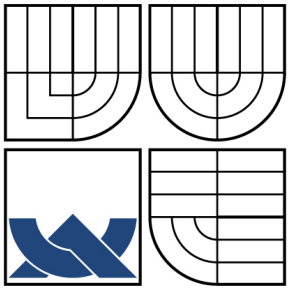
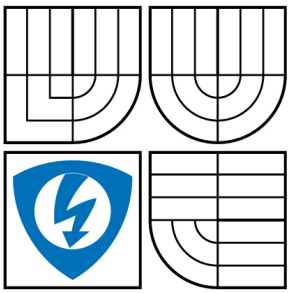


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

EXPERTNÍ SYSTÉMY
ES PRO SAMOSTATNÉ STUDIUM A JEHO VYHODNOCENÍ

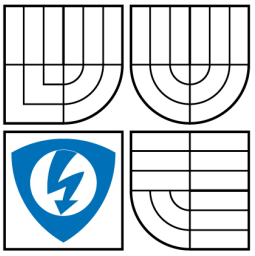
EXPERT SYSTEMS
ES FOR HOME STUDY AND EVALUATION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. JAROSLAV NOVÁK
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. VÁCLAV JIRSÍK, CSc.
SUPERVISOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Jaroslav Novák	***ID:***	83410
Ročník:	2	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Expertní systémy
ES pro samostatné studium a jeho vyhodnocení

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je vytvoření a odladění expertního systému pro samostatné (domácí) studium středoškolské látky s automatickým vyhodnocením úspěšnosti a doporučením pro další studium. Expertní systém bude využívat rámcové reprezentace znalostí.

DOPORUČENÁ LITERATURA:

Mařík, Štěpánková, Lažanský: Umělá inteligence
Leondes: Fuzzy Logic and Expert Systems
Giarratano, Riley: Expert systems - principles and programming
Turban, Aronson: Decision Support Systems and Intelligent Systems PReasoning

Termín zadání:	9.2.2009	***Termín odevzdání:***	25.5.2009

Vedoucí práce: doc. Ing. Václav Jirsík, CSc.

prof. Ing. Pavel Jura, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práve třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Anotace

Tato diplomová práce obsahuje základní informace ohledně znalostních a expertních systémů. Dále se zde zabývám architekturou expertních systémů a důkladně jsou zde popsány způsoby reprezentace znalostí pro expertní systémy. Ke každému způsobu reprezentace znalostí jsou zde zpracovány demonstrující jednotlivé způsoby reprezentace znalostí. Na závěr je zde popsán vytvořený expertní systém, který ukládá znalosti pomocí rámcové reprezentace znalostí.

Annotation

This master thesis contains the basic information about knowledge and expert systems. The thesis contains theoretic text about architecture of the expert systems and representation knowledge. The text regarding on representation knowledge contains examples of different ways of knowledge representation for expert systems. In the next part is described the design and all functions of the expert systems. This expert system uses frames representation.

Abstrakt

V diplomové práci se zabývám expertními systémy a jejich návrhem se specializací na reprezentaci znalostí pro expertní systémy. Diplomová práce obsahuje teoretický text ohledně architektury expertních systémů a reprezentace znalostí. Text o reprezentaci znalostí obsahuje příklady jednotlivých způsobů reprezentace znalostí pro expertní systémy. Hlavním cílem diplomové práce bylo vytvořit jednoduchý expertní systém pro samostudium. V další části práce je detailně popsán návrh a všechny funkce tohoto expertního systému. V poslední části je popsán návrh jednoduché báze znalostí, na které demonstruji funkci programu. Tento expertní systém využívá rámcovou reprezentaci znalostí.

Klíčová slova: Expertní systémy, umělá inteligence, reprezentace znalostí, znalostní systémy, predikátová logika, produkční systémy, sémantické sítě, rámce.

Abstract

In my master thesis I am concerned with the expert systems and their design specialized in knowledge for expert systems. Thesis contains theoretic text about the architecture of the expert systems and about the representation knowledge. The text regarding on representation knowledge contains examples of different ways of representation knowledge for expert systems. The main goal of my master thesis was to create a simple expert system that is useful for self-study. Further is described the design and all functions of the expert systems. In the final part I have described design of knowledge base. This expert system uses frames representation.

Key words: expert systems, artificial intelligence, knowledge representation, knowledge systems, predicate logic, production systems, semantics nets, frames.

Bibliografická citace

NOVÁK, J. *Expertní systémy ES pro samostatné studium a jeho vyhodnocení*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 104 s. Vedoucí diplomové práce doc. Ing. Václav Jirsík, CSc.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma "Expertní systémy ES pro samostatné studium a jeho vyhodnocení" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne : Podpis:

Poděkování

Na tomto místě chci poděkovat vedoucímu mého semestrálního projektu, kterým byl doc. Ing. Václav Jirsík, CSc., a za rady a konzultace ohledně vypracování práce Ing. Petru Poláchovi. Dále bych chtěl poděkovat RNDr. Naděždě Uhdeové, Ph.D a RNDr. Evě Hradilové za jejich připomínky k programu z pohledu expertek. Nakonec děkuji kamarádům a kamarádkám na kterých jsem testoval vytvořenou bázi znalostí. Byly to: Bc. Jan Neužil, Bc. Jaroslav Horák, Petr Svoboda, Petr Novák, Kateřina Bittnerová, Jakub Navařík, Martina Míková, Marek Matiáš, Radek Baránek a Veronika Šeděnková.

Další dík patří mým rodičům, kteří mi poskytli zázemí při vypracovávání této diplomové práce a hlavně v průběhu celého studia.

OBSAH:

1. ÚVOD 13

2. ZNALOSTNÍ SYSTÉMY 14

3. EXPERTNÍ SYSTÉMY 15

3.1 POUŽITELNOST EXPERTNÍCH SYSTÉMŮ V PRAXI 15

3.1.1 Které kategorie problémů je vhodné řešit pomocí expertních systémů? 15

3.1.2 Kdy je vhodné použít expertní systém? 16

3.1.3 Kdy je účelné vytvořit expertní systém? 17

4. ARCHITEKTURA EXPERTNÍCH SYSTÉMŮ 18

4.1 ZÁKLADNÍ SLOŽKY EXPERTNÍCH SYSTÉMŮ 18

4.1.1 Řídící mechanismus 18

4.1.2 Báze znalostí 20

4.1.3 Báze dat 21

4.2 PŘÍDAVNÉ SLOŽKY EXPERTNÍHO SYSTÉMU 22

4.2.1 Komunikační modul 22

4.2.2 Vysvětlovací modul 23

4.2.3 Generátor výsledků 23

4.3 DOPLŇKOVÉ SLOŽKY EXPERTNÍHO SYSTÉMU 24

4.3.1 Modul expertních údajů 25

4.3.2 Modul expertních programů 25

4.4 PROSTŘEDKY VYTVÁŘENÍ A UDRŽOVÁNÍ BÁZE ZNALOSTÍ 26

4.4.1 Modul definice báze znalostí 26

4.4.2 Modul modifikace báze znalostí 26

4.4.3 Modul kontroly báze znalostí 27

4.4.4 Modul prohlížení báze znalostí 27

5. REPREZENTACE ZNALOSTÍ 28

5.1 REPREZENTACE ZNALOSTÍ PROSTŘEDKY PREDIKÁTOVÉ LOGIKY 30

5.2 REPREZENTACE ZNALOSTÍ PRODUKČNÍMI SYSTÉMY 32

5.2.1 Reprezentace znalostí produkčními systémy - Příklad č.1.: gramatika anglického jazyka 33

5.3 REPREZENTACE ZNALOSTÍ SÉMANTICKÝMI SÍTĚMI 36

5.3.1 Reprezentace znalostí sémantickou sítí - Příklad č.1.: Škoda Favorit 37

5.4 RÁMCOVÁ REPREZENTACE ZNALOSTÍ 38

5.4.1 Terminologie rámcové reprezentace znalostí ... 38
5.4.2 Hierarchie rámců ... 40
5.4.3 Propojování rámců pomocí položky „is-a“ ... 42
5.4.4 Dědičnost vlastností rámců ... 42
5.4.5 Výhody rámců ... 43
5.4.6 Rámcová reprezentace znalostí - Příklad č.1.: Škoda Favorit ... 44
5.4.7 Rámcová reprezentace znalostí - Příklad č.2.: Řízení a Regulace 1 ... 45
5.4.8 Rámcová reprezentace znalostí - Příklad č.3.: byt ... 45

6. POPIS VYTVOŘENÉHO EXPERTNÍHO SYSTÉMU ... 49

6.1 ZADÁNÍ PRO VYTVOŘENÍ EXPERTNÍHO SYSTÉMU ... 49
6.2 ÚVOD ... 49
6.3 KOMUNIKAČNÍ MODUL ... 50
6.3.1 Zadávání teorie ... 50
6.3.2 Zadávání otázek ... 52
6.3.3 Zkoušení ... 55
6.4 BÁZE ZNALOSTÍ ... 56
6.4.1 Třída kapitola ... 57
6.4.2 Třída otázka ... 58
6.4.3 Třída baze_znalosti ... 59
6.5 ROZHODOVACÍ MECHANISMUS ... 60
6.5.1 Funkce pro výběr otázky ... 60
6.5.2 Funkce pro výpočet hodnoty umí u vybraných kapitol ... 72
6.5.3 Hranice úspěšnosti ... 80
6.6 VYSVĚTLOVACÍ MODUL A GENERÁTOR VÝSLEDKŮ ... 80
6.7 NÁPADY PRO DALŠÍ VERZE PROGRAMU ... 82
6.8 KONZULTACE S EXPERTEM ... 82

7. NÁVRH BÁZE ZNALOSTÍ ... 85

7.1 POTŘEBNÉ TEORETICKÉ ZNALOSTI ... 85
7.2 OTÁZKY A NASTAVENÍ VAZEB ... 85
7.3 TESTOVÁNÍ VYTVOŘENÉ BÁZE ZNALOSTÍ ... 86
7.4 VYHODNOCENÍ TESTOVÁNÍ ... 89

8. ZÁVĚR ... 91

9. SEZNAM POUŽITÉ LITERATURY ... 93

10. SEZNAM PŘÍLOH ... 95

Seznam obrázků:

Obrázek 4.1.: Základní složky expertních systémů……18

Obrázek 4.2.: Základní a přídavné složky expertních systémů……. 22

Obrázek 4.3.: Základní, přídavné a doplňkové složky expertních systémů.Základní a přídavné složky expertních systémů…….. 24

Obrázek 5.1.: Hierarchie znalostí…….. 29

Obrázek 5.2.: Syntaktický strom……35

Obrázek 5.3.: Příklad reprezentace automobilu Škoda Favorit sémantickou sítí……37

Obrázek 5.4.: Ukázka hierarchie rámcové reprezentace znalostí na příkladu reprezentace znalostí vozidla……..41

Obrázek 5.5.: Příklad rámce osobního auta Škoda Favorit……44

Obrázek 5.6.: Příklad rámce předmětu Řízení a Regulace 1……..45

Obrázek 5.7.: Obrázek znázorňující byt…….45

Obrázek 5.8.: Rámec reprezentující byt z *obrázku 5.7*……..46

Obrázek 5.9.: Obrázek znázorňující pokoj z popisovaného bytu……...46

Obrázek 5.10.: Rámec reprezentující pokoj z *obrázku 5.9.*……47

Obrázek 5.11.: Příklad rámce reprezentujícího televizi umístěnou v pokoji..47

Obrázek 5.12.: Příklad rámce reprezentujícího okno_1 v pokoji……...48

Obrázek 5.13.: Příklad rámce reprezentujícího *dveře_pokoj_předsíň* v pokoji……..49

Obrázek 6.1.: Přepínaní mezi jednotlivými módy programu…….50

Obrázek 6.2.: Obrazovka programu při zapnutém módu „*Zadávání teorie*“.51

Obrázek 6.3.: Obrazovka programu při zapnutém módu „*Zadávání otázek*“.53

Obrázek 6.4.: Obrazovka programu po stisknutí tlačítka *Tabulka vazeb* v módu *Zadávání otázek*…….54

Obrázek 6.5.: Obrazovka programu po spuštění modu zkoušení, kde student vybere kapitolu ze seznamu kapitol a zadává identifikační údaje……..55

