslo Θ). Ve chvíli, kdy proti sobě nastoupí dva hráči i a j, získáváme výsledek zápasu $R[ij] \in 0, 1$, kde 0 značí prohru hráče i a 1 značí výhru hráče i. Pravděpodobnost výsledku zápasu je v Elo systému odhadnuta logistickou funkcí :

$$P(R_{ij} = 1) = \frac{1}{1 + e^{(\Theta_j - \Theta_i)}} \quad , \tag{2.11}$$

kde $P(R_{ij} = 1)$: pravděpodobnost výhry hráče i
Θ_i : Elo skóre hráče i
Θ_j : Elo skóre hráče j

po každém zápasu je skóre hráčů upraveno dle:

$$\Theta_{new} = \Theta + K(n)(R_{ij} - P(R_{ij} = 1)) \tag{2.12}$$

$$K(n) = \frac{a}{1 + bn} \quad , \tag{2.13}$$

kde Θ_new : nová hodnota Elo skóre
Θ : aktuální hodnota Elo skóre
$K(n)$: funkce nejistoty (*uncertainty function*)
n : počet změn skóre
R_{ij} : výsledek zápasu
a, b : meta parametry nastavené dle dat

Dle autorů [Pap15] je Elo systém v porovnání s dalšími metodami (jako metoda maximální věrohodnosti, procento správných odpovědí či lidské hodnocení) spolehlivý a výpočetně nenáročný způsob odhadnutí obtížnosti úlohy a dovednosti studenta.

Analýza dílčích výkonů (*Performance Factor Analysis*, PFA)

PFA je metoda, která zahrnuje více dílčích dovedností, ze kterých se skládá studentův výkon a které jsou potřeba ke splnění úlohy. Výpočet odhadu výsledku je poté velmi podobný IRT. Do hodnoty dosažené znalosti m započítává obtížnost úlohy β, správné odpovědi na danou úlohu s a chybné odpovědi na danou úlohu f:

$$m = \sum_{i \in KC} -\beta_i + \gamma_i s_i + \delta_i f_i \tag{2.14}$$

$$P(correct) = \frac{1}{1 + e^{-m}} \quad , \tag{2.15}$$

kde $P(correct)$: pravděpodobnost správné odpovědi
m : hodnota znalosti studenta
KC : množina dílčích dovednosti (*Knowledge Components*)
β : obtížnost úlohy
s : počet správných odpovědí
f : počet chybných odpovědí
γ : parametr znalosti
δ : parametr neznalosti

Jak připomínají i autoři [Pap15], tento způsob nedokáže rozlišit mezi případy, kdy student nejprve odpovídal na úlohu (danou oblast) chybně a poté správně (tedy se ji naučil) či naopak. Parametry γ a δ mohou být odhadnuty expertem, spočítány pomocí Q matice viz kapitola 2.7.6, případně odhadnuty metodou maximální věrohodnosti [Řih17].

Hluboké neuronové sítě (*Deep neural network*, DNN)

Jak uvádí autoři [PE20], DNN zpracovává přímo hrubá data různých typů. V rámci kontextu ITS je metoda nazývána „*deep knowledge tracing*“ [PBH+5a]. Data mohou obsahovat informace o správnosti, času odpovědi, ale i další [WSLP17]. Dle autorů [PE20] je výhodou absence člověkem vytvořeného pevného modelu studenta, úlohy či hierarchie úloh. Což omezuje možnou chybu způsobenou špatně navrženým modelem. Nevýhodou pro ITS pak dle autorů může být neprůhlednost systému, obtížná interpretace naučeného modelu a nejistota stability v očekávaných předpovědích. Pokud je cílem další práce s naučeným modelem například identifikace příliš obtížných/lehkých úloh nebo úloh nevhodně formulovaných, není snadné z tohoto typu modelu tyto vlastnosti vyčíst. Tématu interpretace naučeného modelu DNN se věnuje pozornost i v rámci medicíny a automatického zpracování lékařských snímků [WGZ+19].

Model dle [Jar07]

Autoři [Jar07] navrhli model v následujícím tvaru:

$$\tau_{sp} = b_p + a_p\theta_s + N(0, c_p^2 + \sigma_s^2 a_p^2) \quad , \tag{2.16}$$

kde τ_{sp} : logaritmus odhadovaného času pro studenta s a úloha i

b_p : obtížnost problému

a_p : diskriminační faktor problému

c_p : rozptyl problému (jak dobře lze tipovat)

θ_s : úroveň znalosti studenta

σ_s : rozptyl studenta (student nepodává konstantní výkon)

U toho modelu není odhad jeho parametrů závislý na reprezentativnosti vzorku, je tzv. *sub-group invariant*. To je dle autorů [Jar07] důležité proto, že obtížnější úlohy obvykle řeší a úspěšně vyřeší pouze studenti s lepší dovedností a tedy jejich nereprezentativní vzorek. Je potřeba normalizovat střední hodnoty θ_s obvykle na 0 a a_p na -1. Rozptyl šumu závisí více na parametru studenta σ_s pro úlohy s vyšším diskriminačním faktorem a_p. Výpočet odhadu poté probíhá postupně ve dvou krocích následovně:

$$p(\theta'|s) = N(\theta'|\theta_s, \sigma_s^2) \tag{2.17}$$

$$p(t|\theta', p) = N(t|b_p + a_p\theta', c_p^2) \tag{2.18}$$

Autoři také představili další modely pro porovnání, mezi kterými nalezneme také model se zahrnutým učením a model pro více dovedností/oblastí problému.

2.7.6 Q-matice (*Q-matrix*)

Metoda Q-matice přiřazuje každé úloze (řádek v matici) skupinu dovedností (sloupce v matici), které jsou potřeba k jejímu vyřešení. Míra dovednosti může být vyjádřena hodnotou prvku matice v rozmezí 0 až 1. Každý student má přiřazeny dovedností a cílem algoritmu je nalézt takové hodnoty těchto dovedností, které nejlépe odpovídají výsledkům studenta Řešení tohoto problému obvykle vede na faktorizaci matice, např. ASL [Řih17].

2.7.7 Model pomocí MDP a Q-učení (*Q-learning*)

Tomuto modelu se věnují např. práce [YXJZ18], [XHJHh21] nebo [THYC20a]. Autoři [CCL16] a [KMR18] se ve svých pracích zabývají využitím zpětno-

vazebního učení (*reinforcement learning*, RL) pro doporučování úloh. Využití hlubokého zpětnovazebního učení (*deep Q-learning*) zpracovali autoři v [THYC20b]. Práce ukazují, že použití RL může vést ke zlepšení výkonu studentů a k vyššímu zapojení do výuky.

Interakce studenta a doporučovacího systému je modelována jako Markovův rozhodovací proces (*Markov decision process*, MDP). Agentem činícím akce je v tomto případě doporučovací systém, akcí je předložení vybrané úlohy studentovi. Stavem MDP je označení úrovně jednotlivých dovedností studenta (*knowledge component*, KC). Těchto dovedností může být z principu více a také jejich hodnoty mohou být buď binární umí/neumí, nebo podrobnější. Přechodová funkce MDP zachycuje, s jakou pravděpodobností se změní hodnoty studentových dovedností při činnosti na daném (doporučeném) materiálu.

Protože tato přechodová funkce není z podstaty problému známa, je řešení založeno na Q-učení. Hledaná strategie (*policy*), tedy v jakém stavu doporučit jaký studijní materiál, je iterativně upřesňována na základě výsledků plynoucích z „vyzkoušení“ vybrané akce v daném stavu.

Obtížných míst tohoto přístupu je dle výše zmíněných prací několik. Jedním je stanovení odměny (*reward*) a jejího výpočtu. Do odměny musí být zahrnut cíl používání ITS. Cíl často souvisí s úrovní dovedností studentů a v takovém případě je potřeba určit, jakým způsobem bude jejich úroveň měřena. Pro smysluplné výsledky je potřeba parametry výpočtu odměny vhodně nastavit, což může být obtížné, např. v [THYC20a] je tento výpočet nahrazen jednoduchou neuronovou sítí. Dalším problémem je velké množství možných stavů. Existuje mnoho kombinací, v jakém pořadí student může studijními materiály procházet. Navíc může student daný materiál řešit při každém pokusu s různým výsledkem, např. velmi snadno vyřešit, těsně nevyřešit a pod.

Část II

Realizované řešení

Kapitola 3

Algoritmus zvolený pro Inteligentní výukový systém v I3T

V této kapitole je podrobně popsán algoritmus Dvojitého Q-učení *Double Q-learning* (DQL), který byl vybrán pro použití v Inteligentním výukovém systému *Intelligent tutor system* (ITS) v I3T. V kapitole je také popsáno zdůvodnění volby právě tohoto algoritmu.
Nejprve jsou v části 3.1 popsána specifika dat, měřených a zpracovávaných v rámci I3T. V navazující části 3.2 jsou popsány požadavky kladené na ITS v rámci programu I3T. Dále je v této části specifikován cíl inteligentního výukového systému pro tuto práci. Následně je v části 3.3 zdůvodněno, proč byl na základě typu měřených dat a požadavků kladených na ITS, zvolen právě algoritmus DQL. V poslední části 3.4 je podrobně popsán algoritmus DQL.

3.1 Data měřená v I3T

V programu I3T, respektive jeho ITS části s úlohami, se vyskytují především otevřené tvořivé úlohy s potřebou znalostí, viz 2.2.2. V některých případech je možné úlohy vyřešit i metodou pokus-omyl[1] a tedy se mohou blížit typu „*puzzle*“. Teoreticky jsou možné i další typy úloh, jako je např. výběr z možných připravených odpovědí. Takové úlohy by však plně nevyužívali možnosti I3T,

[1] Především díky okamžité odezvě a zobrazení efektu operace, kterou student prování. Díky tomu lze relativně rychle např. odhalit, které ze tří políček translace posouvá objekt po ose x i bez znalosti maticového zápisu transformací.

které spočívají především v manipulaci s objekty reálném čase. U úloh lze měřit správnost řešení a čas, který student u úlohy strávil, ať už ji vyřešil správě, či špatně, nebo řešení nedokončil. Dále je u převažujícího typu úloh možné měřit např. počet provedených operací, složitost řešení např. v počtu použitých modulů, nebo využití nápovědy. Pouze některé z těchto měření jsou v I3T zprovozněny. Rozšíření měření o další možnosti a jejich začlenění do ITS je možné provést snadno, díky přístupu popsaném v části 2.2.5. Z důvodů popsaných v předchozích větách je v rámci této práce je položen předpoklad, že z programu I3T, respektive ITS lze získávat tato data:

- Správnost řešení. V rámci úloh dostupných v I3T lze vždy získat informaci o tom, zda student úlohu vyřešil správně čí špatně. Za špatné řešení je považováno i takové, kdy student úlohu nedořeší i když některé její části mohou být vyplněny správně.
- Čas řešení. Čas, který studentovi trvalo úlohu vyřešit či odpovědět chybně, případně nedořešenou úlohu opustit.

3.2 Cíl ITS v rámci I3T

V programu I3T je většina úloh tvořivého charakteru. Student má obvykle za úkol sestavit požadovanou strukturu zapojení jednotlivých bloků a správně nastavit hodnoty parametrů v jednotlivých blocích. V těchto tvořivých úlohách je kombinováno několik požadavků na dovednosti a znalosti studenta, přičemž každý z těchto požadavků má (pravděpodobně) jiný vliv na úspěšnost a čas řešení úlohy. Každý z těchto požadavků má také (pravděpodobně) velmi odlišné možnosti, pomocí jakých akcí je v něm student schopen zvýšit svou dovednost:

- Dovednost ovládat program I3T (např. dovednost přidávat nové bloky do Pracovní plochy, měnit směr a přiblížení pohledu ve 3D scéně, využití manipulátorů a pod.). Tyto dovednosti silně ovlivňují úspěšnost a čas potřebný k vyřešení úlohy. Rychlost, jakou se student tyto dovednosti učí, je silně závislá na uživatelském rozhraní programu. Není u nich mnoho prostoru pro zlepšování a student danou dovednost buď umí nebo ne. U těchto dovedností je pravděpodobně nejefektivnější provést studenta tutoriálem, ve kterém mu budou funkce programu představeny. Další možností je ze začátku používání I3T předložit studentovi jednoduché „technické“ úlohy s podrobným popisem činností, které má v rámci úlohy provést. Při využití tutoriálu i jednotlivých úloh je otevřená otázka,

kolik z těchto dovedností studentovi představit tak, aby některé z nich nezapomněl dříve, než dojde k jejich použití.

- Znalost principu výpočtu geometrických transformací. Tyto znalosti jsou hlavním předmětem činnosti s programem I3T. Zvyšování úrovně těchto znalostí používáním I3T a plněním předkládaných úloh je zásadním předpokladem pro smysluplnost této práce. Tyto znalosti, respektive jejich nedostatek, lze u některých úloh kompenzovat delším časem na řešení. Student může zkoušet různé možnosti propojení bloků a nastavení hodnot a postupně dojít ke správnému řešení. Dalším předpokladem a předmětem výsledků této práce je, že pořadí úloh, v jakém jsou studentem plněny, má významný vliv na změnu dovedností a schopností studenta (jinými slovy učení). Některé pořadí úloh se u studentů projevuje lepším učením, než jiné pořadí úloh. Význam „lepšího učení“ je popsán odstavcích následujících po tomto seznamu.
- Znalost možností programu I3T. Typickým znakem úloh v I3T je existence více možností jejich řešení. Obvykle existuje více možností, pomocí jakých bloků a jakým jejich uspořádáním lze dosáhnout stejného výsledku. Tyto možnosti se obvykle liší svou pracností. Znalost možností I3T kombinuje předchozí dva body. Na jedné straně musí student vědět, jakým způsobem lze řešení dosáhnout z matematického hlediska, na straně druhé musí toto řešení být proveditelné v rámci I3T. Operaci může být v I3T možné provést jednoduchým (rychlým) a složitým (zdlouhavým) způsobem. Zda student nemá dostatečné potřebné matematické znalosti, nebo pouze neví, jak řešení provést, je ze znalosti času řešení obtížně rozeznatelné až nerozeznatelné. Pro jejich rozlišení by bylo třeba analyzovat uživatelské akce. Taková analýza může v rámci vnitřní smyčky (*inner loop*) vést k individualizované nápovědě studentovi.

Zvolený cíl

Vzhledem k (ne)dostupnosti reálných dat a tedy nutnosti pouze simulovat práci studentů v I3T[2], jsou v této práci položeny následující předpoklady:

1. Úspěšná práce studenta s I3T spočívá v úspěšném splnění. všech úloh v systému. Žádnou úlohu tedy nelze vynechat, nebo ponechat neúspěšně řešenou.
2. Cílem ITS je: Pro každého studenta zkrátit na minimum čas, potřebný ke splnění předpokladu 1.

[2]Simulace dat je navíc zcela vykonstruovaná, neboť není k dispozici ani menší množství dat, ze kterých by bylo možné alespoň částečně analyzovat typické vlastnosti.

Zvoleným cílem ITS pro tuto práci je tedy: **Nalézt pro každého studenta na základě jeho dosavadních odpovědí takové pořadí úloh, které povede k úspěšnému splnění všech úloh v co nejkratším čase.**

3.3 Výběr algoritmu

Pro vyřešení této úlohy byl zvolen zvolen algoritmus zpětnovazebního učení tzv. Q-učení(*Q-learning*, QL), respektive jeho rozšíření tzv. Dvojité Q-učení (*Double Q-learning*, DQL).

Tento algoritmus má pro použití v ITS několik vhodných vlastností. Dle autorů [XHJHh21] DQL nepotřebuje pro svůj běh žádný model studenta, prostředí, učení ani jiné části ITS a tak může být použit i v případech, kdy tyto modely nejsou známy. V rámci ITS, ani I3T tyto modely známy nejsou. Další dobrou vlastností je možnost započítání dlouhodobé odměny a nikoliv pouze soustředění na lokálně nejvýhodnější akci v konkrétní okamžik. Autoři [XHJHh21] porovnali výsledek dosažený metodou založenou na DQL s metodou založenou na heuristice. Výsledkem jsou lepší hodnoty při využití DQL. Algoritmus se navíc při simulaci ukázal jako stabilní i při chybě odhadnutí stavu $s(t)$. Testované velikosti chyby byly v rozsahu (1% až 10%) a velikost této chyby neměla příliš velký vliv na rychlost konvergence algoritmu.
V algoritmu je potřeba definovat jednotlivé stavy systému. Takovým stavem mohou být např. různé úrovně dovedností studenta. Stav systému však může být definován libovolně, při dodržení diskrétnosti stavů, a mohou do něj být zahrnuty různé parametry jako např. „zájem“ studenta [THYC20a]. Změnou definice stavů se nijak nezmění chod algoritmu[3] a tak lze v navazujících pracích vyzkoušet jiné možnosti definice stavů bez nutnosti jakkoliv upravovat další části systému. Zde je nutné podotknout, že, s rostoucím množstvím stavů, rostou také požadavky na množství dat. V rámci I3T tedy potřebné množství studentů a jejich odpovědí na úlohy. DQL je také vhodný pro následující možnou situaci: *ITS již po nějakou dobu běží a jsou v něm natrénované hodnoty. Náhle se změní obsah školního vzdělávání a ITS začnou používat studenti s odlišnými znalostmi a tedy odlišnými vhodnými doporučeními ze strany ITS. Tato změna znalostí nebude zaznamenatelná v rámci definice stavů, tedy dřívější a současný student budou vykazovat jiné chování i když v rámci definice stavů budou ve stejném stavu.* S takovou situaci dokáže DQL pracovat a po určité době samovolně přeučit své hodnoty tak, aby jeho doporučení vyhovovala současným studentům.

[3]Při redefinici stavů je potřeba algoritmus spustit znovu od začátku. Navázat při jedné definici stavů na výsledky běhů algoritmu při jiné definici stavů není jednoduchá a mnohdy není možné.

Celkový provoz systému a měření či odhadování hodnot parametrů je zatíženo mnoha nepřesnostmi. Studentovu úroveň dovedností neznáme přesně a pouze ji odhadujeme na základě výsledků práce se studijními materiály, tedy pracujeme pouze s odhadnutým tzv. stavem systému $\tilde{s}(t)$ na místo skutečného stavu $s(t)$. Na studenta mohou v průběhu práce působit externí rušivé vlivy, které není možné v ITS postihnout. Tyto vlivy mohou vést k výkyvům výkonu studenta i při konstantní úrovni dovedností. Stejný studijní materiál může mít z rozličných důvodů[4] různý vliv na různé studenty. V takovém systému je výhodnější využít, jak navrhují i autoři [THYC20a], tzv. Dvojité Q-učení (*Double Q-learning*, DQL) [HGS16] popsaný blíže v části 3.4.1, který spočívá v souběžném běhu dvou QL algoritmů. Akce vybírané pro další postup v prvním z nich, jsou vybírány na základě Q-hodnoty z druhého. Stejně i naopak, jsou akce v druhém běhu algoritmu vybírány na základě Q-hodnoty z prvního běhu. V průběhu práce ITS může vést akce náhodně, vlivem externích vlivů, k extrémnímu výsledku. Např. student opustí systém při zapnuté úloze a jeho čas řešení tak bude extrémně dlouhý. Pokud by součástí výpočtu tzv. odměny $r(t)$ byla i doba řešení, byla by takové akci přiřazena velmi nízká hodnota. V případě jednoduchého QL algoritmu by její „oprava“ trvala dlouho[5]. Při DQL algoritmu je však hodnota „opravena“ rychleji, neboť akce je vybírána na základě Q-hodnoty z druhého běhu algoritmu, který není tímto jedním extrémním výsledkem ovlivněn.

3.4 Podrobný popis algoritmu Dvojitého Q-učení, vybraného pro I3T

Algoritmus Q-učení (*Q-learning*, QL) i jeho rozšířená verze Dvojité Q-učení (*Double Q-learing*, DQL) jsou založeny na tzv. zpětnovazebním učení (*reinforcement learning*, RL). Princip RL je následující: V rámci algoritmu je pro aktuální okamžik, respektive krok t, uložen tzv. stav $s(t)$. V každém stavu $s(t)$ lze v aktuálním kroku t provést jednu akci $a(s,t)$. Akce $a(s,t)$ je vybrána z množiny akcí $A(s)$ možných v daném stavu s, tedy $a(s,t) \in A(s)$, kde $a(s,t)$ je akce vybraná ve stavu s v kroku t a $A(s)$ je množina akcí možných ve stavu s. Vybráním a provedením akce $a(s,t)$ se stav změní na nový stav $s(t+1)$ v kroku $t+1$. Dále je provedením akce $a(s,t)$ v kroku t a tedy přechodem do stavu $s(t+1)$ získána tzv. odměna (*reward*) $r(t)$. V tomto nastavení lze rekurzivně definovat tzv. Bellmanovu rovnici pro tzv. Funkci hodnoty stavu

[4]Student nemusí znát některé pojmy, které se v úloze vyskytují a zároveň nejsou předmětem dané úlohy. Příklad z praxe: Student střední školy neznal pojem „vidle“. Pokud by úloha zněla např. „Kolik vidlí potřebuje koupit hospodář, který má 2 syny a 5 pomocníků, pokud již troje vidle má?“ Nemusí student vědět, jak úlohu vyřešit i když sčítání a odčítání perfektně ovládá, neboť neví, kolik vidlí potřebuje pro práci jeden člověk.