Obrázek 6.6.: Obrazovka programu po zadání identifikačních údajů, kdy student odpovídá na otázky, které program generuje…….56

Obrázek 6.7.: Grafické znázornění tříd vygenerované programem C#.....57

Obrázek 6.8.: Znázornění způsobu výběru otázek v programu……..61

Obrázek 6.9.: Obrazovka nastavení algoritmu „*Výběr*“…….66

Obrázek 6.10.: Obrazovka nastavení algoritmu „*Všechny z vybrané a ostatní orientačně*“……..67

Obrázek 6.11.: Obrazovka nastavení algoritmu „*Rozhodování podle umí*“...68

Obrázek 6.12.: Obrazovka nastavení algoritmu „*Rychlé zjištění umí*“……...69

Obrázek 6.13.: Obrazovka nastavení algoritmu „*Rychlé zjištění neumí*“…...70

Obrázek 6.14.: Znázornění výběru algoritmu pro výpočet hodnoty umí…...72

Obrázek 6.15.: Na grafu je vidět chování algoritmu č.1 v případě, že všechny otázky mají stejnou vazbu (důležitost otázky) na zkoušenou kapitolu. První tři otázky odpověděl student dobře a všechny ostatní špatně…….74

Obrázek 6.16.: Na grafu je vidět chování algoritmu č.2. v případě, že všechny otázky mají stejnou vazbu (důležitost otázky) na zkoušenou kapitolu. Obtížnost otázek je zde nastavena na 0,5 (dvě možné odpovědi), aby se co nejvíce projevil rozdíl mezi touto metodou výpočtu umí a metodou minulou…….76

Obrázek 6.17.: Na grafu je vidět chování algoritmus č.3. v případě, že všechny otázky mají stejnou vazbu (důležitost otázky) na zkoušenou kapitolu…….78

Obrázek 6.18.: V tomto grafu jsou sloučeny všechny tři předchozí verze do jednoho pro lepší srovnání toho jak se který chová při stejných vstupních datech…79

Obrázek 6.19.: Obrazovka nastavení hranice úspěšnosti……....80

Seznam tabulek:

Tabulka 5.1.: Pravdivostní tabulka výrokových formulí……...31

Tabulka 6.1.: Znázorňuje vzájemné vazby mezi kapitolami A, B, C, D, E, F, G a otázkami 1 až 15, pro výběr otázek: „*Všechny od nejsložitější*“…….62

Tabulka 6.2.: Znázorňuje vzájemné vazby mezi kapitolami A, B, C, D, E, F, G a otázkami 1 až 15, pro výběr otázek: „*Všechny od nejjednodušší*“……..63

Tabulka 6.3.: Znázorňuje vzájemné vazby mezi kapitolami A, B, C, D, E, F, G a otázkami 1 až 15, pro výběr otázek: „*Všechny kapitolu po kapitole*“…….65

Tabulka 6.4.: Znázorňuje vzájemné vazby mezi kapitolami A, B, C, D, E, F, G a otázkami 1 až 15, pro výběr otázek: „*Všechny z vybrané a ostatní orientačně*“……..67

Tabulka 6.5.: Nastavení obtížnosti podle počtu odpovědí…….75

Tabulka 7.1.: Nastavení vazeb mezi otázkami a kapitolami v bázi znalostí pro zkoušení komplexních čísel. Pro otázky 1 – 20……..86

Tabulka 7.2.: Nastavení vazeb mezi otázkami a kapitolami v bázi znalostí pro zkoušení komplexních čísel. Pro otázky 21 – 40……86

Tabulka 7.3.: Výsledky zkoušení studentů z otázek 1 - 20. U každého studenta je zde uvedeno zda odpověděl dobře (**o**) nebo špatně(**x**), pro každou otázku. V případě správné odpovědi je uveden čas v [s]…….87

Tabulka 7.4.: Výsledky zkoušení studentů z otázek 21 - 40. U každého studenta je zde uvedeno zda odpověděl dobře (**o**) nebo špatně(**x**), pro každou otázku. V případě správné odpovědi je uveden čas v [s]…….88

Tabulka 7.5.: Výsledky zkoušení studentů rozdělené podle kapitol. Vzhledem k tomu, že jsem hranici pro úspěšné absolvování testu nastavil na 60 %, výsledky pod 60 % jsou označeny červeně, nad 60 % zeleně……89

1. ÚVOD

Prvním úkolem diplomové práce bylo seznámit se s problematikou expertních systémů a zpracovat literární rešerši ohledně expertních systémů. V této rešerši jsem se zaměřil na architekturu expertních systémů.

Další část práce obsahuje literární rešerši na téma reprezentace znalostí pro expertní systémy. Zde se zaměřuji hlavně na reprezentaci znalosti pomocí rámců. Popsal jsem zde čtyři způsoby reprezentace znalostí a ke každému jsem zpracoval vzorový příklad.

Hlavním cílem této diplomové práce je vytvoření a odladění expertního systému pro samostatné domácí studium s automatickým vyhodnocením úspěšnosti a doporučením pro další studium. Tento expertní systém má využívat rámcovou reprezentaci znalostí. Tato část práce obsahuje popis ovládání a všech funkcí tohoto expertního systému. Důkladně je zde popsán způsob ukládání znalostí pomocí rámců a všechny možnosti nastavení rozhodovacího mechanismu.

V posledním úkolem práce je popsat návrh a vytvoření jednoduché báze znalostí v popsaném expertním systému.

2. ZNALOSTNÍ SYSTÉMY

Znalostní systémy jsou programy umožňující řešit problémy produktivními postupy na základě poznatků, nebo-li znalostí, kterými jsou tyto systémy vybaveny. Znalostní systémy se staly důležitou podoblastí umělé inteligence a počítačových aplikací obecně.

Znalostní systémy jsou často používané v oblastech, kde nemáme k dispozici analytický nebo jinak formalizovatelný řešící postup. Jedná se tedy o problémy, kde nelze použít algoritmický postup. K řešení takových problémů se používají produktivní metody vyhledávaní a přerozdělování efektivních posloupností řešících operací. Pravděpodobnost nalezení takového postupu, který nalezne správný výsledek se zvyšuje s počtem poznatků (znalostí), které k řešeni lze použít. Pomocí znalostí a zkušeností se tyto systémy snaží vylučovat nevhodné postupy a zaměřit se na ty, které jsou pro danou problematiku vhodné. Z nich se pak vybírají postupy, které jsou nejlepší, nebo se k nejlepší variantě nejvíce blíží.

Příklady problémů, které se řeší pomocí znalostních systémů jsou z odborných problémů, ale i problémy, které jsou lidé schopni řešit na základě všeobecných poznatků v každodenním životě. K těmto problémům tedy patří například hlavolamy, úlohy elementární matematiky, psychologické testy, odvozování implicitních informací obsažených v určitém příběhu, seskupení faktů a řešení běžných životních situací (interpretace, plánování, monitorování, řízení).

Mezi znalostní systémy řadíme například inteligentní technické systémy (kognitivní roboti[1], systémy zpracování textu a deduktivní systémy, které zpracovávají báze údajů a rad).

[1] „Je kybernetický systém schopný autonomní interakce s reálným prostředím za účelem splnění stanoveného cíle“ [6]

3. EXPERTNÍ SYSTÉMY

Ze všech znalostních systémů jsou nejpoužívanější expertní systémy. Expertní systémy kvůli svým specifickým vlastnostem a v nich zahrnutých odborných poznatcích se považují za podtřídu systémů znalostních. Jejich specifičnost je dána tím, že používají znalosti a řešící postupy, které navrhli odborníci a to většinou experti ve své profesi. Expertní systémy se programují na řešení problémů, kterými se většinou pověřují odborníci. Většinou můžeme naprogramovat expertní systém vybavený znalostmi odborníků náročných oborů z různých oblastí lidské činnosti, ale problém nastává v případě, kde se musí použít tzv. „zdravý lidský rozum“ využívající všeobecné neodborné znalosti.

Inteligentní chování programů, které realizují expertní systém je tedy podmíněné jak odbornými, tak všeobecnými znalostmi z oblasti daného problému. Tyto znalosti jsou nejdůležitější složkou expertního systému.

Nezbytnou součástí expertního systému je také způsob vysvětlení a odůvodnění navržených řešících postupů. To může velice pomoci uživateli expertního systému posoudit úroveň expertízy a na základě toho se pak může rozhodnout navržené řešení přijmout, upravit ho a nebo ho odmítnout. Díky tomuto objasnění svého rozhodnutí není pro uživatele expertní systém pouze tzv. „černou skřínkou“.

3.1 POUŽITELNOST EXPERTNÍCH SYSTÉMŮ V PRAXI

Tato kapitola odpovídá postupně na základní otázky ohledně použití, návrhu a výroby expertních systémů.

3.1.1 Které kategorie problémů je vhodné řešit pomocí expertních systémů?

Pokud jde o odpověď na tuto otázku, tak by řešení mělo být uskutečněné úvahou v případě počítače odpovídajícími symbolickými manipulacemi. Problémy

vhodné na řešení expertními systémy patří podle [7], alespoň do jedné z následujících kategorii:

- Interpretace – Rozpoznání situace z údajů, které ji popisují.
- Predikce – Odvození očekávatelných důsledků dané situace.
- Diagnostika – Určení stavu (poruchy, poškození) systému z pozorovatelných (dostupných) projevů jeho chybného chování.
- Konstrukce – Výběr a sestavení objektů do určitého funkčního celku za daných ohraničujících podmínek.
- Plánování – Sestavení posloupnosti akcí za účelem dosažení daného cíle.
- Monitorování – Sledování a pozorování údajů odpovídajících určité situaci za účelem zjištění (a následného odstranění) odchylek od očekávané situace.
- Ladění a opravování – Výběr sestavení a uskutečnění posloupnosti akcí odstraňujících odchylky či chybové stavy.
- Učení – Diagnostika ladění a upravování studentových vědomostí.

Z dalšího úhlu pohledu je možné tyto kategorie problémů členit na analýzu a syntézu.

3.1.2 Kdy je vhodné použít expertní systém?

Odpověď na tuto otázku najdeme pomocí následujících vlastností:

- nelze-li problém dobře formálně popsat nebo strukturovat
- řešení není deterministické[2] je založené na produktivních postupech vyhledávání vhodné posloupnosti řešících kroků
- princip řešení nemá teoreticky dobré a ucelené podklady, použité znalosti nejsou formálně dobře vyjádřitelné
- používané údaje jsou neurčité, nepřesné, nespolehlivé a vzhledem na nedostupnost neúplné

[2] „Deterministické řešení je takové, které na stejný vstup (resp. na stejné výchozí podmínky) reaguje vždy stejně (tedy předvídatelně) a v každém jeho kroku je vždy jednoznačně definován i krok následující.“ [8]

Čím více daný problém odpovídá uvedeným skutečnostem, tím více přichází do úvahy použití expertního systému.

3.1.3 Kdy je účelné vytvořit expertní systém?

Vytvoření expertního systému je výhodné, když jeho používání přináší užitek, nahrazuje lidské úsilí, ulehčuje nebo ho doplňuje.

4. ARCHITEKTURA EXPERTNÍCH SYSTÉMŮ

Expertní systém je velké množství programů, jejíž činnost se opírá o specifické údajové struktury. V této kapitole budou popsány vzájemné vazby mezi jednotlivými moduly expertních systémů a jejich programovými celky.

4.1 ZÁKLADNÍ SLOŽKY EXPERTNÍCH SYSTÉMŮ

K sestavení nejjednoduššího expertního systému potřebujeme minimálně tři základní části: řídící mechanismus, báze znalostí a báze dat.

Jeden ze základních principů expertních systémů spočívá v oddělení symbolické reprezentace znalostí a báze dat od řídícího mechanismu (vzájemné oddělení báze znalostí a báze dat neplatí pro všechny expertní systémy).

Obrázek 4.1.: Základní složky expertních systémů.

4.1.1 Řídící mechanismus

Na rozhodovací mechanismus se lze dívat z více hledisek. V této kapitole budou popsána tři podstatná hlediska pohledu: vnější (uživatelský), vnitřní (funkční) a realizační pohled.

Z vnějšího pohledu na expertní systém jsou podstatné funkce rozhodovacího mechanismu, které uživatel vnímá při práci s expertním systémem. To je například deduktivní nebo induktivní usuzování[3], usuzování na základě asociování, kauzality[4],

[3] „Myšlenkový proces, v němž se z přijatých výroků dochází k novému, ještě nepřijatému výroku (deduktivní usuzování) nebo k zvětšení míry jistoty o jiném výroku, který již byl v určitém stupni přijat (induktivní usuzování)“ [9]
[4] „Kauzalita (z lat. causa, příčina) znamená příčinnost, vztah mezi příčinou a jejím následkem.“ [10]

generalizace[5], zahrnování, komplementárnosti, kontextu, analogie nebo odhad, heuristické usuzování[6], zaměření či odpoutání pozornosti, usuzování za přítomnosti nejistot a neurčitosti, zohlednění globálních a lokálních hledisek údajů svědčících v prospěch či neprospěch hypotézy.

Při vnitřním pohledu se musíme zaměřit na funkce rozhodovacího mechanismu, které souvisí s teoretickými principy jeho konstrukce. Jsou to například: prostředky z teorie grafů, síťové závislosti (kauzální, hierarchické), konceptuální grafy[7], syntaktická analýza, matematická lingvistika, matematická logika, kvalitativní modely, teorie výpočetní složitosti, teorie množin, teorie pravděpodobnosti.

Funkce rozhodovacího mechanismu jsou z realizačního pohledu sledované přes programovací jazyky a programovací prostředí. Základní funkce rozhodovacího mechanismu jsou realizovatelné dvěmi různými metodami nebo jejich kombinací. Nazývají se zpětný a přímý chod.

Zpětný chod řešení problémů spočívá v nacházení vhodného a efektivního způsobu dosahování určitého dopředu stanoveného cíle. Příkladem této metody je potvrzení nebo vyvrácení uvažované hypotézy: *nápoj obsahuje alkohol, pacient má neštovice, teče obvodem proud*. Zodpovězení dané otázky: *umožňuje daný snímač měřit s požadovanou přesností?* a nebo odvození konstrukce požadovaného zařízení: *sestrojení převodovky s požadovanými parametry, návrh snímače na požadovanou přesnost.*

Řešení problémů přímým chodem vyplývá z určitého objemu známých dat, které je potřebné interpretovat. Je potřebné odvodit co z nich vyplývá a k jakým důsledkům je možné na jejich základě dospět. Vychází se tedy z určitých údajů, které se v procesu průběžně doplňují podle určitého postupu.

---

[5] „Generalizace je proces výběru, zjednodušení a zevšeobecnění obsahu“ [11]

[6] „Heuristická metoda není metodou přísného důkazu, ale postupem pomáhajícím při hledání důkazů. Studuje skutečně se vyskytující případy objevů a vynálezů a pokouší se z nich odvodit obecné zákonitosti objevování a vynalézání, které by nebyly závislé na konkrétní úloze“ [12]

[7] „Konceptuální grafy jsou formální nástroj pro zaznamenání explicitní znalosti. Zachycují abstraktní,konceptuální strukturu jazykové podoby vyjádřené znalosti. Mohou být znázorněny graficky či zapsány v podobě jim odpovídajícího lineárního textového kódu.“ [13]

4.1.2 Báze znalostí

Báze znalostí je nevyhnutelnou částí každého expertního systému. Máme-li jeden rozhodovací mechanismus, můžeme k němu vytvořit více různých bází znalostí. Předpokladem tohoto je vyjádření (reprezentace) různých problémů způsobem, se kterým umí daný rozhodovací mechanismus pracovat.

Uvažovat o všeobecných charakteristikách reprezentace znalostí znamená ve vzájemných souvislostech uvažovat o:

- vyjadřovací účinnosti reprezentačních prostředků, to znamená o tom, co a do jaké míry je možné jimi reprezentovat
- odvozovací účinnosti prostředků, to znamená co a jak umožňují odvozovat
- výpočetní účinnosti (efektivitě) těchto prostředků, tedy o složitosti symbolických procedur, které jsou nimi podmiňované

Některé typy poznatků používané v expertních systémech:

- asociativní – odpovídají obvyklým, pozorovatelným či pravděpodobným, ne však zdůvodnitelným a ani nevyhnutelným vztahem mezi dvojicemi entit[8] (objektů)
- kauzální – odpovídají známé a zdůvodnitelné souvislosti mezi dvěma entitami, které jsou a nebo můžou být ve vztahu příčiny a následku
- taxonomické – odpovídají známým souvislostem mezi entitami, které vystihují vztah všeobecného k speciálnímu (generalizačně-speciální vztahy)
- kontextové – odpovídají různým podmínkám, za kterých se uplatňují určité souvislosti mezi entitami (vyjadřují například okolnosti, od kterých závisí, zda se souvislost uplatní a nebo neuplatní)
- prostorové – odpovídají prostorovým vztahům mezi entitami (například vztahem sousedství celků, nacházení se uvnitř nebo venku, poloha a orientace v prostoru, směr pohybu)

[8] „Entita v informatice zcela obecně může obsahovat jakoukoliv přesně definovanou množinu dat“ [14]

- časové – odpovídají takovým souvislostem mezi entitami, které na základě jejich výskytu a nebo jejich změn v určitých časových okamžicích a nebo intervalech podmiňují okamžité nebo časově následované výskyty či změny jiných entit, tím umožňují uvažovat o jevech minulých a budoucích
- modelové – odpovídají známým či předpokládaným dynamickým souvislostem s pravidla mezi větším počtem určitým způsobem převzatých entit, které se navzájem ovlivňují a tím určují správné řízení nimi vytvořeného systému

4.1.3 Báze dat

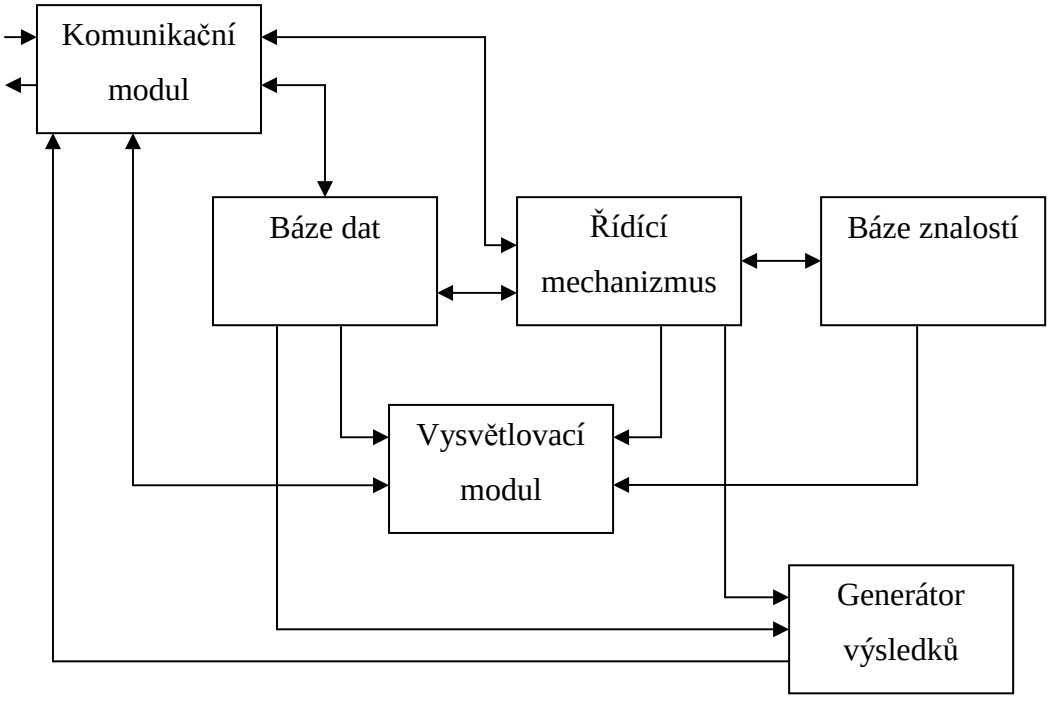
Další pasivní doplněk rozhodovacího mechanismu je báze dat (údajů, faktů). Báze dat je tvořena pasivními údajovými strukturami. Báze znalostí obsahuje symbolickou reprezentaci všeobecně platných poznatků a dané problémové oblasti. Báze dat uchovává symbolickou reprezentaci konkrétních dat souvisejících s právě řešeným problémem. Odpovídající data pak používá rozhodovací mechanismus v průběhu svojí činnosti, také je doplňuje, modifikuje a může je i rušit. Jednotlivé položky báze dat jsou podmíněny reprezentací báze znalostí a i funkcemi rozhodovacího mechanismu.

Jednotlivé položky báze dat mohou být tvořeny jednoduchými symbolickými útvary (například symbolem entity s přiřazenou hodnotou, seznamy symbolů entit se společnými vlastnostmi, n-ticemi uspořádaných symbolů s předdefinovaným významem daných jejich pozicí v uspořádání). Můžou to být i složité útvary (jako například pojmenované seznamy seznamů, pojmenované údajové celky s vnitřní strukturou), rekurzivně definované údajové položky a jiné.

Vztah mezi bází znalostí a bází dat je možné považovat za částečné zobrazení, při kterém položky báze dat vlastně konkretizují položky báze znalostí. Báze dat je na rozdíl od báze znalostí dynamická struktura údajů. Její obsah se v průběhu odvozování zpravidla značně mění. Často se v ní vytvářejí nové položky, ale dochází i k úpravám a rušení položek.

4.2 PŘÍDAVNÉ SLOŽKY EXPERTNÍHO SYSTÉMU

Expertní systémy se nemusí skládat pouze z tří základních částí (báze znalostí, báze dat, rozhodovací mechanismus) popsaných v předchozích kapitolách. Rozeznáváme další tři samostatné celky, které jsou součástí expertních systémů. Prvním je komunikační modul, který zabezpečuje komunikaci mezi expertním systémem a uživatelem. Dalším samostatným celkem je vysvětlovací modul. Vysvětluje a zdůvodňuje stav a průběh řešení. Posledním je generátor výsledků, který sestavuje průběžné výsledky do celku v požadovaném a srozumitelném tvaru. Na následujícím *obrázku 4.2.* jsou znázorněny vazby mezi základními a přídavnými složkami expertního systému.



Obrázek 4.2.: Základní a přídavné složky expertních systémů.

4.2.1 Komunikační modul

Pro toho, kdo potřebuje experta je velice důležitá možnost s ním komunikovat, stejně tak je tomu i pro expertní systémy. Funkce komunikačního

modulu spočívá v zabezpečení interakce (komunikace) mezi uživatelem a expertním systémem. K jeho hlavním funkcím patří:

- inicializace a ukončení činnosti expertního systému jako celku a také inicializace činností některých jeho modulů
- realizace dialogu v průběhu odvozování. Předkládání dotazů uživateli při shromažďování potřebných dat, načítání jeho odpovědí, kontrola jejich správnosti a vypisování chybových hlášek
- obsloužení povelů (příkazů a požadavků) uživatele

4.2.2 Vysvětlovací modul

Pokud máme možnost komunikovat s expertem, tak se od něj očekáváme, že svoje rozhodnutí dokáže vysvětlit a zdůvodnit. Funkce vysvětlovacího modulu má za úkol právě vysvětlování a zdůvodňování stavu a průběhu řešení problému, jeho jednotlivých řešících kroků a taky dosáhnutých výsledků. Není však podmínkou, aby byl vysvětlovací modul v každém expertním systému. V některých expertních systémech je dokonce nežádoucí.

V nejjednodušším provedení by měl vysvětlovací modul odpovídat na dva nejčastější typy otázek:

- Proč?
- Jak?