[5]Při použití ϵ-greedy strategie by byla tato akce v daném stavu vybrána pouze s pravděpodobností ϵ/J, kde J je počet úloh.

- V hodnoty (*Value function*). Systém se může v daný okamžik, respektive krok t, nacházet pouze v jednom stavu s, tedy $s = s(t)$ Bellmanova rovnice však platí pro všechny možné stavy v libovolný okamžik a tedy v jejím zápisu nejsou stavy označeny krokem t:

$$V(s) = \max_a \{r(s,a) + \gamma V(s')\} \tag{3.1}$$

$$s' = T(s,a) \quad , \tag{3.2}$$

kde s, s' : stav
a : akce
$V(s)$: V-hodnota stavu s
$r(s,a)$: odměna za provedení akce a ve stavu s
γ : $\in \langle 0, 1\rangle$ tzv. discount faktor, určuje vliv akcí vykonaných v pozdějších krocích
$T(s,a)$: přechodová funkce určující, jaký bude nový stav s' po vykonání akce a ve stavu s

Řešením Belmanovy rovnice 3.1 získáme hodnotu funkce hodnoty stavu (V-hodnoty) pro každý stav s a také optimální strategii (*policy*) π^*:

$$a^*(s) = \pi^*(s) \quad , \tag{3.3}$$

kde a^* : optimální akce pro daný stav s
$\pi^*(s)$: strategie (*policy*) určující optimální akci ve stavu s
s : stav

3.4.1 Q-učení a Dvojité Q-učení

Dva z možných postupů pro řešení Bellmanovy rovnice jsou tzv. Q-učení (*Q-learning*, QL), respektive tzv. Dvojité Q-učení (*Double Q-learning*, DQL), popsané v této části.

Mějme tzv. strategii (*policy*) π. Strategie π předepisuje akci $a(t) \in A(s(t))$, kde $s(t)$ je stav, ve kterém se systém nachází v kroku, respektive okamžiku t a $A(s(t))$ je množina možných akcí v tomto stavu. Strategie π předepisuje tuto akci pro každý možný stav s systému. Pokud je π dáno, je možné přiřadit tzv. Q-hodnotu[6] každé dvojici $(s(t), a(t))$, kde $s(t)$ je stav v kroku t a $a(t)$ je

[6]Označení Q je v literatuře i této kapitole využíváno ve dvou významech. Prvním je **Q-matice** o rozměrech $K \times J$, kde K je počet témat (*knowledge component*) a J je počet úloh v systému. Q-matice zachycuje vztah mezi úlohami a tématy. Druhým významem je **Q-hodnota**, která je přiřazena každé dvojici stav-akce v rámci algoritmu *Q-learning*.

akce proveditelná ve stavu $s(t)$:

$$Q_\pi^t(s(t), a(t)) = E_\pi[\sum_{t'=t}^{T} R(t')]|s(t), a(t) \quad , \tag{3.4}$$

kde Q_π^t : Q hodnota stavu a funkce pro danou strategii π v kroku t
$s(t)$: stav systému (např. dovednosti studenta) v korku t
$a(t)$: akce (např. navržení studijního materiálu) v kroku t
$R(t)$: (očekávaná) odměna v kroku t, tedy po akci $a(t)$
E_π : očekávaná hodnota vzhledem k dané strategii π
T : číslo posledního kroku, může být konečné i nekonečno.

Q-učení

Základní postup Q-učení (*Q-learning*, QL) [WD92] je založen na nalezení stabilní Q-hodnoty v rovnici (3.4) pro všechny dvojice stav-akce. V QL jsou iterativně procházeny stavy a akce a upravovány Q-hodnoty dle vztahu (3.5). Ve zmíněné práci bylo prokázáno, že, při dobře zvoleném parametru učení α, algoritmus konverguje k optimální hodnotě. Pro konvergenci musí být prostor stavů a akcí dostatečně prozkoumán (*explore*):

$$\begin{aligned} Q_{new}(s(t), a(t)) = Q(s(t), a(t)) \\ + \alpha(r(t) + \gamma * \max_a\{Q(s(t+1), a)\} - Q(s(t), a(t))) \quad , \end{aligned} \tag{3.5}$$

kde Q_{new} : nová Q hodnota pro daný stav a akci
Q : aktuální Q hodnota pro daný stav a akci
$s(t)$: stav v kroku t, např. dovednosti studenta
$a(t)$: akce v kroku t, např. doporučení studijního materiálu
$r(t)$: odměna získaná v kroku t, po akci $a(t)$
α : $\in (0, 1\rangle$, parametr učení, velikost učícího kroku
γ : tzv. *discount* faktor
upravuje hodnotu odměn získaných v pozdějších krocích.

Poznámka: „Učící krok“ α v rovnici 3.5 výše je pouze název parametru. S „krokem“ t je shoda v názvu nechtěná.

Konvergující iterativní postup je například tzv. **ϵ-hladový** (ϵ-*greedy*), při kterém je s pravděpodobností ϵ zvolena akce náhodně. S pravděpodobností $1-\epsilon$ je zvolena akce s nejvyšší Q-hodnotou v daném stavu. Hodnota ϵ obvykle klesá s množstvím provedených akcí v daném stavu, nebo s počtem kroků.

Pro výslednou strategii je poté v každém stavu zvolena ta akce, která má nejvyšší Q-hodnotu v daném stavu:

$$a^*(t) = \underset{a}{\operatorname{argmax}}\{Q(s(t), a)\} \tag{3.6}$$

Dvojité Q-učení

V práci [Has10] autor ukázal, že lepších výsledků lze dosáhnout algoritmem Dvojitého Q-učení (*Double Q-learning*, DQL). Zásadním rozdílem oproti jednoduchému QL algoritmu je držení dvou Q-hodnot pro každou dvojici stav-akce v DQL, oproti jedné hodnotě pro každou dvojici stav-akce v QL. Pokud označíme dvě sady Q hodnot v DQL Q^A a Q^B, můžeme algoritmus DQL zapsat:

Algoritmus 1: Dvojité Q-učení (*Double Q-learning*). Pro přehlednější zápis je v algoritmu notace: $s' = s(t+1)$, $s = s(t)$, $r = r(t)$, $a = a(t)$. Řádek 5 je zjednodušený. Není v něm zahrnuta ϵ-hladová strategie, která je v I3T použita viz 3.4.1. Algoritmus je převzatý z [Has10] a doplněný o upřesnění výběru akce na řádku 5 a inicializaci všech hodnot.

1 Vytvoř Q^A, Q^B pro každou dvojici stav-hodnota;
2 $Q^A(s,a) = Q^B(s,a) = 0 \quad \forall s \forall a$;
3 Urči počáteční stav s ;
4 **while** *uživatel pracuje s ITS* **do**
5 Vyber akci $a \leftarrow \underset{a}{\operatorname{argmax}}[max(Q^A(s,*), Q^B(s,*))]$;
6 Ulož získanou odměnu r a nový stav s' ;
7 Vyber (např. náhodně) jednu z možností UP(A) a UP(B);
8 **if** *UP(A)* **then**
9 Ulož $a^* \leftarrow \underset{a}{\operatorname{argmax}}[Q^A(s',*)]$;
10 $Q^A(s,a) \leftarrow Q^A(s,a) + \alpha(s,a)(r + \gamma Q^B(s', a^*) - Q^A(s,a))$;
11 **else if** *UP(B)* **then**
12 Ulož $b^* \leftarrow \underset{a}{\operatorname{argmax}}[Q^B(s',*)]$;
13 $Q^B(s,a) \leftarrow Q^B(s,a) + \alpha(s,a)(r + \gamma Q^A(s', b^*) - Q^B(s,a))$;
14 **end**
15 $s \leftarrow s'$;
16 **end**

Parametr γ v Algoritmu 1 se nazývá „*discount* faktor“. Jeho nastavení je možné v rozsahu $\langle 0, 1\rangle$. Hodnota parametru určuje, jak velký vliv budou mít na výslednou Q hodnotu následující akce, respektive odměny. Při hodnotách blízkých 0 je algoritmus „krátkozraký“ a ve výsledné Q hodnotě se projeví

především aktuální akce/odměna. Naopak při hodnotách blížících se k 1 je algoritmem sledován dlouhodobý horizont a výslednou Q hodnotu silně ovlivní i akce, respektive odměny vykonané dlouho po aktuální akci. Další možnost představili autoři [VRD15]. Hodnotu γ nenastavili statickou, ale měnící se v průběhu učení. Tento postup vedl k významnému snížení potřebných kroků k dosažení stabilní hodnoty.

Parametr α v Algoritmu 1 se nazývá „velikost učícího kroku“ (*learning rate*). Jak zmiňuje také autor [Has10], dobré výsledky vykazuje nastavení polynomiální hodnoty α vzhledem k počtu úprav hodnoty daného stavu.

3.4.2 Aplikace a specifika Dvojitého Q-učení (*Double Q-learning*) v Inteligentním výukovém systému

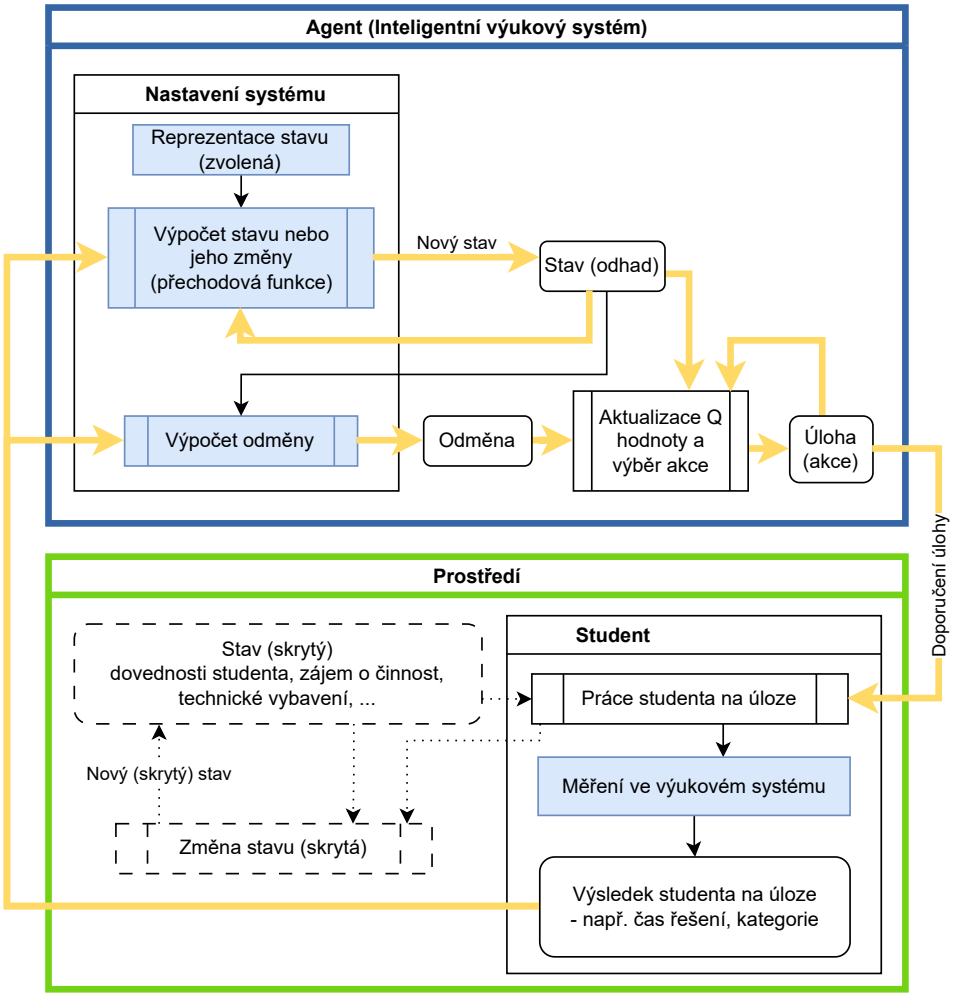
Scénář zpětnovazebního učení[7] je vcelku přímočarý. ITS se v daný moment, respektive v daný krok, nachází v určitém stavu $s(t)$. Tento stav typicky zahrnuje úroveň dovedností studenta, ale může zahrnovat i další parametry, jako „zájem“ studenta [THYC20a] a nebo údaje o věku, pohlaví a pod. ITS provede v daný krok akci $a(t)$, tedy doporučení/předložení studijního materiálu $l \in L$ studentovi, kde L je množina všech dostupných studijních materiálů. Student s předloženým materiálem pracuje a na základě jeho výsledku práce získá ITS odměnu (*reward*) $r(t)$. Do výpočtu odměny lze zahrnout libovolné parametry a jejich kombinaci v závislosti na konkrétním ITS. Typickými parametry odměny jsou dosažené skóre na úloze či čas, který student prací s materiálem strávil. Za předpokladu, že práce se studijním materiálem mění dovednosti studenta se tímto dostane ITS do nového stavu $s(t+1)$ v kroku $t+1$, spolu s informací $r(t)$ o úspěšnosti předchozí akce. Cílem zpětnovazebního učení a tedy ITS je, nalézt co nejlepší strategii (*policy*) π, která maximalizuje celkovou očekávanou odměnu $R_\pi = E(\sum_{t=0}^{T} r(t))$ za daný počet kroků. Tento počet kroků může být konečný (vyučovací hodina, semestr) i nekonečný.
Takový scénář lze zapsat ve formě Markovova rozhodovacího procesu (*Marcov decision proces*, MDP), viz Obrázek 3.2 a řešit např. pomocí metody Q-učení (*Q-learning*) či Dvojitého Q-učení (*Double Q-learning*) [THYC20a], [XHJHh21], [KMR18].

Autoři [THYC20a] zmiňují, že v rámci adaptivního učení je vhodné zahrnout do vyhodnocení stavu, akce a odměny i další faktory, než pouhou úroveň dovedností či čas. Při lidském učení hrají roli také různé studijní styly

[7] učením je myšleno „učení“ Inteligentního výukového systému (ITS), který materiál je pro studenta v aktuální chvíli nejvhodnější

studentů, zájem o téma a další faktory, jejichž zahrnutí vyžaduje interdisciplinární spolupráci.

Inteligentní výukový systém (*Inteligent tutor systém*, ITS) se vyznačuje vysokou mírou možných nastavení. Tato nastavení závisí na rozhodnutí tvůrců ITS a neexistuje postup na nalezení „správného řešení“. Hledané „správné řešení“ není jednoznačné a vždy záleží na konkrétních požadavcích na činnost ITS a kontextu ve kterém je ITS používán. Schematický Obrázek 3.1 zobrazuje průběh QL, nebo DQL v ITS. V částech níže jsou popsány hlavní tři aspekty pro nastavení systému.



Obrázek 3.1: Přehled Q-učení (*Q-learning*) v Inteligentním výukovém systému. Podrobný popis obrázku je zapsán níže, v části 3.4.2.

Na Obrázku 3.1 je rámcově shrnuto využití Q-učení (*Q-learning*) v Inteligentním výukovém systému (*Intelligent tutor system*, ITS) s důrazem na zobrazení návaznosti jednotlivých částí ITS. Vyjádřeno terminologií užívanou v QL je ITS tzv. „agent“ (modrý rámeček), který interaguje s „prostředím“

(zelený rámeček). Prostředí je v kontextu ITS student, který ITS používá. Dále jsou součástí prostředí okolnosti, za kterých student používá ITS. Např. zda pracuje v rámci organizované výuky, nebo samostatně. Zda pouze procvičuje, nebo plní důležitý úkol a pod. Student a okolnosti jsou v terminologii QL zachyceny jako konkrétní „stav“. Tento stav je skrytý, respektive částečně pozorovatelný, což je v obrázku naznačeno čárkovanou čarou. Práce studenta na úloze je závislá na tomto stavu. Tím, že student pracuje na zadané úloze se tento stav změní. Jak se stav změní je, stejně jako stav samotný, skryté. Práci studenta na úloze můžeme sledovat skrze měření ve výukovém systému (modrý rámeček v rámečku „Student“ je součástí ITS, z důvodu přehlednosti je však zakreslen na jiném místě). V ITS máme tedy k dispozici pouze odhad skutečného stavu. Tento odhadovaný stav je závislý na zvolené reprezentaci, kterou je v ITS stav modelován. Na základě aktuálního odhadovaného stavu a výsledku měření je tzv. přechodovou funkcí odhadnut nový stav. Výsledek měření je také využit k výpočtu odměny. Odměna kromě výsledku měření závisí také na odhadovaném stavu (např. zda je stav konečný viz rovnice (4.1) či (4.2)). Odhadovaný stav, doporučená úloha a vypočtená odměna jsou využity k aktualizaci odpovídající Q-hodnoty (viz řádky (10 a 13)). Odhadovaný stav a Q-hodnoty jsou dále využity k výběru další úlohy (viz rovnice 3.6). Při obvyklém použití QL jsou stav prostředí a odměna za akci pevně stanoveny prostředím ve kterém je QL nasazeno. I v jiných problémech může být nutné volit reprezentaci stavu, ale odměna za akci je obvykle dána „pravidly hry“. Rozdílem v ITS je, že způsob reprezentace stavu, ale i odměna za provedenou akci je závislá na nastavení systému.

Odměna (*reward*)

Zásadním místem v QL i DQL algoritmu je volba odměny po provedení akce v daném stavu. V obvyklém scénáři je odměna dána prostředím ve kterém je algoritmus nasazen. V rámci ITS je však odměna, respektive její výpočet, závislá na vůli tvůrců ITS a cíli, ke kterému má použití ITS vést. Viz Obrázek 3.1. Výpočet odměny a parametry, které jsou do výpočtu zahrnuty mohou být libovolné. Hodnota odměny má zásadní vliv na výsledek algoritmu a změnou výpočtu odměny je možné výrazně ovlivnit získanou strategii (*policy*). Např.

[THYC20a] používají k výpočtu odměny funkci:

$$r(t) = \begin{cases} -1 & : \text{doporučení studijního materiálu } l_1, ...l_N \\ 0 & : \text{doporučení ukončení činnosti} \\ r(T) & : \text{konečný stav } t = T \end{cases} \tag{3.7}$$

$$r(T) = \Phi \sum_{k=1}^{K} w_k s_k(T) \quad , \tag{3.8}$$

kde $r(t)$: odměna v kroku t
$r(T)$: odměna v konečném kroku $t = T$
l_i : i-tý studijní materiál
Φ : škálovací parametr
k, K : téma (KC) k, respektive jejich počet v systému
w_k : váha (důležitost) k-tého tématu
s_k : hodnota k-té dovednosti studenta

při jejímž použití je kladen důraz na vážený stav dovedností studenta při ukončení studia a počet doporučených studijních materiálů.

Stejní autoři zmiňují také jiné možnosti jako např. funkci, která zahrnuje zlepšení dovedností v každém kroku:

$$r(t) = \sum_{k=1}^{K} w_k(s_k(t) - s_k(t-1)) \quad , \tag{3.9}$$

kde $r(t)$: odměna v kroku t
k, K : téma (KC) k, respektive jejich počet v systému
w_k : váha (důležitost) k-tého tématu
s_k : hodnota k-té dovednosti studenta

Dle [XHJHh21] je cílem ITS obvykle snaha zlepšit dovednosti studenta na požadovanou úroveň v co nejkratším čase. Pro zahrnutí času do výpočtu odměny uvádí např. vztah:

$$r(t) = \sum_{k=1}^{K} w_k(s_k(t+1) - s_k(t)) + d * \tau(t) \quad , \tag{3.10}$$

kde $r(t)$: odměna v kroku t
k, K : téma (KC) k, respektive jejich počet v systému
w_k : váha (důležitost) k-tého tématu
$s_k(t)$: hodnota k-té dovednosti studenta v kroku t
d : koeficient, kterým lze modulovat vliv doby řešení na odměnu
$\tau(t)$: doba řešení úlohy doporučené v kroku t

Stav a akce

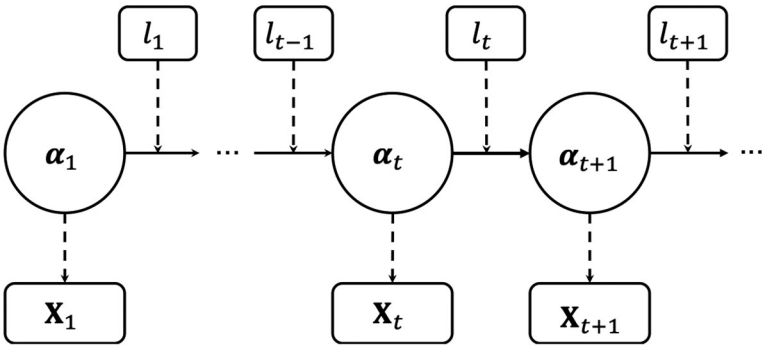
Akce $a \in A$ nemusí reprezentovat pouze studijní materiál obsažený v ITS. Množina A může obsahovat také pokyn k ukončení práce [THYC20a] nebo např. informaci o tom, že systém neobsahuje žádnou vhodnou úlohu a proto doporučuje činnost mimo I3T, např studium literatury.

Autoři [XHJHh21] stav formálně zapisují jako vektor $s(t) = (s_1(t), s_2(t), ..., s_K(t))$ v čase t při K tématech. Počet témat[8] (*knowledge component*, KC) K může být obecný. Dovednost d, jako úroveň studenta pro dané téma, může být zaznamenána ve více úrovních znalosti. Obecně může mít každá dovednost K_d úrovní. Např. pokud $K_1 = 2$, znamená to, že dovednost 1 lze pouze „umět / neumět". Pokud $K_2 = 4$, můžeme úrovně dovednosti 2 interpretovat např. jako „neumí / začátečník / pokročilý / expert". Jak ukázali např. autoři práce [XHJHh21], lze z takových dovedností vytvořit i libovolnou hierarchii prerekvizit. Uvádí také příklad, jak lze libovolnou hodnotu K_d převést na množinu dovedností, kde každá z těchto nových dovedností je binární. $K_{d^{new}} = 2 \quad \forall d^{new}$. Použití pouze binárních dovedností je dle autorů výhodné, neboť odpovídá standardizovaným testům CDM (*Cognitive diagnosis models*), široce využívaných v oblasti psychometrie. Druhou možností je podle autorů [YXJZ18] využití spojitých hodnot dovedností.

Autoři [XHJHh21] také zmiňují, že hierarchická struktura dovedností je důležitým aspektem změny úrovně dovedností. Nezahrnutí hierarchické struktury dovedností může značně ovlivnit efektivitu systému [TWC+19]. Hierarchická struktura se dle autorů objevuje v mnoha různých oblastech výuky. Hierarchii témat se dle [XHJHh21] věnuje tzv. *attribute hierarchy method* (AHM), která zahrnuje čtyři možné případy: lineární, konvergentní, divergentní a nestrukturovaný vztah mezi jednotlivými tématy.

Pro uložení takové struktury témat a úloh zmiňují autoři [XHJHh21] využití Q-matice, která koresponduje také s používanými CDM testy. Pokud ve výukovém systému počítáme s K tématy a je v něm J úloh, bude Q-matice o velikosti $K \times J$. Zprvu je obvykle matice vyplněna expertem v oboru (učitelem), dále může být upravována automaticky na základě naměřených dat. Jak uvádí autoři [THYC20a], obzvláště pro systémy s velkým množstvím stavů a možných akcí v nich je často nepraktické držení všech Q-hodnot v paměti. Autoři využili relativně jednoduchou hlubokou neuronovou síť (*deep neural network*, DNN) s dvěma skrytými vrstvami a dvěma ReLU vrstvami,

[8] Počet témat či témata jsou vlastnostmi ITS jako celku. Úroveň dovednosti pro každé téma je přiřazena studentovi. Zároveň je také všem studijním materiálům přiřazena podmnožina témat. Práce studenta s konkrétním materiálem vede u studenta ke změně úrovně dovedností v daných tématech.



Obrázek 3.2: Inteligentní výukový systém jako MDP. Obrázek zobrazuje průběh interakce studenta a ITS. student se nachází ve stavu $s(t)$ (na obrázku α_t). Studentovi je předložen studijní materiál l_t a tím se stav studenta změní na s_{t+1}. Stav studenta je částečně pozorovatelný X_t, skrze interakci se studijním materiálem (úlohy, testy). Zdroj obrázku [XHJHh21].

pro výpočet Q-hodnoty pro daný (odhadovaný) stav $\tilde{s}(t)$ a čas t. Autoři uvádí, že vstupem DNN mohou být i další informace. V práci [THYC20a] např. zahrnuli „snahu“ (*learning interest*) do výpočtu Q-hodnoty skrze DNN.

Změnu stavu $s(t)$ po práci se studijním materiálem, tedy po doporučené akci $a(t)$ lze zapsat $s(t) \times a(t) \rightarrow s(t+1)$ a pravděpodobnost tohoto přechodu jako:

$$P(s(t+1) = \tilde{s} | s(t) = s, a(t) = a) \quad , \tag{3.11}$$

kde P : pravděpodobnost přechodu do nového stavu
$s(t)$: stav systému (např. dovednosti studenta)
$\tilde{s}$: odhad nového stavu
$a(t)$: akce (doporučení studijního materiálu)

Tuto podmíněnou pravděpodobnost, respektive rozložení pravděpodobnosti nazývají autoři [YXJZ18] přechodovou funkcí (*transition kernel*) a zmiňují, že ji lze využít i při spojitých hodnotách dovedností (hustota pravděpodobnosti) i při diskrétních hodnotách. Taková formulace odpovídá formulaci MDP. Zároveň můžeme získat výsledek práce studenta na předložené úloze v čase $X(t)$ viz Obrázek 3.2. Stav studenta $s(t)$ je, jak zmiňují autoři [YXJZ18] a [THYC20a], skrytý (*latent*), respektive pouze částečně pozorovatelný. Máme k dispozici pouze výkon studenta na předložených studijních materiálech. To dle autorů přirozeně převádí model na skrytý Markovův model (*Hidden Marcov model*, HMM), případně částečně pozorovatelný Markovův model (*Partialy observable Marcov decision model*, POMDP). Skrytý stav viz také na Obrázku 3.1.

Autoři [THYC20a] dodávají, že výsledek studenta na dané úloze nezáleží pouze na dovednostech studenta či dalších parametrech zahrnutých do stavu

$s(t)$, ale také na vlastnostech úlohy. Lze jej pouze odhadnout na základě odpovědi studenta na předložené úlohy:

$$\begin{aligned} Y_j \sim \quad & h_{j,s(t)}(y) \quad , \\ \text{kde } Y_j : \quad & \text{výsledek na úloze } j \\ s(t) : \quad & \text{stav systému (např. dovednosti studenta)} \\ h_{j,s(t)}(y) : \quad & \text{rozložení odpovědí pro daný stav a úlohu,} \\ & \text{závisí také na formě úlohy/testu a pod.} \end{aligned} \tag{3.12}$$

Autoři [YXJZ18] zavádějí tzv. S-matici, která, podobně jako Q-matice, zaznamenává vztah mezi tématy a úlohami. Q-matice zachycuje dovednosti (úroveň studenta v daném tématu), které jsou potřeba ke splnění úlohy. S-matice zaznamenává, na které dovednosti má úloha vliv vzhledem k učení, tedy ke změně úrovně dovedností studenta. Autoři uvádějí, že při známých hodnotách v S-matici, mohou tyto informace značně snížit počet parametrů v MDP modelu, neboť sníží počet možných nových stavů studentových dovedností po práci na dané úloze.

Dále také autoři [XHJHh21] představují možnost sledování změny v dovednostech studenta a vytvoření takzvané „vzdělávací cesty“ (*learning path*). Vzdělávací cesta je postupné zlepšování jednotlivých dovedností skrze práci s předkládanými studijními materiály. Na vzdělávací cestu lze dle autorů nahlížet jako na složenou ze dvou částí (psychometrického modelu a MDP), případně jako na částečně pozorovatelný MDP (*partialy observable Marcov decision process*, POMDP). Lze také zahrnout celkový čas, který studentovi zabralo dosažení požadované úrovně dovedností.

Kapitola 4

Realizace Inteligentního výukového systému v I3T

V této kapitole je podrobně popsána výsledná realizace Inteligentního výukového systému (*Intelligent tutor system*, ITS). V navazující části jsou popsány jednotlivé části ITS, dále detaily použitého algoritmu DQL a konkrétní nastavení parametrů. V dalších částech je popsán způsob, jakým jsou získávána data pro zpracování a také způsob jejich uložení.

4.1 Popis systému a nastavení DQL algoritmu v I3T

Následující popis odpovídá Obrázku 4.1. Při prvním spuštění I3T s inteligentním výukovým systémem je k dispozici „Nabídka úloh v ITS“[1]. Dále „Student pracující s ITS“ jakožto uživatel programu, který v tuto chvíli nemá zaznamenaný žádný pokus o splnění některé z úloh. Tím se systém nachází v úvodním stavu $s(t) : t = 0$. Pro tento stav mají všechny akce v počátečním stavu $a(t) : t = 0 \quad a(0) \in A(s(0))$ přiřazeny své počáteční „Q-hodnoty“[2] hodnoty, kde $A(s(0))$ je množina povolených[3] akcí ve stavu $s(0)$. Akce téměř

[1] Uvozovky v tomto odstavci odkazují na konkrétní bloky v Obrázku 4.1. Obsah uvozovek odpovídá názvu konkrétního bloku v obrázku.

[2] Při druhém a dalším spuštění jsou již počáteční hodnoty určené předchozím během programu. Hodnoty se tedy neresetují.

[3] V počátečním stavu např. není povolena akce „Žádná vhodná úloha k dispozici“.

vždy odpovídá doporučení úlohy. Termíny *akce* a *úloha* jsou proto téměř zaměnitelné. V místech textu, kde je důležitý jejich rozdíl je na tuto skutečnost upozorněno. Podrobnější popis je v části 4.1.3.

Na základě „Q-hodnot“ je vybrána úloha pro studenta, viz také řádek 5 v algoritmu 1. „Student řeší úlohu“, která mu byla vybrána. V závislosti na činnostech studenta a „Nastaveném měření v ITS“[4] jsou „Naměřena hrubá data“. Tato data jsou uložena v bloku „Uložení záznamu pokusu“ do „Databáze hrubých dat všech pokusů“[5]. Hrubá data jsou zpracována v bloku „Výpočet výsledku“, který například kategorizuje studentův výkon při řešení úlohy, viz také 4.1.1. Výstupem bloku je „Výsledek pokusu“ studenta, který je, obdobně jako hrubá data, uložen do „Databáze výsledků pokusů“.

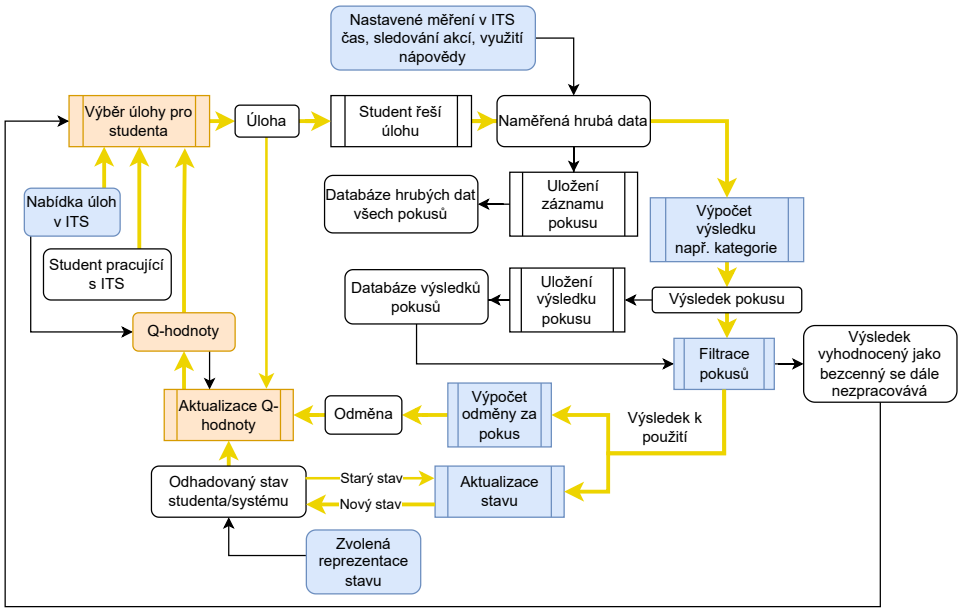
V závislosti na předchozích, obvykle nedávných, výsledcích studenta, je aktuální výsledek zkontrolován ve „Filtraci pokusů“ viz 2.2.5. Filtrace je zavedena pro minimalizaci možnosti, že náhodný extrémní výsledek studenta ovlivní hodnoty v DQL, např. pokud by student nechal program omylem spuštěný. Pokud výsledek filtrem neprojde, přechází se rovnou k výběru další úlohy. Pokud výsledek projde zmíněným filtrem je použit pro dva účely. Za prvé je použit k „Výpočtu odměny za pokus“ (*reward*) pro algoritmus DQL. Za druhé je použit k „Aktualizaci stavu“ systému. Například k odhadu změny dovedností studenta. Tento „Odhadovaný stav systému“ je závislý na „Zvolené reprezentaci stavu“, viz 4.1.3. Na základě aktuálních „Q-hodnot“, „Úlohy“ zvolené na začátku procesu, starého i nového „Odhadovaného stavu systému“ a získané „Odměny“ je „Aktualizována Q-hodnota“, viz řádky 10 a 13 v Algoritmu 1. V tuto chvíli se celý cyklus opakuje.

4.1.1 Kategorizace výsledku

Rozhraní mezi algoritmem doporučujícím další úlohu a hrubými daty naměřenými u každého pokusu studenta o vyřešení úlohy, může být značně komplikované. Různé typy otázek 2.2.3 generují různé typy naměřených dat, které vyžadují různé zpracování. Na základě doporučení, viz část 2.2.5, je definováno následujících šest kategorií výsledku studentova pokusu o vyřešení úlohy. V reálných datech lze kategorizaci provést např. na základě rozložení času odpovědí všech studentů, nebo využití nápovědy. Vzhledem ke zpracování

[4] Jaké činnosti a události je možné zaznamenávat v tzv. Loggeru, viz také část 1.3. Kromě správnosti řešení úlohy a času řešení to mohou být i podrobnější informace jako např. poloha či kliknutí myši, záznam interakcí s moduly I3T a pod.

[5] Vzhledem k omezenému množství uživatelů, obtížnosti sběru dat a velkému množství dat potřebných pro trénink algoritmů strojového učení, je vhodné ukládat i hrubá data pro případné další zpracování.



.

Obrázek 4.1: Přehled Q-učení *Q-learning* v I3T. Podrobný popis obrázku je v úvodu části 4.1. V obrázku[6] je, podobně jako v obrázku 3.1, rámcově shrnuto využití Q-učení (*Q-learning*, QL) v I3T s důrazem na zobrazení návaznosti jednotlivých částí ITS. **Žluté šipky a žlutě zbarvené bloky** jsou součásti hlavního cyklu, který realizuje algoritmus DQL 3.4.1. Žluté bloky jsou neměnné, stanovené vybraným DQL algoritmem. **Modře zbarvené bloky** jsou ty, u kterých je možné změnit jejich konkrétní nastavení, respektive jejich nastavení je závislé na tvůrcích ITS. Popis bloků v rámci I3T a této práce je popsáno v části 4.1. **Černé šipky a nepodbarvené bloky** označují „technické“ části systému, které jsou nutné nebo výhodné pro jeho běh. Případně jsou to bloky, které zaznamenávají nevyhnutelný výsledek ostatních bloků

pouze simulovaných dat, je v této práci zavedena kategorizace na základě pravděpodobnosti úspěšného vyřešení úlohy, viz také vztah 4.3.

C^+ Správná odpověď při pravděpodobnosti úspěšného vyřešení více než 0.8

C^0 Správná odpověď při pravděpodobnosti úspěšného vyřešení mezi 0.2 a 0.8

C^- Správná odpověď při pravděpodobnosti úspěšného vyřešení méně než 0.2

I^+ Chybná odpověď při pravděpodobnosti úspěšného vyřešení více než 0.8

I^0 Chybná odpověď při pravděpodobnosti úspěšného vyřešení mezi 0.2 a 0.8

I^- Chybná odpověď při pravděpodobnosti úspěšného vyřešení méně než 0.2, nebo např. odpověď „Nevím, zobraz další úlohu“

Není nutné, aby bylo u všech úloh možné dosáhnout všech kategorií. Pokud

např. reálná úloha neposkytuje nápovědu, nebude v ní možné odpovědět v kategorii C^- a pod. Systém kategorií je využit jako rozhraní mezi úlohami a ITS. Podle jakého klíče bude odpověď zařazena do jedné z kategorií a zda budou využity všechny kategorie, je tedy vždy na rozhodnutí tvůrce úlohy.

4.1.2 Nastavení parametrů

Hodnota ϵ v ϵ-hladovém postupu je závislá na počtu navštívení daného stavu s, viz část 3.4.1. Použitý vztah je $\epsilon(s) = 1/\sqrt{n(s)}$, kde $n(s)$ je počet navštívení stavu s (včetně aktuálního navštívení stavu, pro první navštívení je tedy $n(s) = 1$). Je možné, že studenti využívající I3T budou mít v budoucnu jiné studijní předpoklady a potřeby, než aktuální studenti. Vzhledem k tomu by mohlo být vhodné v blíže nespecifikovaný okamžik (např. na začátku semestru) nastavit u všech stavů hodnotu $n(s) = 0$ tak, aby se opět zvýšila šance pro náhodný výběr akce. Takovou operaci a její efekt však bude nutné prozkoumat.

Hodnota nastavená **parametru** γ, popsaném v odstavci 16, je v simulacích $\gamma = 0.99$, neboť výsledná odměna je v podstatě přiřazena až na konci výukové cesty, sestávající z mnoha úloh, a cílem je zaznamenat tuto výslednou hodnotu i u prvních akcí. Dynamické nastavení γ dle [VRD15], je možné i v ITS. Například v závislosti na počtu úloh které má student splněné. Takové nastavení by vedlo k tomu, že ze začátku používání I3T studentem, by bylo v rámci systému usilováno o rychlý krátkodobý zisk, zatímco později by úlohy vedly k dlouhodobějším cílům výuky. Takové nastavení by mohlo vést k lepší motivaci studentů (zprvu by se systém snažil o rychlé, krátkodobé úspěchy, což podporuje motivaci), obzvláště ze začátku využívání I3T. Pokusy s tímto nastavením jsou ponechány navazujícím pracím.

Parametr α, viz odstavec 16, je nastaven dle doporučení v [HGS16]. Tedy $\alpha_t(s,a) = 1/n_t(s,a)^{0.8}$, kde $n_t(s,a)$ je počet změn dvojice stav s - akce a. Dále $n_t(s,a) = n_t^A(s,a)$, pokud je upravována hodnota Q^A, nebo $n_t(s,a) = n_t^B(s,a)$, pokud je upravována hodnota Q^B. Hodnoty $n_t^A(s,a)$ a $n_t^B(s,a)$ jsou tedy uloženy pro každou z Q hodnot zvlášť.

Zásadní částí DQL algoritmu je volba odměny r v Algoritmu 1. Při návrhu funkce pro odměnu je v této práci bráno do úvahy několik aspektů:

1. V práci jsou využívána pouze simulovaná data. Tato data navíc vychází ze zcela fabrikovaného modelu a zda tento model odpovídá také reálným

datům z I3T je nutné v budoucnu ověřit.

2. Simulován je model úlohy, model studenta, vliv parametrů úlohy a studenta na výsledek řešení úlohy. Dále také učení studenta.

3. Vzhledem k simulovaným datům je obtížné vytvořit např. standardizovaný test, který by smysluplně ověřoval studentovy dovednosti. Simulované úlohy i studenti jsou pouze pouze číselné hodnoty pro abstraktní témata.

4. V simulovaném, ale i reálném prostředí jsou jednotlivé parametry vzájemně provázány. Např. špatné výsledky studentů na úloze lze interpretovat jako obtížnou úlohu, nebo slabé studenty. Rozhodnutí mezi těmito interpretacemi záleží např. na našem přesvědčení, zda vytváříme systém pro studenty, nebo se studenti mají snažit vyhovět požadavkům systému.

5. V reálném prostředí není znám počet témat, ani jejich hierarchie. Pokud je možné seznam témat a jejich hierarchii vytvořit, je to úloha pro interdisciplinární spolupráci s oblastmi psychologie, didaktiky, psychometrie, a dalších. Vytvoření Tématického modelu je tedy obtížné.

6. Není znám jednoznačný postup vytvoření dalších modelů, ať již studenta, úlohy nebo učení.

7. Při vytváření úloh se může stát, že úloha kromě tématu, na které je zaměřená, vyžaduje i další dovednosti, které si tvůrce úlohy neuvědomí.

8. Při využívání výukových online nástrojů jsou studenti obvykle velmi málo ochotní vyplňovat specializované testy k ověřování jejich dovedností.

9. Měření např. dovednosti studenta je, z důvodů uvedených výše, problematické. Dovednosti studenta jsou odhadovány skrze výsledky na řešených úlohách. Parametry těchto úloh však obvykle nejsou předem známy a jsou opět pouze odhadovány z výsledků studentů. Dovednosti v jednotlivých tématech navíc mohou být v jisté hierarchii a tak nemusí být nezávislé. Situaci navíc komplikují změna dovedností studenta v průběhu práce se systémem a šum v odpovědích způsobený externími okolnostmi působícími na studenta i nestabilitou studentských odpovědí obecně.