4.2.3 Generátor výsledků

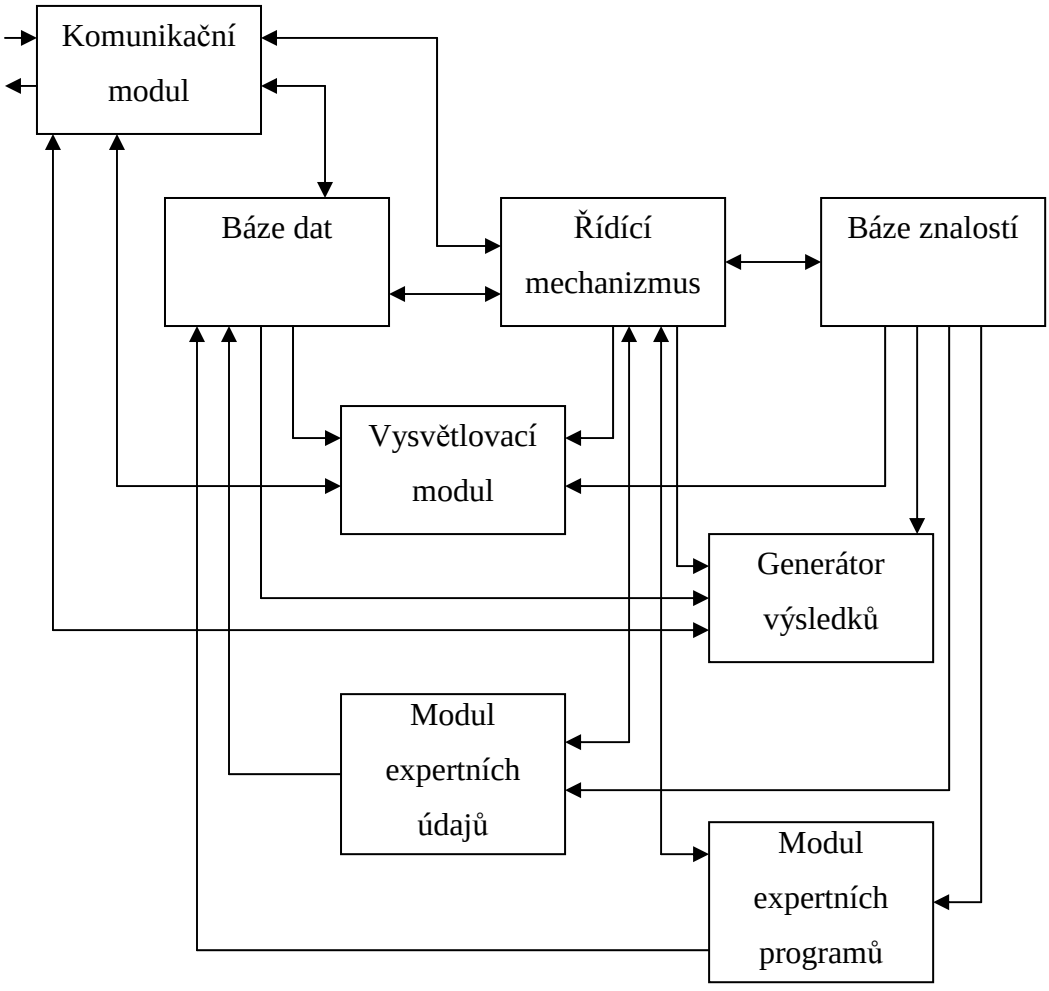
Má za úkol především sestavit částečné výsledky do integrálního a odvozeného celku bez nadbytečných informací v požadovaném tvaru a srozumitelné formě. Zabezpečuje tedy vhodnou prezentaci dosažených výsledků.

Protože je výhodné, když může generátor výsledků pracovat paralelně s řešením problému je vhodné realizovat generátor výsledků jako samostatnou složku expertního systému. Z dat, které byly zjištěny při dotazovaní nebo odvozování, začlení generátor výsledků do předkládané zprávy pouze ty, které bezprostředně

souvisí z dosaženým výsledkem. Všechny tyto data uspořádá do obsahově souvisejících celků a nepoužije ty, které jsou nadbytečné.

4.3 DOPLŇKOVÉ SLOŽKY EXPERTNÍHO SYSTÉMU

V expertních systémech se dále vyskytují další doplňkové složky a to modul expertních údajů a modul expertních programů. Začlenění doplňkových složek do architektury expertního systému je naznačeno následujícím *obrázku 4.3.*



Obrázek 4.3.: Základní, přídavné a doplňkové složky expertních systémů.Základní a přídavné složky expertních systémů.

4.3.1 Modul expertních údajů

V této kapitole nejsou na mysli údaje, které má expertní systém předem připravené. Tyto údaje nejsou obsahem žádné z údajových struktur expertního systému. Když expertní systém pracuje s externími údaji, tak rozhodovací mechanismus vyžaduje určitý fakt na probíhající odvozování. Rozhodovací mechanismus se tento fakt může pokusit získat. Prostředkem pro získání požadovaného faktu může být modul expertních údajů.

Základním úkolem modulu expertních údajů je sledování situací ve kterých rozhodovací mechanismus požaduje údaje nebo-li fakty. Když nastane tato situace, modul expertních údajů přebírá řízení a zabezpečuje prohledávání expertních údajů. V případě, že nalezne požadovaný fakt, umožní vložit ho do báze dat a předává řízení zpět rozhodovacímu mechanismu. Pokud se stane, že modul požadovaný fakt nenajde, musí se tato situace ošetřit.

4.3.2 Modul expertních programů

V této kapitole budou popsány interakce expertního systému s programy ze svého okolí, to je interakce s expertními programy. Vzájemná interakce může být inicializována z obou stran, jak expertním systémem, tak i expertním programem.

Expertní systém ji může vyvolat, když požadovaný fakt může získat aktivací procedury:

- zabezpečující prohledávání vymezených registrů, do kterých se vkládají aktuální snímané údaje
- zabezpečující naplnění registrů požadovanými údaji
- zabezpečující vykonání určité části řešení problémů reproduktivním postupem na základě deterministického algoritmu.

Interakci může vyvolat expertní program aktivující expertní systém, když například:

- interpretace údajů, které zpracovává, vyžaduje produktivní řešící postup opírající se o znalosti experta

- je potřebné rozhodnout na základě znalostí, který z dostupných programů je vzhledem na daná kritéria nejvhodnější pro další zpracování daných údajů (používá se, když jde o volbu vhodné metody)

4.4 PROSTŘEDKY VYTVÁŘENÍ A UDRŽOVÁNÍ BÁZE ZNALOSTÍ

Bázi znalostí vytvářejí odborníci z dané problémové oblasti a nejčastěji s odborníkem, který se vyzná v problémech kolem expertních systémů. Vytvoření báze znalostí je časově náročné, proto se tvůrci prázdných expertních systémů snaží poskytnout tvůrcům bází znalostí co nejúčinnější programové prostředky na přenášení znalostí expertů do počítače a na jejich účinnou reprezentaci. Tyto prostředky se tvoří programovými moduly tvořícími podpůrné složky prázdného expertního systému. Uživatel expertního systému s již vytvořenou bází znalostí s nimi nepřijde do styku. Tyto moduly mohou být v nejjednodušším případě jednoduché textové editory. V složitějších případech může jít o poměrně komplexní systémy programů, které dokáží velice napomáhat tvůrcům báze znalostí. Podpůrný systém může být programově značně náročný, proto je účelné ho rozčlenit na jednotlivé komponenty, které jsou popsány v následujících kapitolách.

4.4.1 Modul definice báze znalostí

Modul definice báze znalostí slouží k naplnění a doplnění báze znalostí strukturami symbolické reprezentace znalostí. Jde tedy o prostředek, kterým znalostní inženýr nebo přímo expert naplňuje bázi znalostí.

4.4.2 Modul modifikace báze znalostí

Tento modul slouží pro možnost úprav v již vytvořené bázi znalostí. Poskytuje interaktivní prostředky na upravování a rušení existujících položek v bázi znalostí. Modul modifikace báze znalostí velice souvisí s modulem definice báze znalostí.

4.4.3 Modul kontroly báze znalostí

Modul kontroly báze znalostí je prostředkem odhalování chyb vznikajících při vytváření tedy definici a modifikaci báze znalostí.

4.4.4 Modul prohlížení báze znalostí

Ani v případě, že se povede odstranit všechny chyby uvedenými prostředky není zaručeno, že expertní systém na základě dané báze znalostí bude řešit problémy, dle představ tvůrců báze znalostí. Pokud jde o jednoduchou bázi znalostí, pak není složité přezkoumat celý její obsah. Dostaneme-li se však do situace, kde je báze znalostí tvořena rozsáhlými reprezentačními strukturami, tak je výhodné, když má tvůrce báze znalostí k dispozici prostředky umožňující „zviditelnit“ obsah báze znalostí. Jde tedy o to získat přehled o reprezentovaných znalostech. Tvůrce tak získá podklady pro zkoumání příčin nežádoucích činností expertního systému. Programy sloužící pro tento účel tvoří modul prohlížení báze znalostí.

5. REPREZENTACE ZNALOSTÍ

Úvahy člověka a ani výpočet počítače si nelze představit bez toho, aniž by objekty, kterých se výpočty týkají, nebyly reprezentovány jinými objekty s kterými dokáže člověk nebo počítač pracovat.

Důležitost reprezentace znalostí zde bude ukázána na jednoduchém příkladu. Úkolem je vypočítat jednu polovinu a jednu pětinu. Postup získání výsledku bude záviset na tom, jak příslušné hodnoty reprezentujeme. První možností je *0,5* a *0,2* a druhou možností je *1/2* a *1/5*. K tomuto jednoduchému příkladu si můžeme položit hned několik otázek. Která z uvedených reprezentací je výhodnější? Existuje z daných hledisek nejvýhodnější reprezentace? Jestliže ano, jak ji najdeme? A tak lze pokračovat dále.

Na znalosti je možné se dívat z různých pohledů. Umělá inteligence je zaměřena pouze na ty, které lze počítačově vhodně reprezentovat. Těmi jsou například: produkční pravidla, sémantické sítě, rámce, znalostní jazyky a konceptuální grafy.

Reprezentace znalostí je nejdůležitější pro expertní systémy ze dvou hlavních důvodů. Prvním je, že expertní systémy jsou navrhovány pro jistý typ reprezentace znalostí a to podle pravidel nebo logiky. Dalším důvodem je, že reprezentace znalostí ovlivňuje vývoj, rychlost, výkonnost a údržbu expertních systémů.

Význam slova „znalost“ je jedno ze slov, které lze jen těžko definovat. Dalším takovým slovem je například slovo „láska“. Další slova jako třeba data, informace, fakta bývají často užívány vzájemně se znalostmi. Mimo filozofické druhy znalostí, které popsaly Aristoteles, Platon, Hume, Kant, a jiní, existují typy znalostí zvané *a priori* a *a posteriori*.

Termín *a priori* pochází z latiny. Jeho příkladem může být prohlášení: *všechno má svůj důvod, všechny trojúhelníky v rovině mají součet vnitřních úhlů 180°*. Tyto příklady *a priori* znalostí jsou považovány za univerzální pravidla. Nemohou být odmítnuty bez rozporu. Dalšími *a priori* znalostmi mohou být například logické prohlášení nebo matematické zákony.

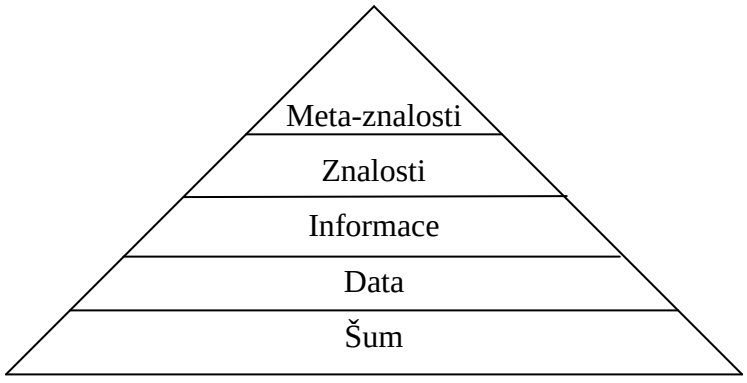
Opak *a priori* znalostí je znalost odvozená ze smyslů, nebo-li *a posteriori* znalosti. Pravdivost nebo nepravdivost *a posteriori* znalostí, může být upravována užíváním smyslů a zkušenostmi. Příkladem může být prohlášení: *světlo je zelené*. Získané zkušenosti nemusí být spolehlivé. *A posteriori* znalosti mohou být tedy odmítány na základě nových znalostí, bez nutnosti rozporu v jednotlivých prohlášeních. Příkladem tohoto může být znalost, že: *osoba má hnědé oči* a všichni budou věřit tomu, že zmíněný člověk má hnědé oči. Tato osoba se později přizná, že nosí hnědé kontaktní čočky a odhalí své modré oči. Znalost se pak přepracuje na novou, a to: *osoba má modré oči*.