S přihlédnutím k okolnostem ze seznamu výše byly jako funkce odměny

použity pro srovnání dvě možnost r_1 a r_2:

$$r_1(a) \quad \begin{cases} \text{nejsou úspěšně splněny všechny úlohy} & : H(V(a)) \\ \text{při úspěšném splnění všech úloh} & : H(V(a)) + \frac{W}{T} \end{cases} \tag{4.1}$$

$$r_2(a) \quad \begin{cases} \text{nejsou úspěšně splněny všechny úlohy} & : \frac{H(V(a))}{t(a)} \\ \text{při úspěšném splnění všech úloh} & : \frac{H(V(a))}{t(a)} + \frac{W}{T} \end{cases} \tag{4.2}$$

$$V(a) \quad \begin{cases} C^+ & \text{správná odpověď při: } 0.8 \leq p(corr) \\ C^0 & \text{správná odpověď při: } 0.2 \leq p(corr) < 0.8 \\ C^- & \text{správná odpověď při: } p(corr) < 0.2 \\ I^+ & \text{chybná odpověď při: } 0.8 \leq p(corr) \\ I^0 & \text{chybná odpověď při: } 0.2 \leq p(corr) < 0.8 \\ I^- & \text{chybná odpověď při: } p(corr) < 0.2 \end{cases} \tag{4.3}$$

$$H(V(a)) \quad \begin{cases} 1 & \text{pro výsledek: } C^+ \\ 2 & \text{pro výsledek: } C^0 \\ 4 & \text{pro výsledek: } C^- \\ 0 & \text{pro výsledek: } I^+ \\ -1 & \text{pro výsledek: } I^0 \\ -4 & \text{pro výsledek: } I^- \end{cases} \tag{4.4}$$

$$W = 3 * \text{počet úloh v systému} \quad , \tag{4.5}$$

kde a : doporučená akce, obvykle studijní materiál/úloha

$r(a)$: získaná odměna po akci a

$p(corr)$: pravděpodobnost správné odpovědi, při simulaci je známa

$V(a)$: výsledek činnosti studenta na doporučené akci a, zde tedy kategorizace dle 2.2.5

$H(V(a))$: hodnota výsledku činnosti studenta na doporučené akci a

W : meta-parametr, váha ovlivňující důležitost celkového času

$t(a)$: čas strávený studentem na doporučené akci a

T : celkový čas strávený studentem na všech úlohách

Nastavený scénář rozděluje hodnotu odměny na dvě možnosti. První z nich je použita do té doby, dokud student nemá úspěšně splněny všechny úlohy v systému a zahrnuje pouze váhu jeho výsledku 4.4 na dané úloze. V případě vztahu (4.2) je do vtahu zahrnut i čas řešení úlohy. Čas řešení úlohy však může být zahrnut i přímo do váhy výsledku (4.4) a je tedy otázka, zda má taková úprava významný vliv. Čas řešení je často jedním z hlavních kriterií pro zařazení výsledku do jednotlivých kategorii.
Druhá možnost je použita při studentově úspěšném splnění všech úloh. Použije se tedy pouze jednou, při úspěšném vyřešení poslední nevyřešené úlohy. Při

této možnosti je, kromě výsledku na poslední úloze, do odměny zahrnut také celkový čas potřebný pro vyřešení všech úloh. Do tohoto času je zahrnuto i případné opakování úloh, které student nevyplnil úspěšně napoprvé.

Konkrétní hodnoty pro $H(V(a))$ a W jsou v této práci zvoleny autorem na základě níže popsané úvahy. Jejich vhodné nastavení může být námětem navazujících prací. Pro učení studenta je nejpřínosnější, pokud pracuje na úloze, která je na hranici jeho dovedností, např. ji zvládne za využití drobné nápovědy (hodnocení C^-), další v pořadí je úloha kterou student zvládne vypracovat na obvyklé úrovni (hodnocení C^0). Úloha, kterou zvládne student perfektně, může studenta motivovat, ale téma již pravděpodobně ovládal a z pohledu nových znalostí tedy měla jen malý přínos (hodnocení C^+). Na druhé straně, pokud student úlohu nesplní úspěšně, ale je na hranici jeho dovedností, pravděpodobně ji nesplní pouze těsně (hodnocení I^+). Horší je úloha, kterou student nezvládl např. z poloviny (hodnocení I^0). Hodnocení I^- je vyhrazeno pro velmi špatný výsledek. U nesplněných úloh (tedy s výsledkem I) se na odměně navíc negativně projeví nutnost tuto úlohu opakovat, čímž se prodlouží celkový čas řešení T.

Konstanta 3 ve výpočtu hodnoty W je nastavena intuitivně tak, aby mírně přesáhla hodnotu průměrného výsledku $V(a)$. Násobena počtem úloh v systému, tedy minimálním počtem akcí/úloh, které musí student splnit. Pokud student nesplní veškeré úlohy napoprvé, počet akcí/úloh, se tím přirozeně zvětší.

4.1.3 Reprezentace stavů a akcí

Termín **„akce“** je obvyklý v oblasti QL. V kontextu ITS v I3T znamená „akce“, předložení studijního materiálu/úlohy l. Student na tomto materiálu pracuje, což je reprezentováno výsledkem $V(a)$, respektive $V(l)$. Studentovi může být předložen stejný studijní materiál vícekrát. Ve scénáři zpracovaném v této práci je koncem práce s ITS stav, kdy má student úspěšně splněné veškeré dostupné úlohy $l \in L$.
Jednou z možností je omezit akce pouze na doporučení úlohy vybrané z těch úloh, které student úspěšně nesplnil, tedy $a(t) = l : l \in L_{\text{nesplněné}} \quad \forall t$, kde t značí počítadlo kroků, respektive počtů průběhu cyklu v Algoritmu 1 pro daného studenta. Pro úplnost je také potřeba po každém vyhodnocení akce upravit množinu nesplněných úloh, pokud student úlohu splní úspěšně.
$a(t) = l_i \implies V(a(t)) = V(l_i); \quad V(l_i) \in C^+, C^0, C^- \rightarrow L_{\text{nesplněné}} := L_{\text{nesplněné}} \setminus l_i$
Druhou možností je vždy vybírat úlohu k doporučení ze všech dostupných úloh, bez ohledu na jejich splnění. To zvyšuje množství možných akcí, přede-

vším v situaci, kdy má student již mnoho úloh splněných. Na druhé straně tato možnost umožňuje systému doporučovat i jednoduší úlohy např. na procvičení. Opakování jednodušších úloh může studentovi umožnit upevnit si některé znalosti, což může vést k lepším výsledkům na obtížnějších úlohách. Zvláštní kategorií jsou studijní materiály typu např. dokumentu k přečtení, videa či obrázku ke zhlédnutí, úlohy v tutoriálu s jednoznačným návodem k řešení a pod. U takového materiálu zřejmě nezle kategorizovat „úspěšnost vyřešení“, neboť na něm není nic, co by mohl student řešit. Obdobně je obtížně hodnotitelný čas strávený studentem prací na takovém materiálu. Jednak může být strávený čas, respektive vliv materiálu na studenta silně zašuměný. Např. každý student čte jinak rychle; video je sice puštěné, ale student ho nesleduje, nebo zcela nechápe jeho obsah a pod. Jednak může být delší čas strávený prací na podobném studijním materiálu pozitivním ukazatelem, neboť student se mu pravděpodobně věnuje důkladněji. Při úlohách, kdy má student za úkol vyřešit zadaný problém je naopak delší čas ukazatelem nižších dovedností studenta a tedy ukazatelem negativním.
V simulacích v rámci této práce jsou zahrnuty pouze úlohy, u kterých je krátký čas řešení hodnocen pozitivně. Zahrnutí studijních materiálů s jiným měřítkem času je ponecháno případným navazujícím pracím.

Stav je v QL nutné reprezentovat diskrétně. Množství stavů významně ovlivňuje množství potřebných trénovací dat. Čím více stavů systému (MDP) existuje, tím více je potřeba trénovacích dat. Extrémním přístupem by bylo reprezentování stavu jako kompletního seznamu všech výsledků studenta na všech pokusech všech úloh včetně pořadí pokusů. Bohužel, takových stavů by bylo neúnosné množství a navíc by jejich počet nebyl omezen. Za předpokladu, že je za konec učení považováno úspěšné splnění všech úloh, musí student úlohu opakovat. Maximální počet opakování každé úlohy není stanoven a tak by počet stavů mohl růst nade všechny meze. Vzhledem k okolnostem této práce 4.1.2 jsou vyzkoušeny následující možnosti.

1. Stav jako vektor posledních výsledků na každé úloze; úlohy jsou ve vektoru v pevném pořadí. Tedy $s = (v_1, v_2, \dots v_{|L|})$, kde L je množina všech úloh, v_i je výsledek posledního pokusu na úloze i a $v_i \in V \cup n$, kde V je množina možných výsledků (viz např. rovnice (4.4)) a n značí „neřešeno“. Počet těchto stavů je $|S| = (|V| + 1)^{|}L|$, neboť každou z $|L|$ úloh může student vyřešit s $|V| + 1$ výsledkem. V každém stavu je možné doporučit libovolnou úlohu, tedy $|L|$ akcí. Celkový počet Q-hodnot je tedy $|S \times L| = |L|(|V| + 1)^{|L|}$, což je hodnota, která s počtem úloh roste velmi rychle.

2. Stav jako počet jednotlivých typů výsledků; Pro každý možný výsledek $v \in V$ máme uložený počet úloh, které při student s tímto výsledkem řešil. Započítáváme vždy pouze poslední výsledek na každé úloze. Těchto stavů

je[7] $|S| = \frac{(|L|-1)!}{|V|!(|L|-|V|-1)!}$. Pokud lze v každém stavu doporučit libovolnou akci, je počet Q-hodnot $|S| * |L|$.

4.2 Sběr dat

Měření reálných dat probíhá v programu I3T pomocí tzv. *loggeru*. *Logger* je již součástí I3T a umožňuje zaznamenávat mnoho typů událostí, např. pohyb kurzoru myši či klik myši včetně polohy kurzoru při této události. Další události jsou např. zaměření prvku na Pracovní ploše I3T, jeho chycení, tažení či puštění. Samozřejmě je také možné zaznamenávat změnu hodnot v jednotlivých modulech. Kromě událostí, pro které je jejich zachytávání již připraveno, je také možné vytvářet a zachytávat vlastní události. Využití vlastní definice události bylo využito v rámci této práce pro zaznamenávání činnosti studenta na předložené úloze.

První definovanou událostí je spuštění úlohy studentem. Zaznamenán je identifikátor studenta, který aktuálně I3T používá, identifikátor spuštěné úlohy a časová značka, kdy byla úloha spuštěna. Druhou definovanou událostí je ukončení řešení úlohy. V této události jsou zaznamenány veškeré informace, které jsou zaznamenány v první události. Dále je v události ukončení úlohy zaznamenáno, zda byla úloha vyřešena úspěšně či nikoliv. V rámci této práce je předpokládáno zpracovávání úspěšnosti vyřešení úlohy a času, jak dlouho student úlohu řešil. Pro tyto dvě informace postačuje definice událostí tak, jak je popsaná výše. Který student řešil jakou úlohu je zaznamenáno identifikátory studenta a úlohy. Úspěšnost vyřešení je zaznamenána přímo v události ukončení úlohy. Konečně čas řešení úlohy je dán rozdílem časových značek v události ukončení a události spuštění úlohy.

Výstupem *loggeru* je textový soubor se všemi událostmi, které *logger* zachytil. Každá událost má svůj jedinečný identifikátor. Vyčlenění pouze událostí důležitých pro Inteligentní výukový systém je založeno na jednoduchém prohledání souboru, vyhledání událostí spuštění úlohy a událostí ukončení úlohy a jejich uložení do druhého textového souboru.

Z druhého, již filtrovaného, souboru jsou události načteny. A dále dle identifikátorů uživatele, úlohy a časových značek spárovány. Výstupem této části zpracování je seznam pokusů jednotlivých studentů na jednotlivých úlohách, včetně informace o době trvání řešení a úspěšnosti výsledku tohoto pokusu.

[7] Z binomické věty.

Záznam pokusu je již zpracován přímo Inteligentním tutor systémem. V této práci jsou konkrétně informace o správnosti řešení a době trvání řešení využity pro výpočet kategorizace výsledku daného pokusu, viz (2.2.5). Každé kategorii výsledku je v rámci zvolené metriky přiřazena hodnota, viz vztah (4.4). Tato hodnota je jíž přímo využita k výpočtu odměny, viz (4.1), pro daný pokus. Na závěr tato vypočtená hodnota vstupuje do výpočtu Q-hodnot, respektive jejich aktualizace, viz obecnou rovnici (3.5), nebo konkrétně v algoritmu DQL viz (10) a (13).

Soubor vytvořený *loggerem* zůstává nezměněný - data jsou z něj pouze čtena. Uložen je druhý soubor s filtrovanými událostmi a databáze jednotlivých pokusů jako „hrubá data“ na Obrázku 4.1. Dále jsou uloženy vypočtené výsledky pokusu, například kategorie, jako „databáze výsledků pokus“ ve zmíněném obrázku. Výsledky pokusu je výhodné mít uloženy zvlášť již spočítané, neboť na jejich základě je filtrován v reálném čase aktuální výsledek studenta. Je tedy vhodné mít tato data rychle k dispozici.

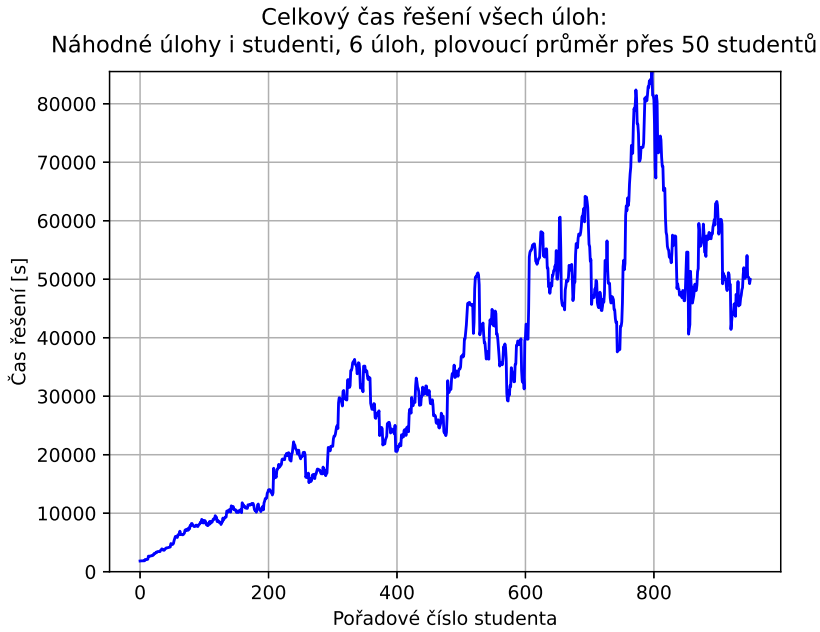
Kapitola 5

Výsledky na simulovaných datech

Výsledky při nastavení parametrů tak, jak je popsáno v kapitole 4.1.2, jsou velmi nepříznivé. Algoritmus nejen že není schopen zkrátit dobu řešení úloh, ale s přibývajícím počtem studentů, tedy trénovacích dat, délka času dokonce velmi výrazně roste. Tento trend je jasně patrný na Obrázcích 5.1 a 5.2. Z těchto obrázků je evidentní, že problém není v nefunkčnosti podstaty algoritmu. Algoritmus velmi úspěšně prodlužuje čas řešení a zvyšuje potřebný počet pokusů.

Na základě pozorovaných výsledků bylo usouzeno, že penalizace za prodloužený čas řešení v rovnicích (4.1) a (4.2) není schopna vyrovnat kladnou odměnu za splnění úlohy při správné odpovědi. Z toho důvodu bylo algoritmem vyhodnoceno, že nevýhodnější strategie je opakovaně řešit úlohy, které již student vyřešil. Do doby, dokud student nesplní všechny úlohy, procvičování stále probíhá a algoritmus tak těží z kladných odměn za správně vyřešené úlohy. Přitom se aktivně vyhýbá vyřešení všech úloh tak, aby mohlo procvičování dále probíhat. K ukončení procvičování dojde z důvodů ϵ-hladové strategie, díky které je, při dostatečném množství pokusů, zajištěno, že je vyzkoušena každá možná akce v každém daném stavu.

Pro správný běh algoritmu je nutné penalizovat každý pokus, ať již student vyřeší úlohu úspěšně, nebo neúspěšně. Tím je zaručeno, že bude v rámci algoritmu vyhledávána taková strategie, která minimalizuje počet pokusů a tím i celkového stráveného času. Pro další experimenty byla nastavena



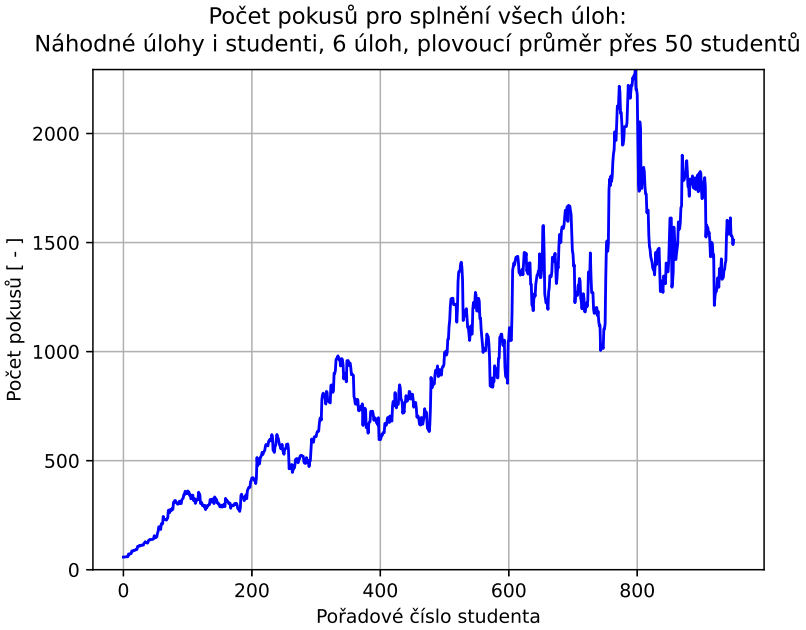
Obrázek 5.1: Čas potřebný pro splnění šesti úloh při nastavení dle 4.1.2. Na Obrázků můžeme vidět, že algoritmus pracuje obráceně, než bylo očekáváno. Algoritmem byla nalezena strategie, jak úspěšně prodloužit dobu, kterou studentovi trvá splnění všech úloh.

hodnota výsledků následovně:

$$H(V(a)) \quad \begin{cases} -2 & \text{pro výsledek: } C^+ \\ -1 & \text{pro výsledek: } C^0 \\ 0 & \text{pro výsledek: } C^- \\ -1 & \text{pro výsledek: } I^+ \\ -2 & \text{pro výsledek: } I^0 \\ -4 & \text{pro výsledek: } I^- \end{cases}, \tag{5.1}$$

(5.2)

přičemž nastavení dalších parametrů popsaných v kapitole 4.1.2, zůstalo nezměněno a byl použit vztah (4.1) pro výpočet odměny. Tyto hodnoty byly zvoleny v rámci následující úvahy: Při učení studenta je vhodné, aby úloha byla na hranici dovedností studenta. Při také úloze dochází k největšímu posunu v dovednostech studenta, viz také 2.6. Takovou úlohu student pravděpodobně vyplní s mírnými obtížemi - hodnoceno kategorií C^-, případně úlohu těsně nevyplní - hodnoceno kategorií I^+. Čím více jistě student úlohu vyplní, nebo naopak nevyplní, tím méně přínosná pro něj úloha je. Z toho důvodů roste penalizace směrem k jednoznačnějším kategoriím C^+ a I^-. Úspěšné vyřešení úlohy je preferováno za prvé z důvodu nutnosti úlohy vyřešit, pro splnění stanoveného cíle. Za druhé proto, že úspěšné řešení úloh vede k větší motivaci studenta. Připomeňme zde cíl, při kterém je procvičování studenta považováno za úspěšně dokončené: *Cílem ITS je zkrátit na minimum celkový čas, potřebný pro úspěšné splnění všech úloh v systémů.* Reprezentace stavů byla pouze

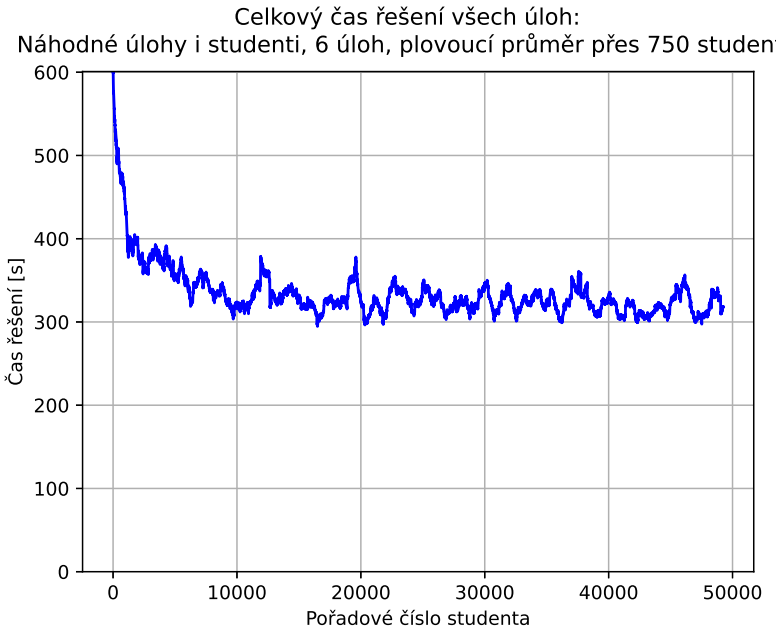


Obrázek 5.2: Počet pokusů pro splnění šesti úloh při nastavení dle 4.1.2. Podobně jako na Obrázku 5.1 vidíme, že s větším množstvím trénovacích dat, tedy studentů, se zvyšuje množství pokusů, které student vykoná pro splnění všech úloh.

mírně upravena, oproti původně navrhované. Stav je reprezentován seznamem *nejlepších* výsledků studenta pro každou úlohu. Pro každou úlohu je tedy v paměti držena jedna hodnota výsledku. Konkrétně nejlepšího výsledku, kterého student na dané úloze dosáhl za celou dobu procvičování.