Znalosti mohou být dále klasifikovány do skupin:

- procedurální znalosti – postup práce, protokol, způsob práce
- deklarativní znalosti – prohlášení, vyhlášení, deklarace
- nevyslovené znalosti – implicitní, nevyslovené

Procedurální znalosti často odkazují na znalost toho, jak se něco dělá. Příkladem této znalosti může být *znalost toho, jak se vaří voda*.

Deklarativní znalost odkazuje na znalost něčeho, co může být pravda nebo není pravda. To se týká deklarativních prohlášení jako například: *nedávej prst do horké vody*.



Obrázek 5.1.: Hierarchie znalostí.

Nevyslovené znalosti bývají, také někdy nazývány jako nevědomé znalosti, protože je nemůžeme vyjádřit jazykem. Příkladem této znalosti může být: *znalost způsobu, jak se pohybuje ruka*. Hrubým odhadem toho, jak se ruka pohybuje, může být, že se ruka pohybuje upjatě nebo uvolněně.

Znalost je část hierarchie nakreslené na *obrázku 5.1*. Na tomto obrázku je dole šum. Šum jsou položky, které nejsou zajímavé. Jsou to nezřetelná data. Vyšší úroveň jsou data obsahující položky, které mohou být zajímavé. Informace nebo data postupu řešení jsou na třetí úrovni. Další úrovní jsou znalosti, které reprezentují velmi speciální informace. Meta-znalosti jsou znalosti o znalostech a odborné znalosti. Expertní systém může být navržený se znalostmi o několika rozdílných oborech. Meta-znalosti mohou určovat, které znalosti báze by bylo vhodné použít.

5.1 REPREZENTACE ZNALOSTÍ PROSTŘEDKY PREDIKÁTOVÉ LOGIKY

Jedná se o teoreticky nejlépe poznaný prostředek reprezentace znalostí. Základní úlohou logiky je poskytnutí prostředků vyjadřování poznatků, které umožňují spolehlivě prezentovat neprotichůdnost tvrzení. Logika se tedy nesoustředí v první řadě na vztahy mezi tvrzeními. Například, když nějaké předpoklady implikují jistý důsledek, logika nás přivede k jeho akceptování. Jestliže je však nějakými předpoklady implikovaný důsledek z jistých nelogických příčin pro nás nepřijatelný, logika nám poradí jen to, abychom zamítly alespoň jeden z předpokladů.

Dále bude rozebrána výroková logika 1.řádu. Tato logika umožňuje vyjadřovat fakty prostřednictvím několika struktur.

Strukturálně nejjednodušší jsou atomy. Atom se skládá z výrokového symbolu, za kterým následuje v závorkách seznam termů. Termy mohou být konstanty, proměnné a nebo funkce závislé na konstantách nebo proměnných. Atomy mohou nabývat dvou logických hodnot *pravda* a *nepravda* tedy *1* nebo *0*. Z atomů můžeme vytvářet složitější logické struktury (formule) pomocí logických spojek: konjunkce, disjunkce, negace, implikace a ekvivalence.

Ve formulích se mohou také vyskytovat klasifikátory, které v nich charakterizují postavení proměnných. Jde o všeobecný a existenční klasifikátor.

Základní výraz výrokové logiky je formule a základními prvky jsou:

- logické hodnoty – *pravda/nepravda, T/F, true/false, 1/0*
- logické spojky – negace(‘), disjunkce($\vee$, *nebo*), konjunkce($\wedge$, *a*), implikace($\Rightarrow$, *když…potom*), ekvivalence($\Leftrightarrow$, *tehdy a jen tehdy*), rovnost(=).
- logické kvantifikátory – univerzální($\forall$), existenční($\exists$)
- funkční symboly – malá písmena a řetězce (*f, g, h, sečti*)
- konstanty a proměnné – malá písmena (*a, b, c*)
- predikátové symboly – velká písmena a řetězce (*A, B, C*)

Tabulka 5.1.: Pravdivostní tabulka výrokových formulí.

p	q	$p‘$	$p \vee q$	$p \wedge q$	$p \Rightarrow q$	$p \Leftrightarrow q$
0	0	1	0	0	1	1
0	1	1	1	0	1	0
1	0	0	1	0	0	0
1	1	0	1	1	1	1

Použití predikátové logiky bude ukázáno na jednoduchých příkladech. Pokud si zavedeme, že *p* znamená „je chladno“ a *q* znamená „prší“, tak pomocí predikátové logiky můžeme napsat:

p‘ …..… není chladno

$p \vee q$….. je chladno nebo prší

$p \wedge q$….. je chladno a prší

$p \wedge q$’….. je chladno a neprší

5.2 REPREZENTACE ZNALOSTÍ PRODUKČNÍMI SYSTÉMY

Z dějin jsou známé i další názvy „Postove systémy“[9] a nebo „přepisovací systémy“.

Používají se v těch oblastech umělé inteligence, které jsou zaměřeny na poznávání lidského myšlení. Základním pojmem produkčních systémů je produkční pravidlo. Z hlediska struktury lze produkční pravidlo rozčlenit na dvě základní části, a to na předpokládanou (situační) a důsledkovou (akční). Pravidlo potom můžeme schématicky znázornit jako strukturu následujícího tvaru:

JESTLIŽE (je splněná) předpokládaná část TAK (platí) důsledková část

Následně zde bude popsán rozdíl mezi pravidly ve významu v jakém se používají v produkčních systémech a pravidly, kterých se používá ve výrokové logice. Ve výrokové logice jde o vyjádření vztahu mezi předpokladem a důsledkem. Výroková logika je pravdivá bez ohledu na to zda je předpoklad pravdivý nebo není. U produkčních pravidel v expertních systémech je důsledek platný pouze tehdy, když je splněná předpokládaná část pravidla.

Následující příklad uvádí nekategorické produkční pravidlo:

JESTLIŽE je někdo vysoký TAK (ASI) má velkou hmotnost

Nekategorická povaha tohoto pravidla je vyjádřená slovem „*ASI*“. Mechanismy testování předpokládaných částí jsou u produkčních pravidel při různých expertních systémech realizovány různě. Vždy jsou, ale založené na dvou základních procedurách, a to prohledávání a porovnávání.

Formální symbolika pro definici produktivity je „Backus-Naur“ z BNF[10] [15]. Tato symbolika je metajazykem[11] pro definování syntaxe jazyků. Metajazyky slouží

[9] Postove systémy se jmenují podle člověka, který je první použil Emil Post
[10] British National Formulary
[11] Význam „meta“ znamená nad a také metajazyky jsou nad ostatními jazyky

jako jazyky pro popis jazyků. Je mnoho typů jazyků: přírodní jazyky, logické jazyky, matematické jazyky a počítačové (programovací) jazyky.

5.2.1 Reprezentace znalostí produkčními systémy - Příklad č.1.: gramatika anglického jazyka

Reprezentaci znalostí produkčními systémy budou vysvětleny na: BNF symbolice pro základní pravidlo anglického jazyka, jehož věty obsahují podstatné jméno a sloveso následované interpunkcí:

<sentence> ::= <subject> <verb> <end-mark>

kde <> a ::= jsou symboly metajazyka a ne jazykem přesně vymezeným. Symbol ::= znamená „je definován jako“ a zároveň je to BNF ekvivalent šipky „→“.

Výrazy uvnitř v závorkách se nazývají *nonterminals* symbol. Je to proměnná, která reprezentuje jiné výrazy. Ostatní výrazy mohou být, kterékoliv z *nonterminals* nebo *terminals*. *Terminals* nemohou být nahrazeny žádnými jinými a tak jsou konstantní. *Nonterminals <sentence>* je speciální, protože je to „start symbol“, z kterého jsou ostatní symboly definovány. V definici programovacích jazyků se start symbol obvykle jmenuje *<program>*.

Produkční pravidlo pro jednoduchou anglickou větu:

<sentence> →<subject> <verb> <end-mark>

oznamuje, že věta je poskládána z *subject*, *verb* a *end-mark*. Následující pravidlo dokončuje *nonterminals* specifikaci jejich možných výrazů. Kde”|” znamená “or” v metajazyku.

<subject> → I | You | We
<verb> → left | came

<end-mark> → . | ? | !

Některé z možných výsledků:

I left.
I left?
I left!
You left.
You left?
You left!

Takto bych mohl pokračovat do vyčerpání všech možností podle uvedeného pravidla.

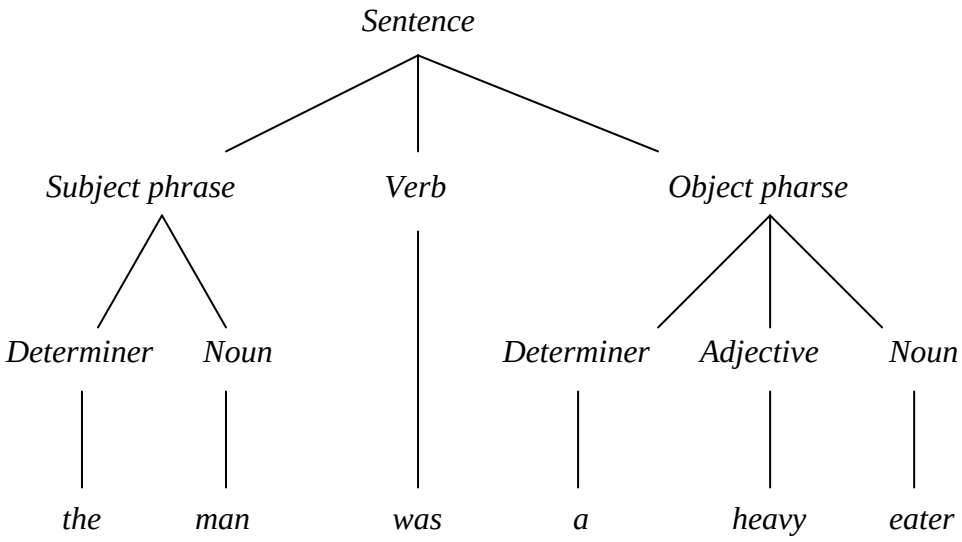
Nastavení *terminálů* se nazývá „řetězec“ jazyka. Jestliže řetězec může být odvozen ze start symbolu, nahrazením *nonterminalem* jejich definičního pravidla, pak je řetězec nazýván platná věta. Například „WeWe“ nebo „leftcamecame“ jsou všechno platné řetězce jazyka, ale nejsou to platné věty. Gramatika je kompletní nastavení produkčních pravidel tak, že definují jazyk, aby nebyl dvojsmyslný. Ačkoliv předešlá pravidla definují gramatiku, je to velmi omezené, protože je málo možných výsledných kombinací. Dalším příkladem může být více propracovaná gramatika:

<sentence> → <subject phrase> <verb> <object phrase>
<subject phrase> → <determiner> <noun>
<object phrase> → <determiner> <adjective> <noun>
<determiner> → a | an | the | this | these | those
<noun> → man | eater
<verb> → is | was
<adjective> → dessert | heavy

Užíváním této gramatiky mohou být generovány například tyto věty:

the man was a dessert eater.
an eater was the heavy

Syntaktický strom nebo-li odvozený strom je gramatika reprezentace věty rozložení uvnitř všech *terminálů* a *nonterminálů* užívaných k odvození věty. *Obrázek 5.2.* znázorňuje syntaktický strom pro větu *"the man was a heavy eater"*. Vezmu-li další podobnou větu: *"man was a heavy eater"* nelze ji považovat za platnou, protože chybí stanovení v <*subject phrase*>. Překladač vytváří syntaktický strom, aby mohl rozhodnout, zda výsledek odpovídá platné syntaxi jazyka.



Obrázek 5.2.: Syntaktický strom.

Strom na *obrázku 5.2* ukazuje, že věta *"the man was a heavy eater."* může být odvozena ze start symbolu použitím vhodných produkcí. Kroky v tomto procesu jsou vystaveny dvojitou šipkou =>, to znamená, že použití ukázané produkce bude:

<*sentence*> => <*subject phrase*> <*verb*> <*object phrase*>
<*subject phrase*> =><*determiner*> <*noun*>
<*determiner*>=> *the*

<noun> => man

<verb> => was

<object phrase> => <determiner> <adjective> <noun>

<determiner> => a

<adjective>=> heavy

<noun> => eater

Alternativní způsob užívání produkce je generování platných vět a nahrazování všech vhodných *terminálů nonterminály*. Samozřejmě ne všechny produkce, tak jako *"the man was a heavy eater"* dávají smysl.