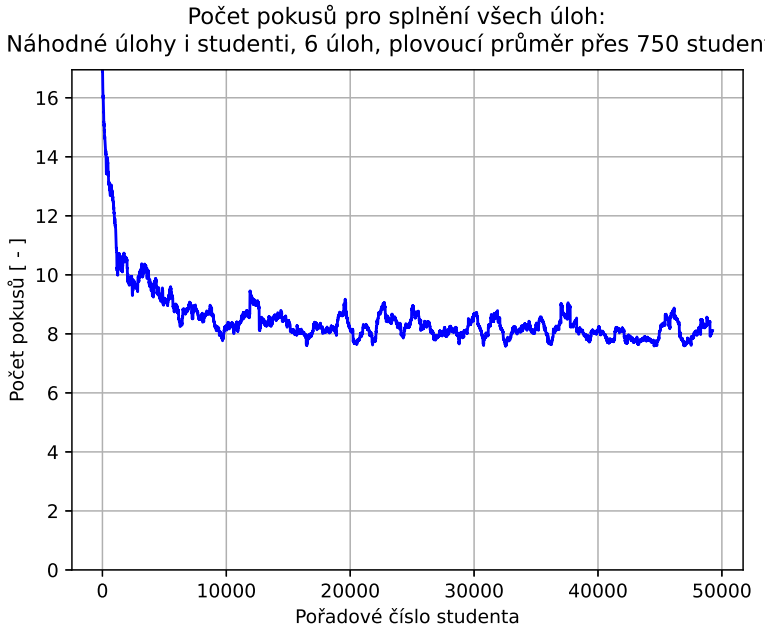
Pří výše zmíněném nastavení již výsledky vycházejí příznivě. Na Obrázku 5.3 můžeme vidět, že při šesti nabízených úlohách, dokáže systém výrazně zkrátit celkový čas řešení pro studenty s pořadovým číslem kolem 1000, od pořadového čísla studentů přibližně 10000 se již hodnota viditelně nesnižuje. Téměř identickou křivku můžeme vidět na druhém Obrázku 5.4, který zobrazuje počet pokusů, které studenti potřebují pro splnění všech úloh. Na první pohled paradoxní výsledek zobrazuje Obrázek 5.5 na kterém vidíme přírůstek dovednosti, který studenti zaznamenali za celou dobu procvičování. Čím rychleji, a tedy i na méně pokusů, studenti splnili všechny úlohy, tím menší přírůstek hodnoty dovednosti zaznamenali. Vysvětlením může být, že hodnota přírůstku přímo úměrně souvisí s počtem pokusů. Tím, že byl činností systému snížen počet pokusů studenta, byl snížen také počet příležitostí zlepšit studentovi dovednosti.

Na Obrázcích 5.6 a 5.7 můžeme vidět, že i při zvýšení počtu úloh v systému na 20, dokázal systém výrazně (o 25%) snížit počet potřebných pokusů pro splnění všech úloh. K tomuto snížení došlo již relativně brzy, přibližně u

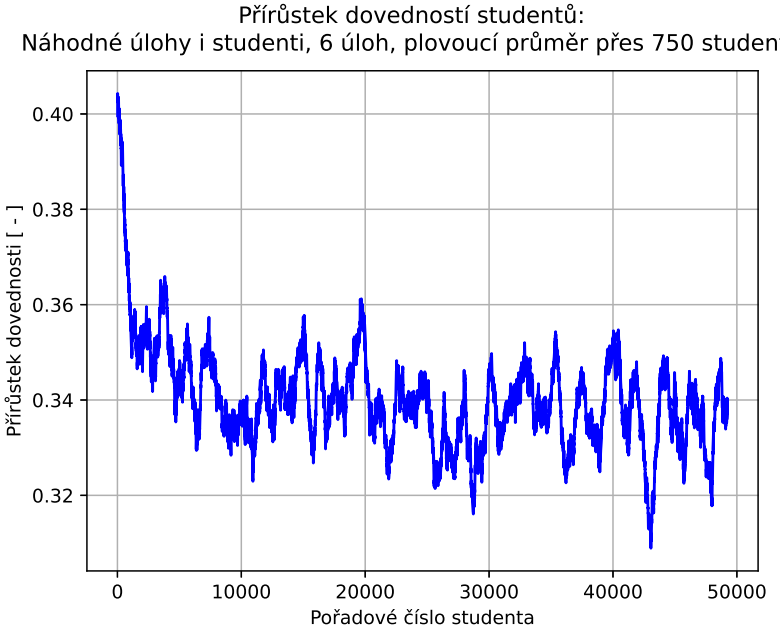


Obrázek 5.3: Čas potřebný pro splnění šesti úloh. Na obrázku můžeme vidět, že potřebná doba rychle klesá s první přibližně tisícovou studentů. Dále klesá mírněji k pořadovému číslu studenta deset tisíc a dále je již zůstává na hodnotě mírně přes 300 vteřin.

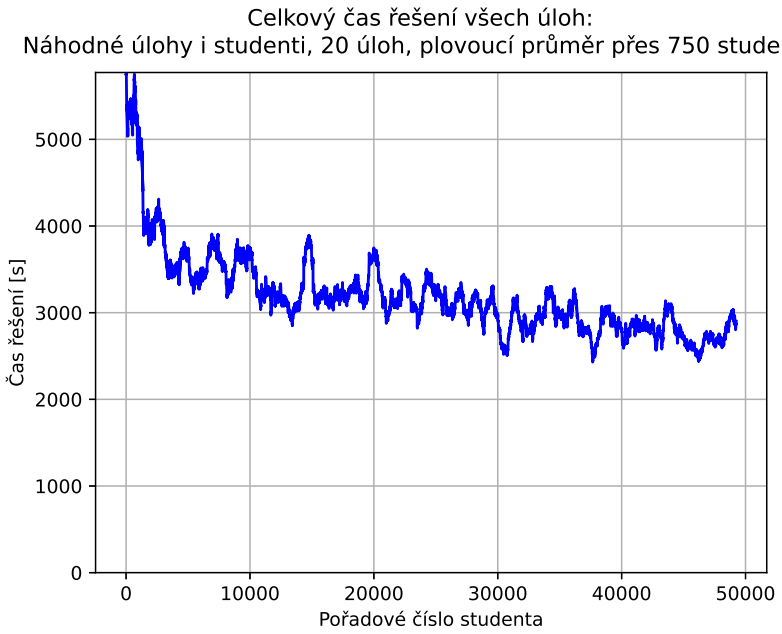
studenta s pořadovým číslem 2000. U studentů s pořadovým číslem přibližně 10000 a více je počet potřebných pokusů snížen až na téměř 50%. Na první pohled mohou tyto počty studentů vypadat velmi vysoké, vzhledem k běžnému počtu studentů např. ve výuce na vysoké škole. Na druhou stranu je možné, že v budoucnu bude nástroj I3T dostupný i on-line, např. skrze webové rozhraní a v takovém případě jsou již nízké desítky tisíc uživatelů realistickým počtem.



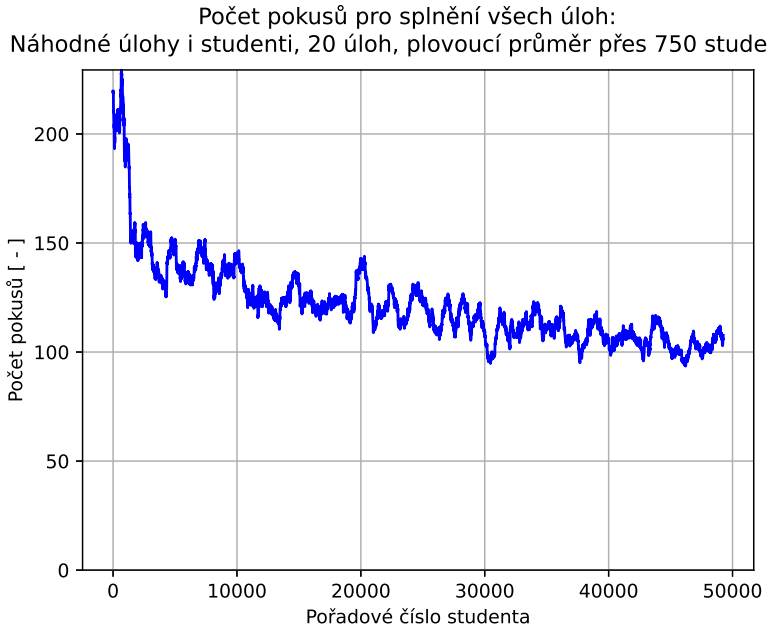
Obrázek 5.4: Počet pokusů pro splnění šesti úloh. Na obrázku můžeme vidět, že od pořadového čísla studenta deset tisíc je množství počtu pokusů přibližně poloviční oproti prvním studentům.



Obrázek 5.5: Přírůstek dovednosti při splnění šesti úloh. Na obrázku můžeme vidět paradoxní situaci, kdy studenti, kteří zvládnou splnit úlohy rychleji, respektive na méně pokusů, zaznamenávají nižší přírůstek dovednosti. Rychlost studentů, respektive počet jejich pokusů můžeme nalézt na Obrázcích 5.3, respektive 5.4



Obrázek 5.6: Čas potřebný pro splnění dvaceti úloh. Na obrázku můžeme vidět, že i při vyšším počtu úloh - zde dvaceti, je systém schopen doporučit vhodné pořadí úloh již pro studenty s pořadovým číslem přibližně dva tisíce a více.



Obrázek 5.7: Počet pokusů pro splnění dvaceti úloh. Oproti Obrázku 5.4, pro situaci se šesti úlohami v systému, můžeme vidět, že při vyšším počtu úloh - zde dvaceti úlohách, klesá potřebný počet pokusů pozvolněji. Již od prvních přibližně dvou tisíců studentů dále, vidíme pokles počtu pokusů o přibližně 25%.

Část III

Zpracování dat

Kapitola 6

Evaluace dat

Při zpracování dat narazili autoři práce [Pap15] na několik problémů. Jejich neúplný seznam a doporučení autorů shrnují odstavce níže.

Získaná data jsou obvykle zkreslena tím, že úlohy vybrané systémem nejsou studentům předkládány náhodně, ale jsou ovlivněny samotným systémem. Nebo je chybovost studentů nezávislá na jejich dovednosti. Takové zkreslení je zásadní především pro systémy, které z nasbíraných dat odhadují parametry pevně stanoveného modelu. Obvykle také systém, respektive jeho tvůrci, nemají možnost ovlivňovat chování studentů a přimět je k práci na úlohách tak, „jak by se to hodilo“. Data jsou velmi často poznamenána výběrovým zkreslením (*attrition bias*), neboť opuštění práce se systémem není náhodné. Většina studentů používá systém pouze krátkou dobu. V rámci systému také obvykle existuje velké množství úloh a tedy velké množství možností, v jakém pořadí a s jakým výsledkem je studenti řeší. Při použití doporučovacího systému není pořadí úloh náhodné, což také ztěžuje porovnání jednotlivých výsledků.

Běžným způsobem pro vyhodnocení účinnosti systému by bylo otestování studentů před a po využití systému. Využití testů je v on-line doporučovacích systémech problematické, protože uživatelé obvykle nemají motivaci účastnit se testování před využitím systému a už vůbec ne po jeho, mnohdy náhlém, opuštění. Autoři zdůrazňují, že v rámci on-line systémů je relativně snadné získat velké množství dat, která jsou ale silně zatížena různými zkresleními a je tedy nutné je zpracovávat a interpretovat obezřetně. Zároveň je dle autorů měření účinnosti použití on-line systémů nevyhnutelné, neboť je nutné kontrolovat, zda činnost systému naplňuje očekávání pro jeho zavedení.

Na jedné straně je smyslem zlepšení dovedností studentů, na druhé straně podobné systémy ve velké míře používají uživatelé dobrovolně a je tedy potřeba nezapomenout i na jejich „přitažlivost“ tak, aby je uživatelé používali a vraceli se k nim. Dle autorů [Pap15] jsou současné práce zaměřené hlavně na evaluaci modelu studenta, tady jak přesně popisuje a předpovídá studentovy výsledky.

Volba metriky, kterou porovnáváme jednotlivé systémy, či modely je zcela zásadní. Výsledky porovnání různých modelů, týkající se přesnosti předpovědi (např. pravděpodobnosti správné odpovědi, času řešení) vykazují velmi malé rozdíly a vedla autory studií k zamyšlení nad dalšími možnostmi porovnání různých modelů. Autoři [Pap15] ukázali, že malé rozdíly jsou způsobeny tím, že úlohy řešené uživatelem jsou (samozřejmě) ty, které mu systém sám doporučí. Toto zkreslení vede k velmi rozdílným scénářům napříč uživateli a k relativně dobrým výsledkům i u jednoduchých systémů odhadujících výkon studenta. Výsledkem takových úvah jsou návrhy na zaměření se např. na zahrnutí přínosu použití systému pro uživatele. Takový přínos může být měřen například celkovým časem potřebným k dosažení určité úrovně dovedností. Čím rychleji uživatel zlepší své dovednosti, tím je systém lepší. Měření přínosu pro uživatele je nicméně dle autorů [Pap15] obtížné z důvodů popsaných výše. Mnoho systémů se tak zaměřuje např. na sledování celkového času stráveného v systému, což lze interpretovat jako míru zaujetí (*engagement*). Bohužel, míra zaujetí nemusí souviset se zlepšením dovedností studenta [LPFK13].

Možnost představená také autory [Pap15] je využití simulovaných dat. Autoři zdůrazňují, že tato data by měla být simulována na základě reálných naměřených dat. Výhodou simulovaných dat, je absolutní znalost všech hodnot parametrů, které zůstávají v reálných datech skryté, obtížně měřitelné, nebo zatížené různými typy zkreslení.

Kapitola 7

Simulace dat

Jak je uvedeno v úvodu práce, z důvodu nedostatku reálných dat jsou v práci zpracovávána simulovaná data. Rozdíl mezi výsledky na simulovaných datech a skutečných studentech může být způsoben několika aspekty (seznam je neúplný):

- model neodpovídá skutečnému vztahu mezi úlohami a chováním studentů z důvodu chybně nastavených parametrů
- model nezahrnuje některou z podstatných součástí vztahu mezi úlohami a studenty (například je důležité také zahrnout únavu studenta či zapomínání)
- v rámci modelu je nevhodně simulováno učení studentů

Při tvorbě zadání práce byl dále předpoklad možnosti vytvořit pořadí úloh, například ve formě tutoriálu, na základě intuitivního přístupu experta (učitele) dané oblasti. Výsledky studentů by poté mohly být porovnávány při dodržení stanoveného pořadí úloh, oproti výsledkům dosaženým při využití ITS. Simulované úlohy jsou reprezentovány pouze hodnotami svých parametrů a není tedy jednoznačné, jak úlohy seřadit. To je další z důvodů, který brání splnění cíle práce, zaměřeného na porovnání výsledků různých skupin studentů.

Výsledkem je zhodnocení vlivu doplňku vytvořeného v této práci na model studenta, který je v této práci použit. A dále zjištění potřebného počtu studentů a jejich odpovědí pro použití doplňku.

7.1 Model úlohy, studenta a pokusu

Pro vytvoření umělých dat je v této práci použit model, který pokrývá kombinaci parametrů úlohy, studenta a konkrétního pokusu studenta o zvládnutí úlohy. Model je podobný modelu, který navrhl [VDL09b]. Zásadní rozdíly jsou dva. Níže popsaný model umožňuje zachytit situaci, kdy student již téma zcela ovládá. V jiných modelech je obvyklé, že dovednost studenta může růst do nekonečna a tedy se může student vždy zlepšovat. Druhým rozdílem je zahrnutí minimálního času potřebného pro řešení úlohy. V jiných modelech lze vždy kompenzovat kratší strávený čas pomocí vyšší dovednosti. V níže popsaném modelu i student s perfektní znalostí, snižuje pravděpodobnost na úspěšné vyřešení úlohy, pokud věnuje úloze kratší čas.

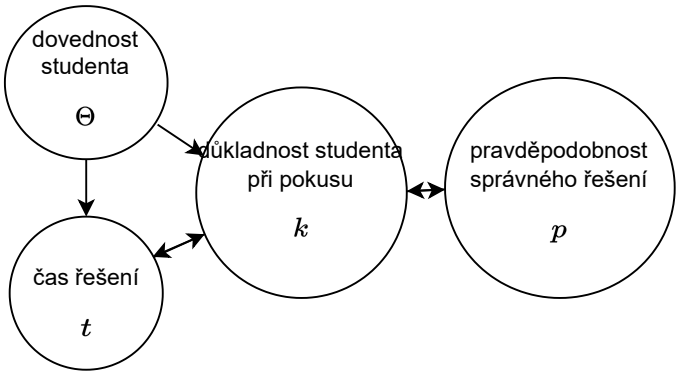
Model je založen na logistické funkci v souladu s běžnou praxí v rámci IRT [VDL09b], [Řih17]. Tato funkce vyjadřuje měnící se pravděpodobnost úspěšného řešení úlohy. Obvykle je nezávislou proměnnou logistické funkce dovednost studenta, čas řešení, nebo součin dovednosti a času, obvykle ve formě součtu logaritmů těchto hodnot [VDL09b]. V modelu v této práci je kombinována dovednost a čas odlišným způsobem viz kapitola 7.3, který umožňuje stanovení minimálního času, po který je nutné úlohu řešit. Tento minimální čas nelze kompenzovat lepší znalostí tématu. Lze ho brát jako čas potřebný k technickému provedení odpovědi, např. napsání slovní odpovědi, zapojení blokového schématu, výběr odpovědi z nabídky a pod. Minimální potřebný čas se v rámci I3T výrazně projevuje[1], neboť i relativně jednoduchá úloha vyžaduje obvykle vytvoření několika objektů na obrazovce, jejich propojení a nastavení správných hodnot. Tyto činnosti trvají minimální čas i u studenta, který naprosto jistě ví, jak úlohu vyřešit.

Použití pro vytvoření umělých dat je přímočaré. Nejprve je třeba vygenerovat simulovaného studenta, tedy určit hodnoty parametrů náležejících studentovi (např. dovednost). Stejným způsobem vytvořit simulované úlohy (například diskriminační faktor, minimální čas). A na závěr zvolit parametry konkrétního pokusu, kde jeden student řeší jednu úlohu (např. jak dlouho student úlohu řešil). Z výše zmíněných hodnot je na základě modelu vypočtena pravděpodobnost úspěšného řešení. S danou pravděpodobností je poté řešení označeno jako úspěšné či nikoliv a jsou zaznamenány všechny parametry pokusu.

[1] Podle názoru autora - tuto domněnku je třeba ověřit.

7.1.1 Proměnné a parametry modelu

Vzájemný vztah proměnných modelu zobrazuje schematicky Obrázek 7.1.



Obrázek 7.1: Náčrtek vztahu důkladnosti k, času řešení t, dovednosti studenta Θ a pravděpodobnosti úspěšného řešení p v modelu. **Dovednost studenta** Θ ovlivňuje, při dané dané důkladnosti k čas řešení t - čím vyšší dovednost, tím kratší čas řešení. **Čas řešení** t ovlivňuje důkladnost k při dané dovednosti Θ - čím déle se student řešení věnuje, tím roste důkladnost k. **Důkladnost** k ovlivňuje pravděpodobnost úspěšného vyřešení úlohy p - čím důkladněji student úlohu řeší, tím vyšší je pravděpodobnost na úspěch.

Na vztahy mezi proměnnými modelu, viz Obrázek 7.1 lze pohlížet i z druhé strany, než je vysvětleno v popisku pod obrázkem. Např. pokud je stanovený čas řešení t, pak dovednost Θ ovlivňuje důkladnost k - čím vyšší dovednost, tím vyšší důkladnost. Dále například, pokud požadujeme, aby student úlohu splnil úspěšně s $p = 0.7$ (viz část 2.6), spočítáme odpovídající hodnotu důkladnosti k. Z této hodnoty důkladnosti můžeme vypočítat čas t, který bude potřebovat student s danou či odhadnutou dovedností Θ, pro vyřešení úlohy. Tím jsme odhadli čas řešení úlohy pro daného studenta, při požadované pravděpodobnosti úspěchu.