5.3 REPREZENTACE ZNALOSTÍ SÉMANTICKÝMI SÍTĚMI

V sémantických sítích jsou znalosti reprezentovány pomocí objektů (entit) a binárních nebo víceargumentových relací mezi nimi. Víceargumentovou relaci lze vždy převést na relaci binární. Sémantické sítě můžeme kreslit do grafů, kde každý uzel grafu odpovídá objektu a každá hrana odpovídá binární relaci. Ve srovnání s rámci z kapitoly 5.4. odpovídají hrany sémantických sítí jménům položek u rámců a uzlům odpovídají hodnoty položek.

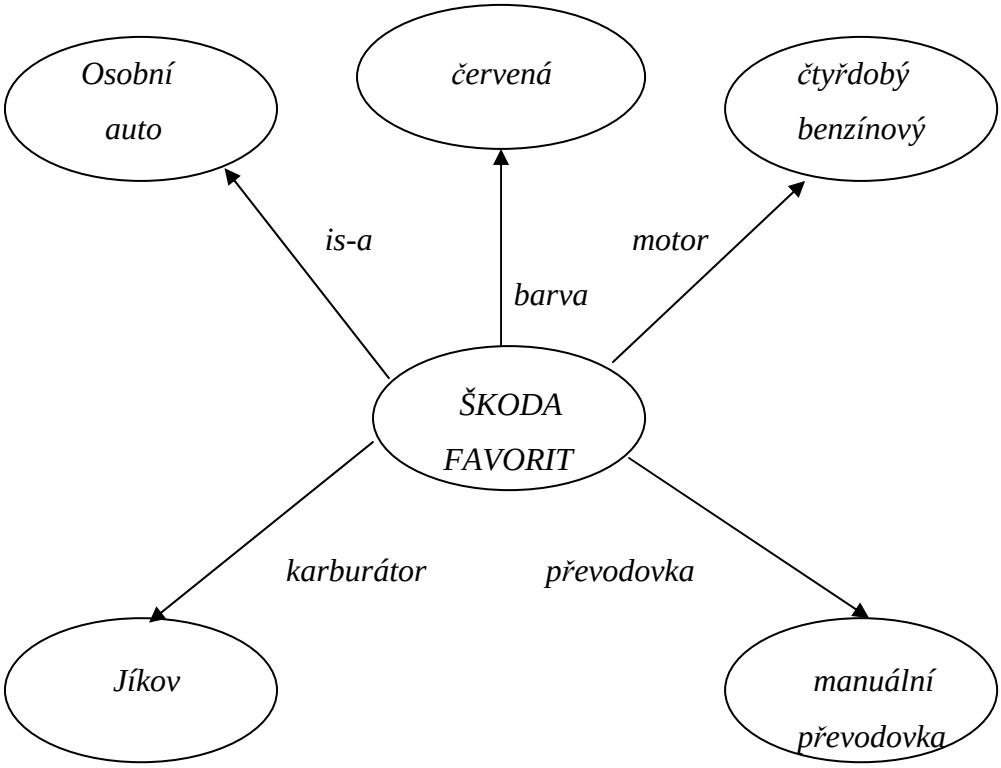
V sémantických sítích lze vyjadřovat vztahy množinové inkluze[12] a příslušnosti v množině. Můžeme zde jednoduše reprezentovat jedinečné i obecné pojmy. Vztahy množinové inkluze a příslušnosti ke třídě umožňují efektivně reprezentovat hierarchické uspořádání objektů. Pak lze přímočaře provádět odvozování specializací[13] či generalizací[14]. Využívá se zde tzv. „is-a hierarchie". V jedné sémantické síti mohou být znalosti asociovány z mnoha různých hledisek. Díky tomu můžeme vytvářet rozsáhlé komplexy, ve kterých můžeme efektivně vyhledávat důležité informace.

[12] „Vztah mezi třídami nebo množinami vyjadřovanými slovy, je podmnožinou" [16]
[13] Informace se v hierarchii přenášejí od obecnějších typů k speciálnějším
[14] Informace se v hierarchii přenášejí od speciálních typů k obecným

Tím, že údaje ukládáme na co nejobecnější úrovni, dosahujeme velmi úsporné reprezentace a rychlého vyhledávání informací. Objekty vzniklé specializací mohou dědit vlastnosti od objektů obecnějších. Nové údaje můžeme přidat zavedením uzlů a relací. Jednoduchá reprezentace sémantických sítí spolu s dědičností pomocí hierarchií, však může způsobit problémy. Lze tak odvodit rozporuplné informace. Sémantické sítě nedovolují efektivně reprezentovat předpokládané (default) vlastnosti. To umožňuje pouze reprezentace pomocí rámců, která bude popsaná v kapitole 5.4.

5.3.1 Reprezentace znalostí sémantickou sítí - Příklad č.1.: Škoda Favorit

Jak už bylo uvedeno, na začátku kapitoly v sémantických sítích, odpovídají hranám jména položek rámců a uzlům hodnoty položek rámců. Pro srovnání bude uvedena reprezentace znalostí sémantickou sítí na stejném příkladu automobilu Škoda Favorit, jako později u prvního příkladu na rámce.



Obrázek 5.3.: Příklad reprezentace automobilu Škoda Favorit sémantickou sítí.

5.4 RÁMCOVÁ REPREZENTACE ZNALOSTÍ

Prostředky sloužící pro reprezentaci znalostí uvedené v minulých kapitolách, vždy zvýrazňovaly pouze jediný ze třech atributů znalostí. Byly to buď vyjádřitelnost, začlenitelnost nebo použitelnost. To vedlo k vytvoření prostředku, který bude slučovat všechny tyto tři atributy. Tento prostředek reprezentace znalostí se nazývá rámcový a slučuje pozitivní vlastnosti deklarativních, asociativních a procedurálních prostředků reprezentace znalostí.

Rámce jsou takové struktury údajů, do kterých je možné ukládat znalosti deklarativní, asociativní i procedurální povahy. Díky asociativní složce reprezentace znalostí můžeme z rámců vytvářet explicitní provázané systémy rámcových sítí. Jak již vyplývá z toho, že rámce jsou struktury údajů, můžeme rámce jednoduše naprogramovat pomocí objektově orientovaného programování.

Rámce umožňují reprezentaci některých znalostí, na které nejsou předcházející prostředky přizpůsobeny. Jde o reprezentaci takových znalostí, které v případě, že k řešení problémů neznáme všechny potřebné skutečnosti, můžeme je nahradit pouze podmíněnými nebo dočasně platnými.

Díky výhodám oproti ostatním způsobům reprezentace znalostí se rámce často používají v rozsáhlých expertních systémech.

5.4.1 Terminologie rámcové reprezentace znalostí

Rámce se skládají z položek, pomocí nichž jsou popsány jednotlivé vlastnosti entit (objektů). Tyto položky v průběhu používání nabývají konkrétních hodnot. Každá z položek se skládá ze jména například: *motor* a z hodnoty, kterou nabývá například: *benzínový*. Jednotlivé položky můžeme dále členit, a tak zaznamenávat další vlastnosti popisovaného objektu. Položky se tedy dále dělí na fasety. V tomto případě se původní hodnota položky stává pouze jednou z faset. Její jméno je hodnota položky a do této fasety uložíme příslušný údaj. Rámcová reprezentace znalostí používá pro tyto údaje speciální terminologii. Než začnu jednotlivé pojmy používat, nejprve vysvětlím, co tyto pojmy znamenají.

V následujících odstavcích budou vysvětleny pojmy: slot (položka), faseta (pod slot), value (hodnota), default (standart), range (rozsah), démon, if-added (jestliže se přidá), if-needed (jestliže je potřeba) a if-deleted (jestliže se smaže).

Vzhledem k tomu, že rámce organizují znalosti v slotech, je základním pojmem právě „**slot**“. Všechny znalosti v rámcích jsou rozděleny do slotů. Slot může popisovat deklarativní znalosti. Příkladem deklarativní znalosti může být třeba *barva auta - červená*, *typ motoru - benzínový* a nebo *typ převodovky - manuální*. Ve slotu může být uložena, také postupová znalost (příkladem postupové znalosti je „aktivace určitého pravidla, jestliže hodnota překračuje danou úroveň“). Na reálném příkladu si lze aktivaci pravidla představit na rozsvícení kontrolky paliva v momentě, kdy je v nádrži málo benzínu.

Každý slot v rámci obsahuje jednu nebo více faset. „**Faseta**“ se také někdy nazývá podslot. Příklad fasety si lze představit u rámce *byt*, který obsahuje slot *místnosti*. *Místnosti* pak obsahují fasety: *kuchyň, jídelna, obývák, ložnice, pokoj, předsíň, koupelna, WC*. Dalším slotem rámce *byt* je majitel. Slot *majitel* již obsahuje pouze jednu fasetu: *Jan Novák*.

Sloty a fasety mohou obsahovat mnoho forem a to: values (hodnota), default (standart), range (rozsah), if added (jestliže se přidá) nebo if needed (jestliže je potřeba). Sloty mohou také obsahovat další rámce, pravidla, sémantickou síť a nebo úplně jiný typ informace.

První formu faset, která již byla použita, ale nebylo napsáno, že se jedná o „**values**“ (hodnota). Příkladem values může být u rámce automobil slot *barva*, který obsahuje jednu z hodnot: *červená*, *bílá*, *černá* …

Další formou uložení hodnoty do položky rámce je „**default**“. Jedná se o standardní hodnotu, kterou položka nabývá ve většině případů. Jednoduchým příkladem může být počet kol u automobilu. Pokud se tato hodnota nepřepíše jinou bude se u všech automobilů uvádět, že mají 4 kola.

V položce rámce může být uložen také „**range**“ (rozsah). Rozsah říká, jaký druh informace se může v slotu objevit. Rozsah nám může omezit hodnoty například na celá čísla. Příklad použití, kdy potřebujeme omezit položku na celá čísla může být počet pasažérů v dopravním prostředku. Další příklad omezení může být třeba na

čísla s přesností na jedno desetinné místo. Další reálný příklad ukážu na palivové nádrži. Pokud víme, že se do nádrže vejde maximálně 60 litrů benzínu, pak obsah palivové nádrže bude mít vždy hodnotu v rozsahu od 0 do 60 litrů paliva, proto tuto položku můžeme omezit na čísla od 0 do 60.

Každá faseta může obsahovat jeden nebo několik démonů. **Démon** je procedura, která se aktivuje nastanou-li určité události. Nejčastěji používaní démoni jsou „když je potřeba“, „když se přidá“ a „když se smaže“.

Další možností fasety je „**if-added**“. Jedná se o jednoho z démonů „když se přidá“. Fasety if-added obsahují postupové informace. Tyto fasety se používají k vyvolání požadované akce, jestliže se do slotu přidá hodnota (případně se akce provede, jestliže se tato hodnota změní na jinou).

Podobnou procedurou je „**if-needed**“. Jedná se o situaci, kdy slot neobsahuje žádnou hodnotu, ale tato hodnota je pro další postup potřeba. V tomto případě se spustí výpočet této hodnoty.

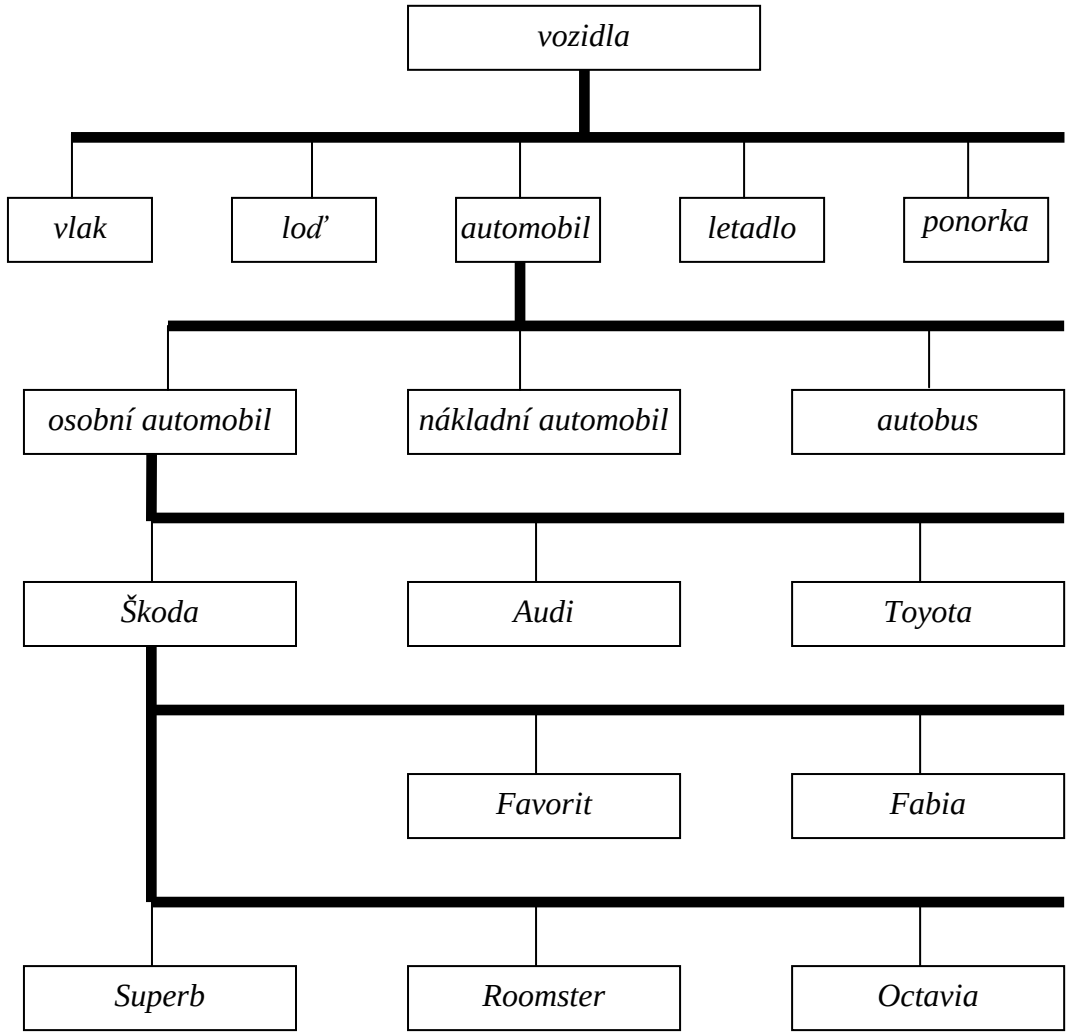
Další procedurou, která funguje na stejném principu je „**if-deleted**“. Procedura if-deleted se aktivuje v případě, kdy se určitá hodnota v slotu vymaže.

5.4.2 Hierarchie rámců

Co se týče hierarchie jsou rámce velice přehledný a na první pohled jasný systém. Rámce využívají pro svoji činnost hierarchii zvanou v anglické literatuře „taxonomic hierarchi“[15]. Do češtiny tento pojem lze přeložit jako systematická hierarchie.

Hierarchie bude vysvětlena na příkladu reprezentace znalostí *vozidla*. Příklad rámce *vozidla* je na *obrázku 5.4*. Na první úrovni je pouze rámec *vozidlo*. Další úroveň obsahuje rámce *vlak, loď, automobil, letadlo* a *ponorka*. Určitě by se, ale našlo spousta dalších dopravních prostředků od jízdního kola, až po méně tradiční vozidla jako je například kůň. Každý z uvedených rámců v druhé vrstvě by se dále dělil na další rámce. Na obrázku je uvedeno pouze rozdělení automobilů. Uvedené rámce jsou *osobní automobil*, *nákladní automobil* a *autobus*. Jistě si každý dovede

představit rozsáhlost příkladu a obrázku, kdyby zde byly uvedeny všechna rozdělení *vozidel*. Proto příklad pokračuje pouze v rozdělení jedno rámce a to rámce *osobní automobil*. Osobní automobily lze dělit podle výrobců. Proto další vrstva hierarchie bude obsahovat několik výrobců osobních automobilů: *škoda*, *toyota* a *audi*.



Obrázek 5.4.: Ukázka hierarchie rámcové reprezentace znalostí na příkladu reprezentace znalostí vozidla.

Podle příkladu uvedeného na *obrázku 5.4.* si lze představit jak by vypadala hierarchie rámce *bydlení*. V druhé vrstvě by byl například *rodinný dům* a *panelák*.

[15] Synonyma pro „taxonomic hierarchi“ jsou: is-a hierarchy, class and subclass hierarchi, AKO (a kind of) hierarchi a nebo inheritance hierarchi. [20]

V další vrstvě do paneláku patří rámec *byt*. Další popis rámce *byt* bude podrobně rozebrán později v kapitole 5.3.8.

5.4.3 Propojování rámců pomocí položky „*is-a*“

V minulé kapitole bylo uvedeno jak funguje hierarchie rámců. K tomu, aby jednotlivé rámce „věděli“ o tom, že k sobě patří, potřebují systém, který jim toto „řekne“.

K propojování rámců slouží speciální položka “*is-a*“. Do češtiny lze přeložit jako „*je ….*“ nebo „*patří do …*“. Pro uvedení praktického použití položky *is-a* se vrátíme se k příkladu rámce *vozidla*, na kterém byla vysvětlena hierarchie rámců. Příklad je na *obrázku 5.4.*

Na *obrázku 5.4.* je vidět, jak jsou jednotlivé rámce rozděleny do vrstev. V první vrstvě je pouze rámec *vozidla* – ten položku *is-a* nepotřebuje, protože je na vrcholku pomyslné pyramidy a nemá žádný prvek do kterého patří. V další vrstvě jsou rámce: *vlak*, *loď*, *automobil*, *letadlo* a *ponorka*. Protože jsou přímo napojeny na rámec *vozidla*, každý rámec z druhé vrstvy hierarchické struktury bude obsahovat položku *is-a* a její hodnota bude *vozidlo*. V třetí vrstvě jsou rámce *osobní automobil*, *nákladní automobil* a *autobus*. Analogicky jako v předchozím případě budou tyto tři rámce obsahovat položku *is-a* s hodnotou *automobil*.

5.4.4 Dědičnost vlastností rámců

U rámců se díky jejich hierarchické struktuře může využívat dědictví vlastností z rámců vyšší úrovně na rámce pod nimi. Hierarchie rámců je již popsána na *obrázku 5.4* a na tomto obrázku bude popsáno i dědění vlastností rámců. Kořen stromu je v první vrstvě hierarchické struktury. Ve většině případů rámce z nižších vrstev obsahují kromě vlastních vlastností, které nemají žádné jiné rámce, také zděděné vlastnosti všech rámců ve vyšších vrstvách, na které jsou napojeny.

Dědičnost je tedy mechanismus, který dovoluje předávání podobných vlastností rámců z rámce vyšší vrstvy do rámce nižší vrstvy. K popisu dědičnosti se

stejně jako u objektově orientovaného programování, tak i u rámců používají dva základní pojmy a to rodič a potomek. Rodič je v našem případě rámec, který je v hierarchické struktuře ve vyšší vrstvě. Rodič narozdíl od potomka obsahuje obecnější vlastnosti. Potomek je pak rámec, který zdědil některé vlastnosti od rodiče (rámce vyšší vrstvy) a obsahuje již konkrétnější vlastnosti.

Dědictví lze využít prakticky vždy, pokud používáme jen trochu složitější hierarchickou strukturu. Pokud se vrátím k *obrázku 5.4*, tak je jasně vidět, že u příkladu rámce *vozidla*, lze s výhodou dědění vlastností uplatnit. Každý rámec má s rámcem vyšší vrstvy minimálně tu vlastnost, že patří do nadřazené skupiny ve vyšší vrstvě. Tuto vlastnost mají snad všechny logické struktury, takže dědictví lze uplatňovat téměř vždy.

Rámec *vozidla* v první vrstvě hierarchické struktury je jediný rámec co nemá svého rodiče. Tento rámec se nazývá „master-frame“ (hlavní-rámec). Ostatní uvedené rámce mají vždy v nadřazené vrstvě svého rodiče a zároveň jsou rodiči rámců z nižší vrstvy. Rámce z poslední vrstvy hierarchické struktury jsou jediné co nemají žádné potomky.

Příklad dědění vlastností bude uveden na automobilu *Toyota* (*rodič*) a konkrétního typu *Toyota Yaris 1,0 3D (potomek)*. Pokud máme rámec *Toyota* pak bude v položce *počet dveří* uveden rozsah *2 – 5*. Pokud by se pak použil rámec konkrétního modelu *Toyota Yaris 1,0 3D*, zdědil by tento potomek rámce *Toyota* i vlastnost *počet dveří*. U rámce *Toyota Yaris 1,0 3D* již lze uvést konkrétní hodnotu počtu dveří – 3. V předchozím případě rámce *Toyota* to nebylo možné, protože počet dveří musel zahrnovat všechny modely automobilů, které jsou napojeny na rámec *Toyota*.

5.4.5 Výhody rámců

Jak již bylo uvedeno v úvodu kapitoly popisující reprezentaci znalostí pomocí rámců, rámce mají oproti ostatním způsobům reprezentace znalostí řadu výhod. Zde jsou popsány nejvýznamnější výhody reprezentace znalostí pomocí rámců:

- Schopnost jasně dokumentovat informace o okruhu působnosti modelu.
- Modularita informací, povoluje jednoduše systém rozšiřovat a udržovat.

- Informace jsou více čitelné a lépe logicky uspořádané, než je tomu u ostatních způsobů reprezentace znalostí.
- Rámce používají syntaxe pro odkazování na okruh působnosti objektů v pravidlech.
- Rámce jsou vhodné pro budování expertních systémů, které mají grafické rozhraní.
- Přístup k mechanismu, který podporuje stejně jako objektově orientované programování dědictví informací směrem dolů hierarchii rámců.

5.4.6 Rámcová reprezentace znalostí - Příklad č.1.: Škoda Favorit

Reprezentaci znalostí pomocí rámců se lze ukázat na mnoha příkladech. *Obrázek 5.5*. znázorňuje jednoduchý příklad reprezentace znalostí vlastností osobního auta Škoda Favorit. Stejný příklad byl uveden i u sémantických sítí.

JMÉNO RÁMCE: Škoda Favorit
Položky:
IS-A[1]: osobní auto
MOTOR: čtyřdobý benzínový
PŘEVODOVKA: manuální
KARBURÁTOR: Jíkov
BARVA: červená

Obrázek 5.5.: Příklad rámce osobního auta Škoda Favorit.

5.4.7 Rámcová reprezentace znalostí - Příklad č.2.: Řízení a Regulace 1

Dalším příkladem může být předmět Řízení a Regulace 1. Rámec reprezentující vlastnosti předmětu Řízení a Regulace 1 by mohl vypadat, asi jak je uvedeno na *obrázku 5.6.*

JMÉNO RÁMCE: Řízení a Regulace 1

Položky:

IS-A: předmět

ZKRATKA PŘEDMĚTU: BRR1

GARANT: prof. Ing. Petr Vavřín, DrSc.

JAZYK VÝUKY: čeština

FAKULTA: FEKT

GARANTUJÍCÍ USTAV: UAMT

SEMESTR: letní

Obrázek 5.6.: Příklad rámce předmětu Řízení a Regulace 1.

5.4.8 Rámcová reprezentace znalostí - Příklad č.3.: byt

Na posledním příkladu, který se vztahuje k rámcům. Bude pomocí rámců reprezentován byt, který si jistě, každý dovede představit. Na *obrázku 5.7.* je naznačeno jak popisovaný byt vypadá:

Byt:

kuchyň	*jídelna*	*obývák*	*ložnice*
pokoj	*předsíň*	*koupelna*	*WC*

Obrázek 5.7.: Obrázek znázorňující byt.