Proměnné jsou v modelu použitém v této práci čtyři:

- Čas řešení úlohy $t \in \langle 0, \infty)$, případně jeho logaritmus $\tau = \ln(t)$. Jeho vztah k ostatním proměnným vyjadřuje rovnice (7.1) a graficky zobrazuje Obrázek 7.2.
- Dovednost studenta $\Theta \in (0, 1\rangle$, případně více dovedností $\overrightarrow{\Theta}$, kde hodnota 0 (v limitě[2]) znamená, že student není tuto dovednost schopen vykoná-

[2]Hodnota 0 není zahrnuta ze dvou důvodů. V rovnicích (7.1) a (7.3) vede k nejednoznačným hodnotám pro t a k, dále při zvyšování dovednosti studenta (7.8), je počítáno se závislostí na dovednosti a při hodnotě 0 by nikdy nedošlo ke zlepšení. Vedlejším důvodem je

vat/použít a hodnota 1 znamená, že student dovednost ovládá perfektně. Vztah k ostatním proměnným vyjadřuje rovnice (7.5).

- Důkladnost (*thorough*) $k \in \langle 0, \infty)$ označuje množství „energie“, kterou student řešení úlohy věnoval. Pro intuitivní představu (pro detaily viz kapitola 7.1.2): V situaci, kdy $k = 1$, bude studentovi s $\Theta = 1$ trvat vyřešení úlohy přesně minimální čas, za který lze úlohu vyřešit. I student s perfektní dovedností však může úloze věnovat více času (potom bude $k > 1$). Vztah k ostatním proměnným vyjadřuje rovnice (7.3) a graficky zobrazuje Obrázek 7.3.
- Pravděpodobnost úspěchu $p \in \langle 0, 1\rangle$ určuje, jaká je šance, že student úlohu vyřeší správně. Vztah k ostatním proměnným vyjadřuje rovnice (7.6), případně normalizovaná verze v rovnici (7.7) a graficky zobrazuje Obrázek 7.4.

Mezi výše uvedenými proměnnými není pevně stanoveno, která z nich je závislá a která nezávislá. Vztah proměnných záleží na konkrétní situaci v ITS. V situaci, kdy jsou známy nebo odhadnuty hodnoty Θ a k a dle nastavení ITS je požadavek na předložení úlohy s danou hodnotou p, bude závislou proměnnou t. V jiném případě, kdy z provedeného pokusu o řešení úlohy je známa hodnota t, dále jsou již odhadnuty k a p, je závislou proměnnou Θ. A tak dále.

Počet parametrů modelu je čtyři až pět. Částečně stranou pak stojí parametry pro rychlost učení studenta popsané v části 7.1.3, které nemají na řešení aktuální úlohy vliv. Parametry modelu zahrnují:

- Minimální čas úlohy $min_t \in \langle 0, \infty)$ určuje minimální čas potřebný pro vyřešení úlohy. Tedy jak dlouho potrvá úlohu vyřešit (bez hádání a s požadovanou jistotou) studentovi s perfektní znalostí. Minimální čas může zahrnovat např. dobu potřebnou pro zapsání odpovědi, provedení potřebných kroků v úloze a pod. Tato hodnota se také může měnit s typem zařízení na kterém student úlohu vypracovává (např. napsat slovní odpověď může na chytrém telefonu trvat déle než na stolním počítači bez ohledu na dovednost studenta).
- Pravděpodobnost uhodnutí $c \in \langle 0, 1\rangle$ určuje šanci na uhodnutí správného řešení úlohy bez znalosti tématu, nebo s velmi malým časem řešení. Tato hodnota je důležitá hlavně v úlohách s výběrem možných odpovědí (*multiple choice*).

také význam „absolutní neschopnosti“, který nedává v praktickém použití smysl - student je vždy schopen, alespoň náhodně, zkoušet s programem pracovat. V opačném případě by jakékoliv měření postrádalo smysl.

- Diskriminační faktor dovednosti $A \in \langle 0, \infty)$ moduluje vztah dovednosti Θ a času řešení t a jejich vzájemného vlivu na řešení úlohy. Viz Obrázek 7.2 a 7.3 vpravo dole.

- Diskriminační faktor pravděpodobnosti $a_p \in \langle 0, \infty)$ udává, jak ostře jsou v řešení úlohy rozděleni studenti, kteří řeší úlohu s vysokou důkladností k oproti těm, kteří řeší úlohu s nízkou důkladností.

- Koeficient obtížnosti $b_p \in (-\infty, \infty)$ udává obtížnost úlohy.
  Pokud přidáme požadavek na normalizaci: pokud řeší úlohu perfektní student $\Theta = 1$ přesně po dobu potřebného času $t = min_t$, bez hádání $c = 0$, bude jeho pravděpodobnost úspěchu $p \stackrel{!}{=} norm_p$, plyne z tohoto požadavku, že hodnota b_p je svázána s hodnotou a_p a tedy lze tento parametr vypustit.

7.1.2 Rovnice modelu

Rovnice (7.1), (7.2), (7.3), (7.4) a (7.5) jsou různými zápisy vztahu mezi t, k a Θ. Každý z těchto vztahů, respektive vyjádření různé proměnné, je použit v různých situacích v rámci ITS.

$$t = exp(\Theta^{-A} \ln(k * min_t)) = (k * min_t)^{\Theta^{-A}} \tag{7.1}$$

$$\tau = \ln(t) = \Theta^{-A} \ln(k * min_t) \tag{7.2}$$

$$k = \frac{t^{\Theta^A}}{min_t} \tag{7.3}$$

$$\ln k = \tau \Theta^A - \ln min_t \tag{7.4}$$

$$\Theta = \sqrt[A]{\frac{\ln k * min_t}{\tau}} \quad , \tag{7.5}$$

kde t, τ : doba, respektive logaritmus doby, řešení úlohy
k : důkladnost
Θ : dovednost studenta
min_t : minimální doba pro řešení úlohy
A : diskriminační faktor dovednosti

Pro danou (vypočtenou) hodnotu k, resp. $\ln k$, je poté pravděpodobnost správného řešení p dána vztahem:

$$p(correct|\dots) = c + (1-c)\frac{e^{a_p(\ln k - b_p)}}{1 + e^{a_p(\ln k - b_p)}} \overset{d=e^{b_p}}{=} c + (1-c)\frac{k^{a_p}}{d^{a_p} + k^{a_p}} \quad , \tag{7.6}$$

kde c : pravděpodobnost uhodnutí
a_p : diskriminační faktor pravděpodobnosti
b_p : koeficient obtížnosti úlohy
k : důkladnost
d : $= e^{b_p}$

Na vztah v rovnici 7.6 můžeme položit normalizační požadavek. Perfektní student, tedy takový s $\Theta = 1$, který řeší úlohu právě po minimální potřebný čas $t = min_t$, s důkladností $k = 1$ (při daném $\Theta = 1$ jsou podmínky na $t = min_t$ a $k = 1$ ekvivalentní) a výsledek nebude ovlivňovat možnost hádat $c = 0$, vyřeší úlohu úspěšně s pravděpodobností $p = norm_p$. Hodnota $norm_p$ je meta-parametr (např. $norm_p = 0.95$). Při použití této normalizace má člen d^{a_p} v rovnici (7.6) vždy hodnotu $\frac{1-norm_p}{norm_p}$ a tedy je ve zmíněné rovnici jeden z parametrů a_p nebo b_p, dle naší volby, vyjádřen pomocí meta-parametru $norm_p$ a zbývajícího parametru. Rovnici tedy můžeme přepsat např. následovně:

$$p(correct|\dots) = c + (1-c)\frac{k^{a_p}}{\frac{1-norm_p}{norm_p} + k^{a_p}} \quad , \tag{7.7}$$

kde c : pravděpodobnost uhodnutí
a_p : diskriminační faktor pravděpodobnosti
k : důkladnost
$norm_p$: meta-parametr, pravděpodobnost úspěšného řešení při $\Theta = 1$, $k = 1$, $t = min_t$, $c = 0$

Graficky zobrazují vztah mezi t, k a Θ obrázky 7.2, na kterém je závislá proměnná čas t a 7.3, na kterém je závislá proměnná důkladnost k. Průběh pravděpodobnosti v závislosti na důkladnosti k, respektive kombinaci času t a dovednosti Θ, a také normalizovaný průběh pravděpodobnosti zobrazuje obrázek 7.4.

7.1.3 Model učení

Zachycení učení, tedy změně dovednosti studenta, bylo modelováno např. pomocí tzv. funkce aktivace paměti 2.7.2, která započítává vliv každého pokusu o řešení úlohy a také, v kombinaci se zapomínáním, časový odstup od každého pokusu.

Modelování učení je obecně komplikovaný úkol, neboť na změnu dovednosti studenta má vliv celá řada okolností, které nejsou v rámci ITS měřitelné. Příkladem může být studium a aktivity studenta v době mezi dvěma přihlášeními do ITS; postupné zapomínání, které opět úzce souvisí s činností studenta v rámci ITS ale i mimo něj. Na učení má také vliv kontext a další dovednosti a znalosti studenta.

V této práci je učení modelováno zjednodušeně, jako okamžitá změna dovednosti studenta po vystavení úloze. Tato změna závisí pouze na aktuální úloze, úspěchu s jakým ji student vyřešil, obtížnosti úlohy a na aktuální hodnotě dovednosti studenta. Model předpokládá možnost existence více témat (*knowledge component*), které úloha obsahuje. Přírůstek v dovednosti pro každé téma je v takovém případě nezávislý na dalších tématech. Součástí modelu jsou koeficienty, upravující vliv konkrétní úlohy, konkrétního studenta a konkrétního tématu, který je v dané úloze obsažen. Model nijak nepracuje se zapomínáním, tedy postupným snižováním dovedností a znalostí v průběhu času. Zapomínání není do modelu zahrnuto, protože hlavní předpokládané využití programu I3T a tedy i doplňku vyvíjeného v rámci této práce, je využití jako podpůrný systém při výuce počítačové grafiky. V průběhu semestru má tak dle názoru autora zapomínání zanedbatelný vliv vzhledem k množství nových informací, u kterých má výraznější vliv jejich neznalost studentem. Pro zachycení změny dovednosti studenta je v rámci práce použit vztah (7.8), zobrazen na Obrázku 7.5. Použití tohoto vztahu je zavedeno na základě několika požadavků modelu a předpokladů[4]:

1. Maximální hodnota dovednosti je $\Theta = 1$.
2. Existuje hodnota dovednosti Θ různá od 0, pro kterou dochází k nejrychlejšímu učení, tedy největšímu přírůstku dovednosti. Předpoklad vychází z domněnky, že pokud student ví o tématu velmi málo, jeho učení je pomalejší, než učení studenta, který již v tématu dosáhl jisté úrovně.
3. Úspěšné řešení obtížnější úlohy, přináší vyšší přírůstek dovednosti než úspěšné řešení jednodušší úlohy.

[4]Tyto předpoklady je nutné ověřit na naměřených datech, což nebylo v práci provedeno z důvodu nedostatku (absence) reálných dat.

4. Na přírůstek dovednosti má vliv pouze aktuální dovednost studenta a obtížnost úlohy. Tedy nikoliv např. čas strávený řešením úlohy a pod.

$$\Theta_{new} = \Theta + (1 - \Theta) l_{KC} \frac{\Theta^{l_\Theta} e^{l_b b_p}}{1 + \Theta^{l_\Theta} e^{l_b b_p}} \quad , \tag{7.8}$$

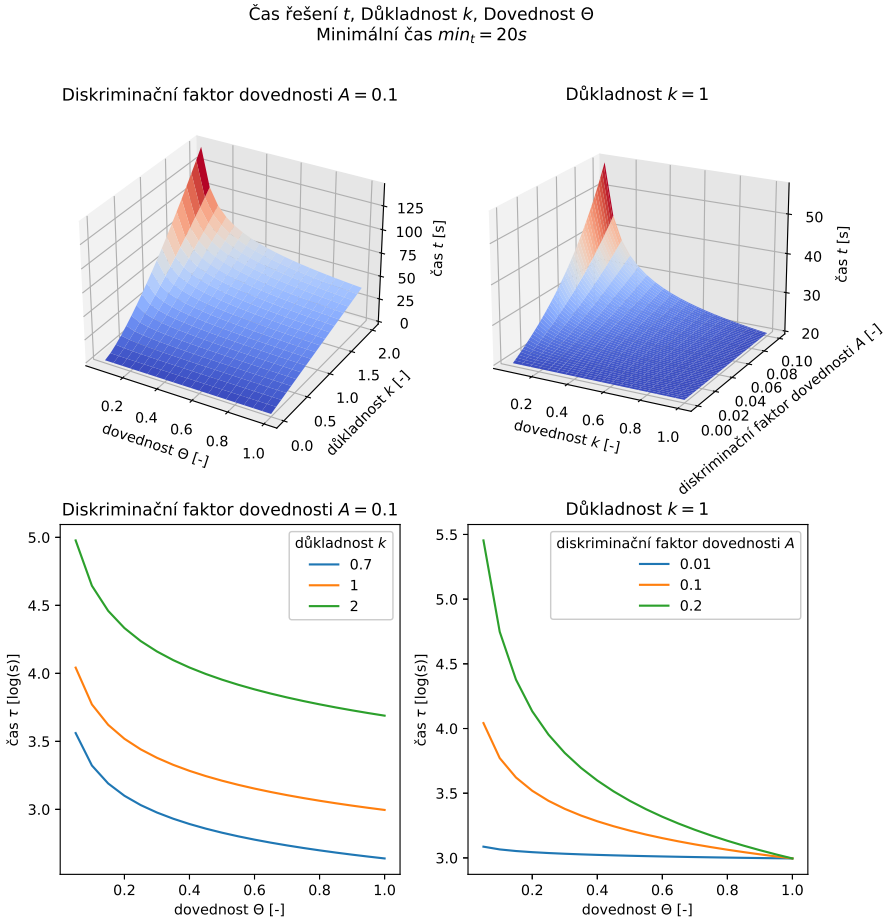
kde Θ_{new}, Θ : jsou dovednosti studenta po změně a před ní

l_{KC} : koeficient učení pro danou oblast (*knowledge component*) $\in \langle 0, 1 \rangle$

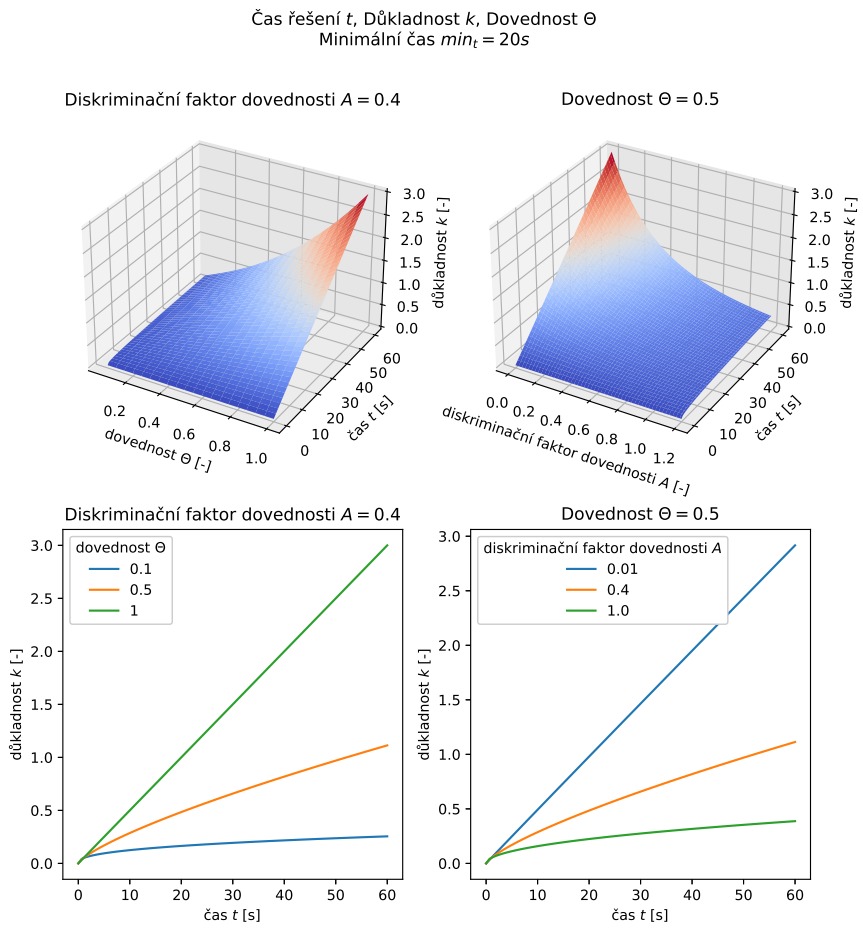
l_Θ : koeficient učení studenta

l_b : koeficient učení úlohy

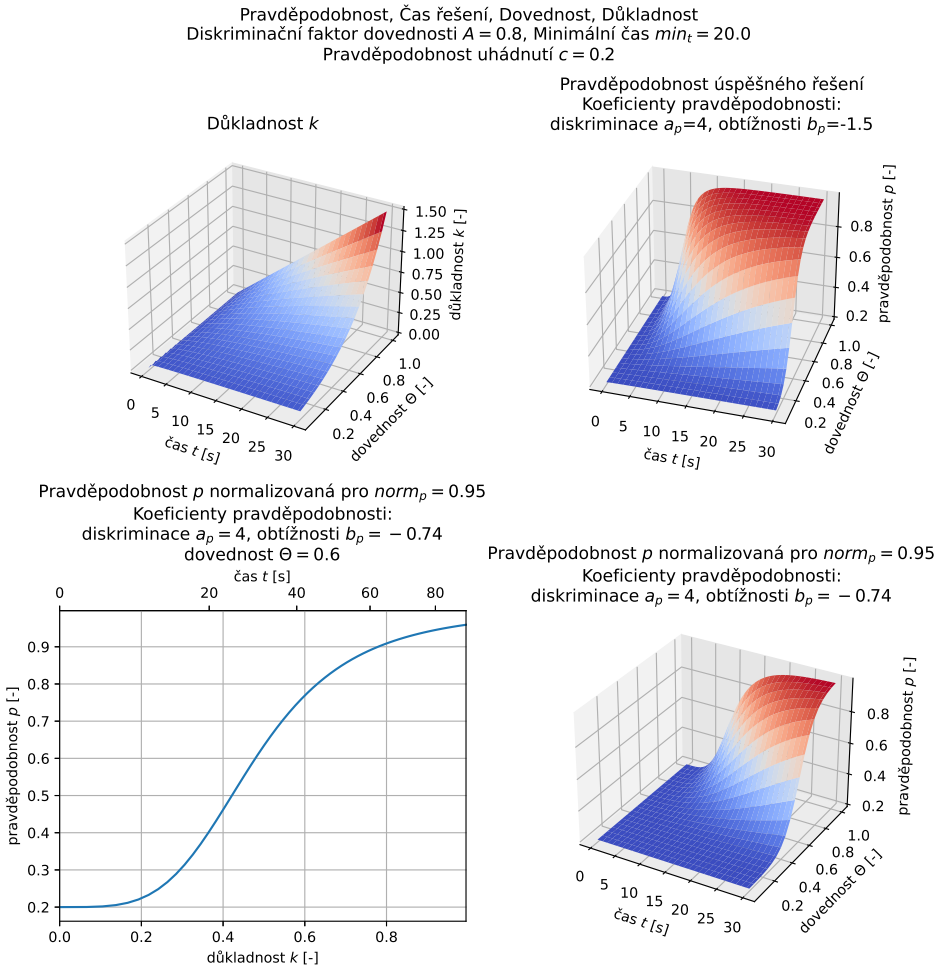
b_p : obtížnost úlohy (viz také 7.6)



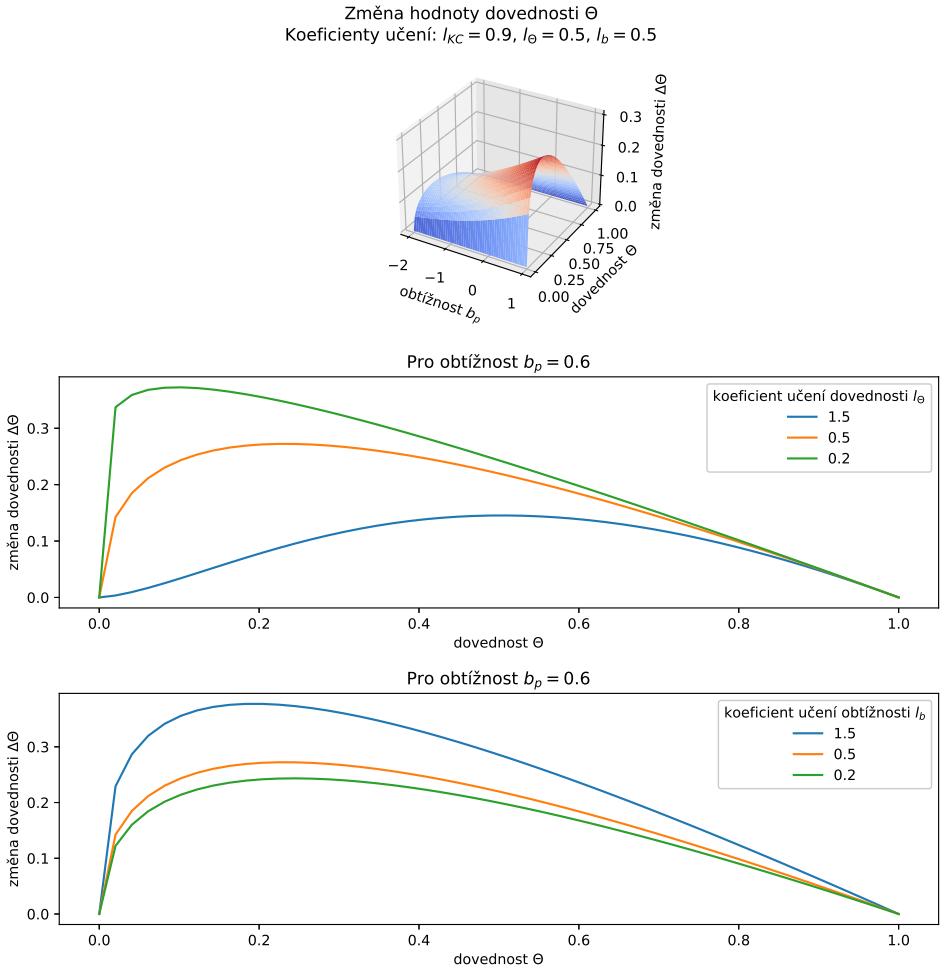
Obrázek 7.2: Závislost času t na důkladnosti k a dovednosti Θ. Na obrázku[3] **vlevo nahoře** můžeme vidět, jak s rostoucí důkladností roste čas řešení úlohy. Na rychlost nárůstu času vzhledem k důkladnosti má vliv dovednost studenta, kdy pro nízkou hodnotu dovednosti roste čas rychleji. Pro $\Theta = 1$ a $k = 1$ je $t = min_t$. Na obrázku **vpravo nahoře** můžeme vidět vliv Diskriminačního faktoru dovednosti A. Pokud úloha studenty dle jejich dovednosti nediskriminuje ($A = 0$), všichni studenti budou schopni úlohu splnit v minimálním čase. Čím vyšší je hodnota A, tím větší je rozdíl v potřebném čase řešení mezi studenty s vysokou a nízkou dovedností. Na **spodních obrázcích** jsou zobrazeny řezy vrchních obrázků pro vybrané hodnoty. Vlevo můžeme vidět, že zvyšováním důkladnosti prodlužuje každý, i perfektní, student čas řešení úlohy. Vpravo pak můžeme vidět, vliv Diskriminačního faktoru dovednosti na rozdíl času řešení pro studenty s vysokou a nízkou dovedností.



Obrázek 7.3: Závislost důkladnosti k na času t a dovednosti Θ. Na obrázku **vlevo nahoře** je zobrazen jednoduchý vztah, kdy s rostoucím časem řešení roste důkladnost, kterou student úloze věnoval, a delší čas má větší vliv na důkladnost u studentů s vyšší dovedností. Student s $\Theta = 1$, řešící úlohu po dobu $t = min_t$ řeší přesně s důkladností $k = 1$. Obrázek **vpravo nahoře** zobrazuje vliv Diskriminačního faktoru dovednosti A. Čím více diskriminuje úloha studenty na základě dovednosti, tím méně lze důkladnost „dohnat“ delším časem řešení. Na **spodních obrázcích** můžeme vidět řezy vrchními obrázky pro vybrané hodnoty. Vlevo můžeme vidět, že čas řešení má větší vyšší vliv na důkladnost věnovanou úloze u studentů s vyšší dovedností. Vpravo můžeme vidět, že čím více úloha studenty diskriminuje na základě dovednosti, tím má na důkladnost menší vliv čas strávený řešením úlohy.



Obrázek 7.4: Závislost pravděpodobnosti p na důkladnosti k (resp. času t a dovednosti Θ. **Vlevo nahoře** je zobrazena závislost důkladnosti na času řešení a dovednosti. Na základě výsledné důkladnosti je dále vypočtena pravděpodobnost úspěšného řešení. **Vpravo nahoře** můžeme vidět průběh nenormalizované pravděpodobnosti. Čím nižší dovednost má student nebo čím kratší čas věnuje řešení úlohy, tím více pravděpodobnost klesá až k hodnotě pravděpodobnosti, s jakou lze řešení úlohy uhádnout. Čím vyšší je dovednost studenta nebo čím vyšší čas úloze věnuje, tím pravděpodobnost úspěšného řešení roste. Na **spodních obrázcích** můžeme vidět průběh pravděpodobnosti v normalizovaném tvaru. Při důkladnosti $k = 1$ je pravděpodobnost úspěšného řešení rovna zvolené hodnotě meta-parametru $norpm_p$, zde $norm_p = 0.95$.



Obrázek 7.5: Učení, tj. změna dovednosti Θ. Obrázek zobrazuje jakým způsobem je v této práci modelována změna dovednosti studenta. Hodnoty na svislých osách jsou přírůstky dovednosti studenta po pokusu o vyřešení úlohy. Zda je řešení úspěšné či nikoliv je reflektováno hodnotami parametrů. Na **vrchním** obrázku můžeme vidět, že pokud se dovednost studenta Θ blíží k 0 nebo 1, je změna dovednosti malá a pokud student řeší obtížnější úlohu, je přírůstek jeho dovednosti větší. Na **středním** obrázku můžeme vidět, že změnou koeficientu l_Θ se mění poloha maxima a tedy hodnota dovednosti, při které dochází k nejrychlejšímu učení. Na **spodním obrázku** můžeme vidět, vliv koeficientu l_b, který neovlivňuje tvar křivky, ale pouze velikost přírůstku.

Kapitola 8

Diskuze a další možnosti rozvoje práce

Při zpracování této práce vyvstalo několik možných rozšíření a úprav ITS vytvořeného pro I3T. Možnosti, které by dle autora bylo vhodné prověřit, jsou uvedeny v následujícím seznamu:

1. Vytvořit jiný model znalosti studenta, respektive témat v systému. Např. Q-maticí [THYC20a] nebo pomocí metody hierarchie atributů (*attribute hierarchy method*, AHM) [XHJHh21] z naměřených dat by bylo možné odhadovat postupně parametry těchto modelů. Ve zmíněných modelech je možné zachytit návaznost jednotlivých témat ve formě prerekvizit i efektivnějšího učení jednoho tématu, při znalosti jiného tématu.

2. Metriky cíle ITS v této práci popsané ve vztazích (4.1), (4.2), (4.4) a (4.5) jsou zvoleny s ohledem na omezení práce pro simulovaná data. Volba jiné vhodné metriky může silně ovlivnit výsledný vliv ITS na studenty. Jiné metriky mohou například zahrnout i možnost vynechat některé úlohy a jejich cílem může být dosažení požadované úrovně znalostí, tzv. *mastery learning*, viz kapitola 2.7.1. Při tomto cíli je nutné definovat postup, kterým bude měřena úroveň dovedností studenta. To je možné několika způsoby, např. odhadem z odpovědí na úlohy v průběhu procvičování, nebo zvlášť vytvořenými testy.

3. Jedním z hlavních problémů představeného modelu je velké množství stavů, ve kterých se student, respektive systém může nacházet. Definice stavů může významně ovlivnit chování systému a prozkoumání možností jak stavy reprezentovat stojí dle autora za pozornost. Vzhledem k množství reálných dat, které je možné naměřit při praktickém nasazení systému, je nutná úspornější reprezentace stavů, čímž je na druhou

stranu ztrácena schopnost rozlišit stavy, které mohou potenciálně vykazovat velmi odlišné chování. Možným řešením je odhad Q-hodnoty 3.4 využitím hluboké neuronové sítě (*deep neural network*, DNN). Relativně jednoduchou DNN představili autoři v práci [THYC20a] a [HGS16].

4. V rámci této práce nedošlo k ověření, zda navržený model, popsaný v části 7.1.2 dobře popisuje reálná data. Ověření modelu, případně navržení modelu, který by data popisoval lépe, by vedlo k možnosti generovat věrohodná simulovaná data. Tato data by bylo možné využít při ověřování funkčnosti případných úprav systému a také jako tréninková data pro algoritmy v rámci ITS.

5. Jednou z oblastí, která není v této práci postihnuta je offline zpracování naměřených dat. Například výpočtem podobnosti úloh [Pel19] by bylo možné identifikovat oblasti, pro které je již vytvořeno dostatek úloh a naopak oblasti, pro které je úloh nedostatek. Podobně lze identifikovat úlohy, u kterých studenti vykazují nestandardní chování chování. Taková analýza může upozornit např. na úlohy, které jsou nevhodně formulované.

6. Autoři práce [VRD15] představili možnost dynamické změny parametru γ v Algoritmu 1. Tato úprava by mohla být v ITS vhodná, vzhledem k velmi odlišnému množství informací, které jsou o studentovi známy ze začátku a po delším používání ITS. Ze začátku by mohl systém, právě pomocí úpravy hodnoty parametru γ, cílit na krátkodobější úspěchy studenta a postupem času optimalizovat doporučení vzhledem k dlouhodobějším cílům. To by mohlo vést k větší motivaci studentů, obzvláště při prvních zkušenostech s I3T.

7. Celou velkou oblastí ITS, které není v této práci věnována pozornost, je tzv. vnitřní smyčka, viz část 2.1. Vzhledem k charakteru práce s I3T by vnitřní smyčka mohla být velkým přínosem pro I3T a jeho atraktivitu a efektivitu pro studenty.

Kapitola 9

Závěr

Cílem práce bylo vytvoření tzv. Inteligentního výukového systému (*Intelligent tutor system*, ITS) pro Interaktivní nástroj pro výuku transformací (*Interactive Tool for Teaching Transformation*, I3T). Smyslem činnosti ITS je zefektivnění práce učitele i studenta.

V práci byla navržena tzv. „Vnější smyčka“, kterou je v rámci ITS studentovi vybírána nejvhodnější úloha pro jeho následující postup. Úloha je doporučena na základě výsledků studenta při předchozí práci se systémem, ale zahrnuje také „zkušenosti“ ITS získané při jeho používání předchozími studenty.

Byly sepsány obecné klíčové vlastnosti ITS a dále byla provedena rešerše přístupů a algoritmů používaných pro doporučování úloh. S ohledem na cíl práce: *„vytvořte doplněk, jehož výsledkem bude efektivnější práce studentů s I3T, ve smyslu rychlejšího učení nových dovedností, oproti studentům, kteří úlohy procházejí dle svého uvážení nebo dle stanoveného postupu“*, bylo ze zmíněného přehledu zvoleno využití zpětnovazebního učení (*reinforcement learning*, RL), konkrétně algoritmu Dvojité Q-učení (*Double Q-learning*, DQL).

DQL byl zvolen z důvodu schopnosti přizpůsobení potenciální změně dovedností studentů v průběhu času. Dále pro možnost zahrnutí dlouhodobých cílů. Vyhodnocení úspěšnosti studia i vlivu ITS, je obvykle zaměřeno na cíle v horizontu minimálně jednoho semestru, např. hodnocení studenta za celý předmět. I při definování dlouhodobého cíle je v DQL možné zahrnout současně také krátkodobější cíle, které mohou např. vést k lepší motivaci

studentů.

V průběhu práce na ITS bohužel opustila pracovní skupinu většina zúčastněných. Nepodařilo se dokončit novou verzi I3T včas, aby mohla být k dispozici studentům. Z tohoto důvodu nebylo možné provést měření reálných dat, což znemožnilo splnit jeden z bodů zadání práce. Konkrétně ověření funkčnosti vytvořeného ITS porovnáním různých skupin studentů. Bez reálných naměřených dat nebylo možné měřit reálný dopad využití ITS na efektivitu učení studentů a už vůbec nebylo možné porovnávat výsledky různých skupin, neboť nebylo možné tyto skupiny vytvořit.

Proto bylo po dohodě s vedoucím práce přizpůsobeno zaměření práce nastalým okolnostem. Nejprve byla vytvořena knihovna DIWNE, jako jedna z klíčových součástí I3T, s vidinou možného zprovoznění I3T i v malé skupině pracovníků. Tvorba této knihovny byla časově náročná a bohužel se ani tentokrát nepodařilo I3T dokončit včas. V návaznosti na tyto události byl vytvořen model, pomocí kterého je možné generovat simulovaná data. Struktura a nastavení tohoto modelu bylo vyvozeno z výše zmíněné rešerše. Navíc byl do modelu zahrnut parametr pro nastavení minimálního potřebného času pro řešení úlohy. Dle pohledu autora se tento parametr ve výsledcích studentů, na úlohách v I3T, významně projevuje. Tuto domněnku je ale potřeba ověřit na reálných datech. Cíl ITS při využití simulovaných dat byl stanoven na *nalezení takového pořadí úloh, které povede k úspěšnému splnění všech úloh v co nejkratším čase, přičemž toto pořadí úloh je vytvářeno pro každého studenta zvlášť na základě jeho dosavadních odpovědí a systémem natrénovaných dat.*

Konkrétně byla simulována data o správnosti řešení předložené úlohy a času, který student řešením úlohy strávil. Také bylo nutné zvolit metriku pro měření kvality výběru úloh. Zvolená metrika zahrnuje hodnotu výsledku simulovaného studenta na každé jednotlivé úloze a případně také čas, potřebný pro splnění těchto jednotlivých úloh. Dále započítává celkový čas, který student potřeboval pro úspěšné vyřešení všech úloh v systému. Pro hodnocení výsledku studenta na úloze byla využita tzv. kategorizace výsledku. Kategorizace výsledku umožňuje oddělit hodnocení každé jednotlivé úlohy od rozhraní části ITS, která slouží k doporučování úloh. To bylo důležité především pro univerzalitu vytvořeného systému, neboť množina typů úloh je velmi obsáhlá a stejně tak množina typů dat, která tyto úlohy a jejich řešení generují. Tímto přístupem bylo zajištěno, že systém bude funkční nezávisle na případných nových typech úloh, které se mohou v rámci I3T objevit v budoucnu.

Při spuštění systému na simulovaných datech, vykazoval systém nejprve zcela opačné chování, než bylo očekáváno. S narůstajícím počtem trénovacích dat, systém „úspěšně“ vytvářel taková doporučení, která vedla k významnému nárůstu počtu pokusů a tedy i času stráveném studentem v systému. Z tohoto

výsledku vyplynula funkce pro výpočet odměny, jako jedna ze zásadních části systému. Konkrétně musela funkce odměny penalizovat každý pokus, ať už úspěšný, nebo neúspěšný. Po opravě této chyby pracoval systém očekávaným způsobem. Slibným výsledkem bylo, že systém byl schopen nalézt doporučení, které o 25% až 50% zlepšilo výkon studentů. Tohoto výsledku bylo dosaženo již u přibližně jednoho až dvou tisíců studentů pro 25% a přibližně deseti tisíci studentech pro 50%. Velmi podobné hodnoty systém vykazoval i při zvýšení počtu úloh v systému z šesti na dvacet. Systém je možné dále optimalizovat, některé jeho části nahradit výkonnějšími přístupy, například využití DNN pro odhad Q-hodnoty, a také lze důkladněji nastavit jednotlivé parametry. Vzhledem ke zmíněným skutečnostem jsou, dle pohledu autora, dosažené výsledky slibným ukazatelem pro budoucí vývoj Inteligentního výukového systému, založeného na principu Q-učení, v nástroji I3T i dalších výukových systémech.

Možná rozšíření této práce jsou podrobně sepsána v Příloze 8. navržená rozšíření zahrnují úpravu částí DQL algoritmu, strukturování témat kterých se úlohy týkají, nebo např. analýzu výsledků studentů při řešení úloh, která může vést k identifikaci schopných či slabých studentů nebo nevhodně formulovaných úloh.

Přílohy

Příloha A

Knihovny IMGUI a DIWNE

A.1 IMGUI - Knihovna pro tvorbu uživatelských rozhraní

Immediate Mode Graphical User Interface, známé také jako IMGUI, je typ grafického uživatelského rozhraní, který se často používá pro vývoj her nebo interaktivních aplikací. IMGUI se odlišuje od tradičních grafických uživatelských rozhraní tím, že se skládá z jednotlivých prvků, které jsou vždy znovu vytvářeny na obrazovce při každém snímku, aniž by se uchovávaly v paměti.

IMGUI má několik výhod v porovnání s tradičními grafickými uživatelskými rozhraními. Například, podle studie publikované v časopise *ACM Transactions on Graphics* ([HHBZ04], může IMGUI umožnit rychlejší vývoj grafického uživatelského rozhraní, protože vývojáři nemusí spravovat žádné prvky typů *widget* nebo *layout*. IMGUI také neuchovává žádná data v paměti, což snižuje nároky na paměť a může zlepšit výkon aplikace.

IMGUI má však také několik nevýhod. Například, podle studie publikované v časopise *ACM Transactions on Graphics* [Kil03], může být náročné implementovat IMGUI tak, aby vypadal dobře a byl přívětivý pro uživatele, protože vývojáři musí ručně navrhovat zvlášť každý prvek grafického uživatelského rozhraní. IMGUI také může být obtížné udržovat, protože změny v jednom prvku mohou mít vliv na celé rozhraní.

V poslední době se IMGUI stal populárním pro vývoj her a aplikací [Ope21].

Konkrétní knihovna použitá pro vývoj I3T je DearIMGUI [Cor22].

A.2 Knihovna pro tvorbu editoru grafů DIWNE

Jednou z hlavních částí I3T je okno Pracovní plochy (*Workspace*), viz Obrázek 1.1. Uživatel na Pracovní ploše vytváří jednotlivé moduly a vzájemně je propojuje. Na takovou strukturu lze obecně pohlížet jako na zobrazení grafu, kde moduly jsou uzly (*nodes*) grafu a propojení modulů jsou hrany grafu. Knihovna poskytující rozhraní pro základní operace s tímto grafem je obvykle nazývána „Editor grafů“ (*Node editor*, NE).

Jako nejvhodnější pro I3T byla nejprve vybrána existující knihovna „IMGUI Node Editor“ [Mic22]. V průběhu vývoje však bylo zjištěno, že tato knihovna neumožňuje implementovat vše, co je v rámci I3T potřeba. Jedním z problémů je nemožnost plně kontrolovat grafickou reprezentaci uzlů. V knihovně není možné plné nastavení mezer na různých místech uvnitř uzlu, např. odsazení od levého a horního okraje. To by při použití v I3T vedlo ke zbytečně velkým uzlům s nevyužitým vnitřním prostorem. Na Pracovní ploše I3T se obvykle vyskytuje mnoho uzlů současně a efektivní využití místa je zásadní pro přehlednost vytvořeného grafu. Druhý, zásadnější problémem je nemožnost vykreslit jeden uzel jako vnitřní obsah jiného uzlu. Tato vlastnost je podstatou modulu „Sekvence“, který je navíc jedním ze základních a nejdůležitějších modulů v rámci I3T.

Z důvodů popsaných v předchozím odstavci jsem po domluvě s vedoucím práce vytvořil knihovnu Dear IMGUI Wrapper Node Editor, zkráceně DIWNE. DIWNE je knihovna založena na grafické knihovně DearIMGUI [Cor22]. Podstatou knihovny je obecné rozhraní, funkce a objekty potřebné pro vytvoření „Editoru grafů“ (*node editor*). Editor grafů je grafická reprezentace dat, sestávající z uzlů a hran mezi nimi. Ve skutečnosti vedou hrany mezi přípojnými body, které nazýváme „piny“. Tyto piny jsou součástí každého uzlu, který lze propojit s jiným uzlem a naopak nemohou existovat samostatně. Uzly jsou typicky graficky reprezentovány jednoduchými geometrickými tvary jako je obdélník, elipsa. Uvnitř těchto uzlů, jsou pak zobrazeny informace nesené tímto uzlem. V kontextu I3T jsou to např. název uzlu, 16 hodnot matice reprezentující geometrickou transformaci, názvy vstupních a výstupních hodnot na jednotlivých pinech a pod. Spojení uzlů je graficky zobrazeno jako křivka začínající a končící ve dvou spojovaných uzlech, respektive jejich

pinech.

Knihovna poskytuje základní funkcionalitu pro práci s takto vytvořeným grafem. Je možné vybírat/označovat jednotlivé uzly a hrany. Přidávat a mazat uzly. Spojovat existující uzly hranami a mazat existující hrany. Přesouvat uzly po neomezené pracovní ploše, při automatickém zachování jejich propojení.

Vývojář využívající DIWNE má možnost libovolně měnit grafické zobrazení uzlů, pinů i hran. Uvnitř uzlů, pinů i hran lze zobrazovat libovolné prvky v rámci DearIMGUI a knihovna je otevřena pro možné rozšíření funkcionality, potřebné pro konkrétní projekt.

A.2.1 Implementace knihovny DIWNE

Základní, velmi stručný, přehled knihovny zobrazuje Obrázek A.1. Veškeré objekty, tedy Uzel *Node*, Hrana *Link*, „Spojovací pin“ *Pin* i samotná instance Editoru grafů *diwne*, dědí společné vlastnosti od abstraktní třídy „DiwneObject“. Společně vlastnosti jsou např. možnost být zaměřen myší *focus*, dále funkce pro vykreslení své grafické reprezentace na obrazovku „drawDiwne()“ a pod. Každá z tříd, která je níže v hierarchii dědičnosti pak přepisuje (*override*) konkrétní funkce, dle specifik daného objektu. Principem využití knihovny DIWNE je vytvoření vlastních objektů, dědících od objektů knihovny DIWNE. V těchto vlastních objektech jsou, kromě děděných atributů, uloženy atributy specifické pro konkrétní projekt. Dále také knihovna DIWNE umožňuje přepsat funkce připravené v jejích třídách. Takovou funkcí je např. „content()“, ve které jsou vykresleny konkrétní prvky knihovny DearIMGUI, které reprezentují daný objekt. Další takovou funkcí je např. „processFocused()“, která je spuštěna ve chvíli, kdy je objekt na obrazovce zaměřen kurzorem myši. Jako poslední příklad z mnoha dalších uveďme funkci „bypassFocusAction()“, která předepisuje pravidlo, podle kterého je rozhodnuto, zda je objekt zaměřen kurzorem myši či nikoliv.

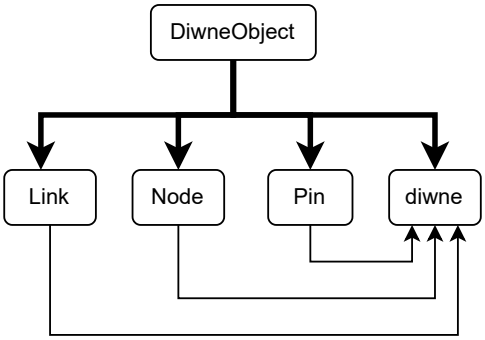
Jednou z klíčových vlastností přístupu IMGUI v knihovně DearIMGUI je, že akce provedené s konkrétním prvkem jsou spuštěny okamžitě po vykreslení tohoto prvku. Stejně tak je okamžitě po vykreslení prvku detekováno, zda a která akce má být spuštěna. V uživatelském rozhraní, které pouze zobrazuje stav systému a odesílá do systému pokyny pro změnu jeho stavu, s touto vlastností není žádný problém. V Pracovní ploše I3T však nastává komplikace. Jednotlivé prvky spolu interagují, např. hranu lze zapojit do pinu, a navíc spolu mohou interagovat v libovolném pořadí, ve smyslu pořadí ve kterém jsou vykreslovány na obrazovce. Komplikace nastává ve chvíli, kdy je vykreslen

pin a až poté hrana, podle které má pin změnit své vlastnosti, např. zobrazit popisek, zda je tuto hranu možné do pinu zapojit. Kontrola, zda má být spuštěna daná akce na zmíněném pinu, je totiž dokončena dříve, než je spuštěna akce na kterou má tento pin reagovat. V přístupu IMGUI je proto nutné držet také informace o akcích z předchozího vykreslovaného snímku (*frame*). V Editoru grafů se mohou jednotlivé prvky překrývat. A to nejen prvky různých typů, ale také mohou být přes sebe zobrazeny např. dva uzly. Tato situace také činí v přístupu IMGUI problémy, protože každý vykreslovaný prvek a interakce s ním, jsou zpracovávány nezávisle na ostatních prvcích. Příklad problémové sitace je následující: Na druhém vykreslovaném (vrchním) prvku je spuštěna akce - například zaměření prvku kurzorem, která má blokovat interakci s dalšími prvky - zaměřený kurzorem může být vždy pouze jeden prvek. V dalším snímku však první vykreslovaný (spodní) uzel nemá možnost zjistit, zda bude nebo nebude spuštěna tato akce na uzlu vrchním. Vrchní uzel ještě nebyl v tomto snímku vykreslen a tedy ani nebyla detekována žádná akce na tomto vrchním uzlu. V Editoru grafů je tedy, podobně jako v předchozím případě, nutné držet uloženou informaci o akcích z předchozího snímku a reagovat na ně ve snímku aktuálním. Situace je však ještě o něco složitější. Při některých akcích, např. tažení hrany, tato akce některé jiné akce blokuje, např. zaměření uzlu, ale jiné akce musí být možné, např. zaměření pinu a případné zapojení hrany. Pro vyřešení těchto komplikací má každý objekt uložený ukazatel na společný objekt editoru, objekt „diwne“. Skrze tento společný objekt je poté možné přenášet informace mezi jednotlivými objekty v editoru a také mezi jednotlivými snímky.

Samostatnou kapitolou je možnost přizpůsobení ovládání jednotlivých prvků, potřebám konkrétního projektu. Příkladem je např. nastavení, zda se mají uzly přesouvat levým, nebo pravým tlačítkem myši, nebo jakkoliv jinak. V knihovně DIWNE je tento problém vyřešen vytvořením funkcí typu „bypass...()“, které určují při jaké události je daná akce spuštěna. Například funkce „bypassHoldAction()“ určuje, jaká událost vede k „chycení“ objektu, například uzlu. Tyto „bypass“ funkce mají své základní nastavení, ale je možné je přepsat v objektech, které dědí od objektů z DIWNE, a tím změnit chování objektů v konkrétním projektu.

Množství různých objektů v knihovně DIWNE není velké - skládá se z výše zmíněných čtyř objektů a jedné abstraktní třídy, viz také Obrázek A.1. Obtížnost vývoje knihovny spočívá především ve velkém množství různých akcí s vykreslovanými prvky a také mnoha různých interakcí prvků mezi sebou. U těchto akcí a interakcí je velké množství výjimek a speciálních situací, jejichž odhalení a hlavně konceptuální řešení je obtížné. Příkladem budiž situace: *Pokud je tažen jeden uzel, žádný další již na zaměření myší nereaguje*[1]*. Kromě uzlu typu „Sekvence“, do kterého je potřeba tažený uzel*

[1]Při rychlém pohybu myši při držení uzlu se může stát, že kurzor bude po dobu jednoho snímku mimo uzel, který je tažen.



Obrázek A.1: Přehled prvků v knihovně DIWNE. Nadřazenou abstraktní třídou pro všechny objekty je třída „DiwneObject“. Vztah dědičnosti je naznačen tlustými šipkami. Objekt „diwne“ je unikátní, respektive každý takový objekt reprezentuje jednu instanci celého Editorů grafů. Každý objekt typu Uzel *Node*, Hrana *Link* a „Spojovací pin“ *Pin* může být přiřazen pouze k jedné instanci Editoru. tedy objektu „diwne“. Každý z těchto objektů má uloženu referenci na objekt „diwne“, tedy na instanci editoru, do které daný objekt patří. To je v obrázku naznačeno slabou šipkou.

vložit a tedy Sekvence musí na zaměření myší reagovat. Ale musí reagovat pouze detekcí pozice taženého uzlu a zda byl uživatelem „puštěn“ nebo ne, například „sebrat“ zaměření taženému uzlu nesmí, neboť tím by došlo k jeho puštění. Ale to celé jen v případě, že již uzel není nad jinou Sekvencí... To celé také platí pouze v případě, že tažený uzel je typu „Transformace“, neboť uzly jiného typu nelze do Sekvence vložit. Podobných situacích, ve kterých se projeví mnoho různých podmínek, je v Editoru grafů celá řada. Podmínky jsou často provázané a řešení jedné situace způsobilo vznik problému v situaci jiné.

Příloha B

Literatura

[Ack94] Terry A Ackerman, *Using multidimensional item response theory to understand what items and tests are measuring*, Applied Measurement in Education **7** (1994), no. 4, 255–278.

[Bak01] Frank B Baker, *The basics of item response theory*, ERIC, 2001.

[BCK10] Ryan SJd Baker, Albert T Corbett, and Kenneth R Koedinger, *Intelligent tutoring goes to school in the big city*, International Journal of Artificial Intelligence in Education **20** (2010), no. 2, 167–207.

[BDK13] Robert A Bjork, John Dunlosky, and Nate Kornell, *Self-regulated learning: Beliefs, techniques, and illusions*, Annual Review of Psychology **64** (2013), 417–444.

[BKRI08] Andrew C Butler, Jeffrey D Karpicke, and Henry L Roediger III, *Correcting a metacognitive error: feedback increases retention of low-confidence correct responses*, Journal of Experimental Psychology: Learning, Memory, and Cognition **34** (2008), no. 4, 918.

[BKV07] R Bell, Y Koren, and C Volinsky, *The bellkor solution to the netflix prize*, Netflix Prize Progress Award (2007), –.

[Bru96] Peter Brusilovsky, *Methods and techniques of adaptive hypermedia*, User Modeling and User-Adapted Interaction **6** (1996), no. 2-3, 87–129.

[CA95] A Corbett and J Anderson, *Knowledge tracing: Modeling the acquisition of procedural knowledge*, User modeling and user-adapted interactions 4, 1995, pp. 253–278.

[CCL16] Xiaodong Chen, Hui Cen, and Yanchun Liu, *A reinforcement learning approach to personalized task recommendation in a collaborative learning system*, IEEE Transactions on Learning Technologies **9** (2016), no. 1, 3–15.

[Cor22] Omar Cornut, *Dearimgui*, `https://github.com/ocornut/imgui`, 2022, Online; Cit. 8.1.2023.

[Csi97] M Csikszentmihalyi, *Creativity: Flow and the psychology of discovery and invention*, Harper Perennial, 1997.

[DB12] Michel C Desmarais and Ryan SJ Baker, *A review of recent advances in learner and skill modeling in intelligent learning environments*, User Modeling and User-Adapted Interaction **22** (2012), no. 12, 9–38.

[DDGR07] A Das, M Datar, A Garg, and S Rajaram, *The google news personalization: Scalable online collaborative filtering*, WWW 07: the 16th International Conference on World Wide Web, 2007, pp. 271–280.

[Elo78] Arpad E Elo, *The rating of chessplayers, past and present*, vol. 3, Batsford London, 1978.

[FMF+18] Petr Felkel, Alejandra J. Magana, Michal Folta, Alexa Gabrielle Sears, and Bedrich Benes, *I3T: Using Interactive Computer Graphics to Teach Geometric Transformations*, EG 2018 - Education Papers (Frits Post and Jiri Zara, eds.), The Eurographics Association, 2018.

[Gus00] Thomas R Guskey, *Evaluating professional development*, Corwin Press, 2000.

[Has10] Hado Hasselt, *Double q-learning*, Advances in Neural Information Processing Systems (J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, eds.), vol. 23, Curran Associates, Inc., 2010.

[HGS16] Hado van Hasselt, Arthur Guez, and David Silver, *Deep reinforcement learning with double q-learning*, Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16, AAAI Press, 2016, p. 2094–2100.

[HHBZ04] Aaron Hertzmann, Hugues Hoppe, Ronen Barzel, and Denis Zorin, *Immediate mode graphical user interfaces*, ACM Transactions on Graphics **23** (2004), no. 3, 633–641.

[HT07] John Hattie and Helen Timperley, *The power of feedback*, Review of Educational Research **77** (2007), no. 1, 81–112.

[Jar07] Petr Jarušek, *Modeling problem solving times in tutoring systems [online]*, Disertační práce, Masarykova univerzita, Fakulta informatiky Brno, 2013 [cit. 2023-01-07], SUPERVISOR: prof. RNDr. Ivana Černá, CSc.

[JDE+22] Hunter John, Dale Darren, Firing Eric, Droettboom Michael, and the Matplotlib development team, *Pyplot*, `https://matplotlib.org/stable/tutorials/introductory/pyplot.html`, 2022, Online; Cit. 2023-01-07.

[JGr22] JGraph Ltd., *Drawio*, `https://app.diagrams.net/`, 2022, Online; Cit. 2023-01-07.

[KB08] Nate Kornell and Robert A Bjork, *Optimising self-regulated study: The benefits—and costs—of dropping flashcards*, Memory **16** (2008), no. 2, 125–136.

[Kil03] Mark J Kilgard, *The opengl graphics system: A specification*, 3 ed., Addison-Wesley, Reading, MA, 2003.

[KMR18] Sunil Kumar, Abhilasha Mishra, and Rohit Rai, *Adaptive task recommendation in e-learning systems using reinforcement learning*, 2018 International Conference on Computing, Communication and Automation (ICCCA), IEEE, 2018, pp. 1–6.

[LKM14] Ran Liu, Kenneth R Koedinger, and Elizabeth A McLaughlin, *Interpreting model discovery and testing generalization to a new dataset*, Proc. of Educational Data Mining, 2014, pp. 107–113.

[LPFK13] Derek Lomas, Kishan Patel, Jodi L Forlizzi, and Kenneth R Koedinger, *Optimizing challenge in an educational game using large-scale design experiments*, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 2013, pp. 89–98.

[LSY03] G Linden, B Smith, and J York, *Amazon.com recommendations: Item-to-item collaborative filtering*, IEEE Internet Computing (2003), –.

[Mic22] Cichoń Michal, *Imgui node editor*, `https://github.com/thedmd/imgui-node-editor`, 2022, Online; Cit. 2023-01-08.

[Ope21] OpenAI, *Information provided by the openai language model assistant*, 2021, Accessed on 27.12.2022 - 10.1.2023, checked, eventualy corrected and paraphrased.

[PA05] Philip I Pavlik and John R Anderson, *Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect*, Cognitive Science **29** (2005), no. 4, 559–586.

[PA08] _____, *Using a model to compute the optimal schedule of practice*, Journal of Experimental Psychology: Applied **14** (2008), no. 2, 101–117.

[Pap15] Jan Papoušek, *Computerized adaptive practice of factual knowledge*, Ph.D. thesis, Masarykova univerzita, Fakulta informatiky, 2015, Rigorózní práce.

[PBH+5a] Christopher Piech, John Bassen, J Huang, S Ganguli, M Sahami, LJ Guibas, and J Sohl-Dickstein, *Deep knowledge tracing*, Proceedings of Advances in Neural Information Processing Systems, 2015a, pp. 505–513.

[PE20] Radek Pelánek and Thorsten Effenberger, *Beyond binary correctness: Classification of students' answers in learning systems*, User Modeling and User-Adapted Interaction **30** (2020), 867–893.

[Pel15] Radek Pelánek, *Metrics for evaluation of student models*, Journal of Educational Data Mining (2015), –, To appear.

[Pel19] _____, *Measuring similarity of educational items: An overview*, IEEE Transactions on Learning Technologies **PP** (2019), 1–1.

[Piz07] Z Pizlo, *Human problem solving in 2006*, The Journal of Problem Solving **1** (2007), no. 2, 3.

[Pv17] Radek Pelánek and Jiří Řihák, *Experimental analysis of mastery learning criteria*, Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization, ACM, 2017, pp. 156–163.

[Pv18] _____, *Analysis and design of mastery learning criteria*, New Rev. Hypermed. Multimed. **24** (2018), 133–159.

[PWT12] Zachary A Pardos, Qing Yang Wang, and Shubhendu Trivedi, *The real world significance of performance prediction*, Proc. of Educational Data Mining, 2012.

[Pyt22] Python Software Foundation, *Python*, `https://www.python.org/`, 2022, Online; Cit. 2023-01-07.

[Řih17] Jiří Řihák, *Modeling techniques for adaptive practice systems [online]*, Doctoral theses, dissertations, Masaryk University, Faculty of InformaticsBrno, 2017, SUPERVISOR: doc. Mgr. Radek Pelánek, Ph.D.

[Tak08] Gábor Takács, *Matrix factorization and neighbor based algorithms for the netflix prize problem*, Proceedings of the 2008 ACM conference on Recommender systems, 2008.

[THYC20a] C Tan, R Han, R Ye, and K Chen, *Adaptive learning recommendation strategy based on deep q-learning*, Applied Psychological Measurement **44** (2020), no. 4, 251–266.

[THYC20b] Cheng Tan, Rui Han, Ru Ye, and Kui Chen, *Adaptive learning recommendation strategy based on deep q-learning*, Applied Psychology: Measurement **44** (2020), no. 4, 251–266.

[TWC+19] Dongbo Tu, Shiyu Wang, Yan Cai, Jeff Douglas, and Hua-Hua Chang, *Cognitive diagnostic models with attribute hierarchies: Model estimation with a restricted q-matrix design*, Applied psychological measurement **43** (2019), no. 4, 255–271.

[Van06] K Vanlehn, *The behavior of tutoring systems*, International Journal of Artificial Intelligence in Education **16** (2006), no. 3, 227–265.

[vdL06] Wim J van der Linden, *A lognormal model for response times on test items*, Journal of Educational and Behavioral Statistics **31** (2006), no. 2, 181–204.

[vdL09a] ———, *Conceptual issues in response-time modeling*, Journal of Educational Measurement **46** (2009), no. 3, 247–272.

[VDL09b] Wim J. Van Der Linden, *Conceptual issues in response-time modeling*, Journal of Educational Measurement **46** (2009), no. 3, 247–272.

[VRD15] François-Lavet Vincent, Fonteneau Raphaël, and Ernst Damien, *How to discount deep reinforcement learning: Towards new dynamic strategies*, CoRR **abs/1512.02011** (2015), –.

[WD92] Christopher JCH Watkins and Peter Dayan, *Q-learning*, Machine learning **8** (1992), no. 3, 279–292.

[WGZ+19] J Wang, L Gou, W Zhang, H Yang, and HW Shen, *Deepvid: Deep visual interpretation and diagnosis for image classifiers via knowledge distillation*, IEEE Transactions on Visualization and Computer Graphics **25** (2019), no. 6, 2168–2180.

[WSLP17] L Wang, A Sy, L Liu, and C Piech, *Learning to represent student knowledge on programming exercises using deep learning*, Proceedings of Educational Data Mining, 2017, pp. 324–329.

[XHJHh21] Li Xiao, Xu Hanchen, Zhang Jinming, and Chang Hua-hua, *Optimal hierarchical learning path design with reinforcement learning*, Applied Psychological Measurement **45** (2021), no. 1, 54–70.

[YK13] Michael V Yudelson and Kenneth R Koedinger, *Estimating the benefits of student model improvements on a substantive scale*, EDM 2013 Workshops Proceedings, 2013.

[YXJZ18] Chen Yunxiao, Li Xiaoou, Liu Jingchen, and Ying Zhiliang, *Recommendation system for adaptive learning*, Applied Psychological Measurement **42** (2018), no. 1, 24–41, PMID: 29335659.

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Holeček** Jméno: **Jaroslav** Osobní číslo: **406290**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Specializace: **Umělá inteligence**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Adaptivní učení v softwarovém nástroji I3T pro výuku geometrických transformací

Název diplomové práce anglicky:

Adaptive learning in I3T software for education of geometric transformations

Pokyny pro vypracování:

Interaktivní nástroj na výuku transformací (I3T) obsahuje řadu úloh. Obsahuje také doplněk pro zachytávání uživatelských akcí - logger.
Vytvořte doplněk, který bude zpracovávat data získaná loggerem u aktuálního uživatele a od předchozích uživatelů. Na základě těchto dat bude doplněk navrhovat aktuálnímu uživateli pro něj nejvhodnější úlohy, které mu program I3T nabízí.
Proveďte rešerši aktuálních metod, algoritmů a přístupů, které se pro tyto účely využívají, a na základě výsledku rešerše navrhněte a implementujte vhodnou metodu pro nástroj I3T.
Výsledkem doplňku by měla být efektivnější práce studentů s nástrojem I3T - student se učí nové dovednosti a znalosti rychleji, než při procházení úloh dle svého uvážení, nebo procházením na základě pevně stanoveného průchodu.
Ověřte tuto hypotézu na třech skupinách studentů: První používá doplněk I3T, druhá volí úlohy dle svého uvážení, třetí se drží pevně stanoveného pořadí úloh.

Seznam doporučené literatury:

1. Petr Felkel, Alejandra Magana, Michal Folta, Alexa Gabrielle Sears, Bedrich Benes I3T: Using Interactive Computer Graphics to Teach Geometric Transformations. Eurographics Education Papers 2018, http://www.i3t-tool.org/
2. R. Pelánek, "Measuring Similarity of Educational Items: An Overview," in IEEE Transactions on Learning Technologies, vol. 13, no. 2, pp. 354-366, 1 April-June 2020, doi: 10.1109/TLT.2019.2896086.
3. Pelánek, R., Effenberger, T. Beyond binary correctness: Classification of students' answers in learning systems. User Model User-Adap Inter 30, 867–893 (2020). https://doi.org/10.1007/s11257-020-09265-5
4. Papoušek, J., (2017) Computerized Adaptive Practise of Factual Knowledge, Doctoral thesis, Masaryk University, https://www.fi.muni.cz/adaptivelearning/?a=publications
5. Řihák. J., (2017), Modeling Techniques for Adaptive Practice Systems, Doctoral thesis, Masaryk University, https://www.fi.muni.cz/adaptivelearning/?a=publications
6. Jarušek, P., (2013), Modeling Problem Solving Times in Tutoring Systems, Doctoral thesis, Masaryk University, https://www.fi.muni.cz/adaptivelearning/?a=publications

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Petr Felkel, Ph.D. Katedra počítačové grafiky a interakce

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **21.02.2021** Termín odevzdání diplomové práce: **10.01.2023**

Platnost zadání diplomové práce: **19.02.2023**

Ing. Petr Felkel, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

.	
Datum převzetí zadání	Podpis studenta