Rámec popisující tento byt z předchozího *obrázku 5.7.* je na *obrázku 5.8.*

JMÉNO RÁMCE: byt

Položky:

IS-A: panelák

ADRESA: Olomouc, Dlouhá 45, 77900

PATRO: 4

MAJITEL: Novák Jan

VELIKOST: 110 m^2

TYP BYTU: 4+1

MÍSTNOSTÍ: kuchyň, jídelna, obývák, ložnice, pokoj, předsíň, koupelna, WC

Obrázek 5.8.: Rámec reprezentující byt z *obrázku 5.7.*

Na dalším *obrázku 5.9.* je znázorněno, jak by mohl vypadat pokoj v tomto bytu. Na tomto příkladu by šlo reprezentovat všechny místnosti. V případě toho, že by to byl příklad z praxe, musely by se do znalostí expertního systému tyto místnosti přidat. Ale vzhledem k tomu, že tento příklad slouží pouze k vysvětlení, není potřeba zde rozebírat všechny místností bytu. Proto další postup při reprezentaci znalostí bytu bude ukázán právě na pokoji z *obrázku 5.9.*

pokoj:

skříň	*postel*	*noční stolek*	*okno*
dveře	*váza*	*lustr*	*koberec*
stůl	*židle*	*polička*	*televize*
lampa	*počítač*	*rádio*	…

Obrázek 5.9.: Obrázek znázorňující pokoj z popisovaného bytu.

Na dalším *obrázku 5.10.* je příklad toho, jak by mohl vypadat rámec pokoj.

JMÉNO RÁMCE: pokoj
Položky:
IS-A: byt
VELIKOST: 30 m^{2j}
ROZMĚRY: 5 x 6 m
PODLAHA: koberec
OKNA: okno_1, okno_2
DVEŘE: dveře_pokoj_předsíň, dveře_pokoj_obývák
NÁBYTEK: skříň, postel, noční stolek, stůl, polička, židle,
ELEKTRICKÉ SPOTŘEBIČE: televize, lustr, rádio, lampa, počítač
…

Obrázek 5.10.: Rámec reprezentující pokoj z *obrázku 5.9.*

V popisu pokoje nebude dále pokračovat popisováním všech položek, které pokoj obsahuje. Pro názornost zde bude uvedeno pouze několik dalších příkladů rámců. A to okna, dveří z pokoje do předsíně a televize.

JMÉNO RÁMCE: televize
Položky:
IS-A: pokoj
TYP: LCD televize
ÚHLOPŘÍČKA: 70 cm
POMĚR STRAN: 4:3
ROZLIŠENÍ: 1024x768
ZNAČKA: Samsung

Obrázek 5.11.: Příklad rámce reprezentujícího televizi umístěnou v pokoji.

JMÉNO RÁMCE: okno_1
Položky:
IS-A: pokoj
TYP: plastové okno
ROZMĚRY STRAN: 1,2 x 1,4 m
POLOHA: severní strana
BARVA RÁMU: bílá

Obrázek 5.12.: Příklad rámce reprezentujícího okno_1 v pokoji.

JMÉNO RÁMCE: dveře_pokoj_předsíň
Položky:
IS-A: pokoj, předsíň
TYP: jednokřídlé
ROZMĚRY: 0,8 x 2,1 m
POLOHA: jižní strana
BARVA: bílá
OTEVÍRÁNÍ: do předsíně

Obrázek 5.13.: Příklad rámce reprezentujícího *dveře_pokoj_předsíň* v pokoji.

Jak je vidět z rámců položek, které zde byly uvedeny, celkový příklad reprezentace bytu by byl značně rozsáhlý. V reprezentaci dalších položek bytu a jeho místností, by se dalo pokračovat prakticky do nekonečna. Pokud by příklad pokračoval v koupelně, popis by vedl přes umyvadlo, vanu, zrcadlo a skončit by se dalo na koření, které je obsahem poliček v kuchyni.

6. POPIS VYTVOŘENÉHO EXPERTNÍHO SYSTÉMU

6.1 ZADÁNÍ PRO VYTVOŘENÍ EXPERTNÍHO SYSTÉMU

Hlavním zadaným úkolem je navrhnout jednoduchý expertní systém (strukturu jeho báze znalostí a rozhodovací mechanismus), který bude pro ukládání znalostí využívat rámce. Expertní systém je určen pro samostatné studium žáků středních škol a umožní jejich automatické hodnocení a doučování podle jejich potřeb.

Dalším úkolem bylo vytvořit k expertnímu systému uživatelské rozhraní pro provádění konzultace a tento expertní systém odladit.

6.2 ÚVOD

Vytvořený expertní systém obsahuje tři hlavní módy a to zadávání teorie, zadávání otázek a zkoušení. První dva módy slouží k nastavení báze znalostí expertního systému. V módu zkoušení lze pak spustit konzultaci. Po dokončení konzultace následuje vyhodnocení výsledků.

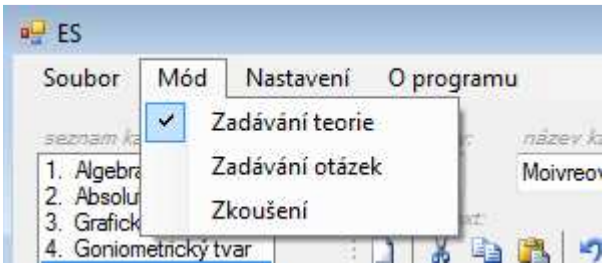
Nejprve zde bude popsáno nastavení báze znalostí, kde v prvním módu expert zadává teoretické kapitoly, které bude zkoušet. V dalším módu se nastavují otázky, které jsou nejpodstatnější částí nastavení expertízy. Expert zde nastavuje otázky, odpovědi a vazby otázek na jednotlivé kapitoly. Vazby jsou při expertíze velice důležité pro rozhodování, jak při výběru otázek, tak v algoritmech rozhodovacího mechanismu. K nastavení báze znalostí patří také položka nastavení, kde expert vybere jeden z algoritmů pro výběr otázek a jeden z algoritmů pro výpočet hodnot, které znázorní zda student zadané kapitoly umí. Nakonec zde nastavuje hranici úspěšnosti.

V předchozích módech se nastavovalo vše co bylo potřeba pro konzultaci. V posledním módu zkoušení se vybere kapitola a po zadání identifikačních údajů studenta se spustí konzultace. Po ukončení expertízy se zobrazí výsledky konzultace.

Z hlediska architektury expertních systémů obsahuje program pouze některé části z popsaných v kapitole 4. Kromě základních částí expertního systému, což jsou báze znalostí, báze dat a rozhodovací mechanismus, obsahuje program komunikační modul, generátor výsledků a vysvětlovací modul.

6.3 KOMUNIKAČNÍ MODUL

Komunikační modul expertního systému se skládá ze tří hlavních částí. První část slouží pro nastavení kapitol, ze kterých budou při konzultaci studenti zkoušeni. Druhou důležitou částí programu je možnost zadávání otázek pro pozdější konzultaci. Poslední částí programu je samotné zkoušení. Tyto tři části lze v programu přepínat pomocí volby: „*Hlavní menu → Mód →…*“.



Obrázek 6.1.: Přepínaní mezi jednotlivými módy programu.

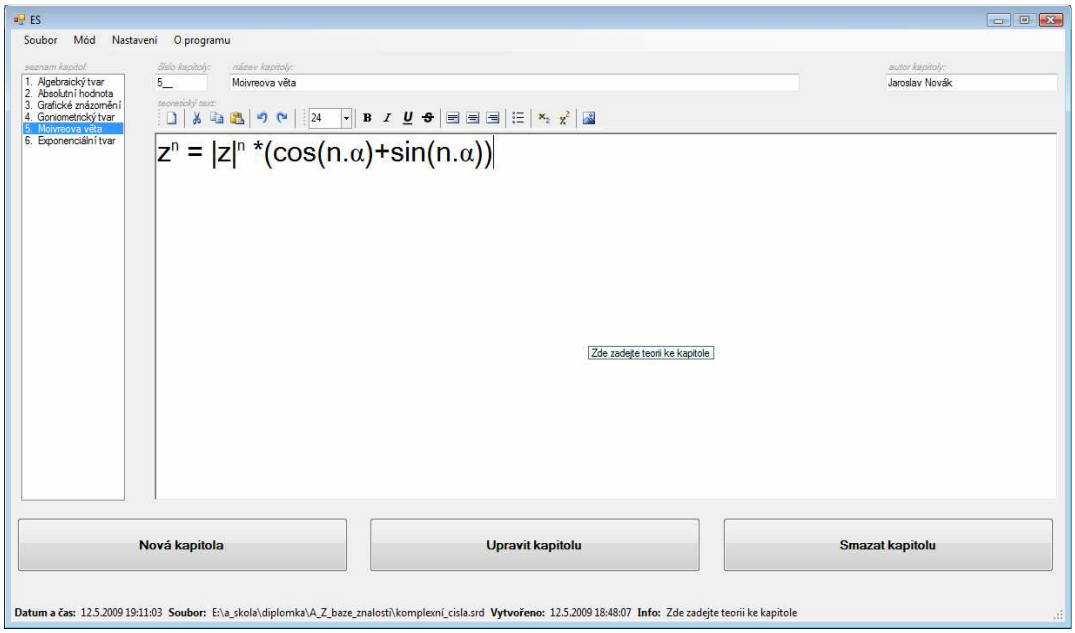
6.3.1 Zadávání teorie

Po zvolení volby „*Hlavní menu → Mód →* ***Zadávání teorie***“ se expertovi ukáže obrazovka z *obrázku 6.2*, která slouží pro zadávání teorie.

Pro zadání nové kapitoly musí expert vyplnit políčka: *číslo kapitoly*, *název kapitoly*, *autor kapitoly* a *teoretický text*. Po zadání všech zmíněných polí může expert kliknutím na tlačítko *Nová kapitola* přidat do seznamu vytvořených kapitol. V případě, že chce expert již zadanou kapitolu upravit, označí ji v *seznamu kapitol*, zvolená kapitola se mu zobrazí do patřičných políček a expert po provedení potřebných úprav klikne na tlačítko *Upravit kapitolu*. Podobným postupem se

vytvořené kapitoly mažou. Expert označí patřičnou kapitolu v *seznamu kapitol* a kliknutím na tlačítko *Smazat kapitolu* odebere kapitolu ze *seznamu kapitol*.

Při zadávání kapitol je pro správnou funkci expertního systému velice důležité, aby se pořadí kapitol shodovalo s návazností jednotlivých kapitol. V případě, že by expert zaměnil pořadí kapitol, mohla by při konzultaci nastat situace, kdy si student vybere zkoušení určité kapitoly a v konzultaci by pak dostával otázky na kapitolu, kterou ještě umět ani nemá.



Obrázek 6.2.: Obrazovka programu při zapnutém módu „*Zadávání teorie*“.

Program vždy po přidání nové kapitoly seřadí všech *n* vytvořených kapitol do posloupnosti od *1* do *n*. Pokud chce expert vložit novou kapitolu mezi již vytvořené, napíše do políčka *číslo kapitoly* číslo, na které chce novou kapitolu vložit. V případě, že expert zadává kapitoly ve správném pořadí, nemusí se o tuto hodnotu starat a nová kapitola vždy dostane číslo o 1 větší, než poslední přidaná. Může nastat situace, že se do políčka zadá číslo větší, než je aktuální počet kapitol. Potom po klinutí na tlačítko *Nová kapitola* se do hodnoty *číslo kapitoly* uloží číslo o 1 větší, než měla poslední vytvořená kapitola.

Políčko pro *název kapitoly* neslouží pouze pro pojmenování této kapitoly, ale také před spuštěním konzultace informuje studenta o tom, z čeho bude zkoušen. Po konzultaci je každá zkoušená kapitola hodnocena zvlášť.

Políčko *autor* je zde pro zadání autora teoretického textu, aby studenti věděli komu nahlásit případné chyby nebo nesrovnalosti.

Poslední políčko *teoretický text* slouží jako učební text pro studenty, kteří v testu neobstáli. Pro tento text program obsahuje základní funkce pro textový editor. To jsou: vyjmout, kopírovat, vložit, zpět, vpřed, změna velikosti písma, tučné písmo, kurzíva, podtržené písmo, přeškrtnuté písmo, dolní index a horní index. Text v políčku t*eoretický text* lze také zarovnat vpravo, vlevo nebo na střed. Do textu lze načíst také obrázek nebo text formátovat pomocí seznamu s odrážkami.

6.3.2 Zadávání otázek

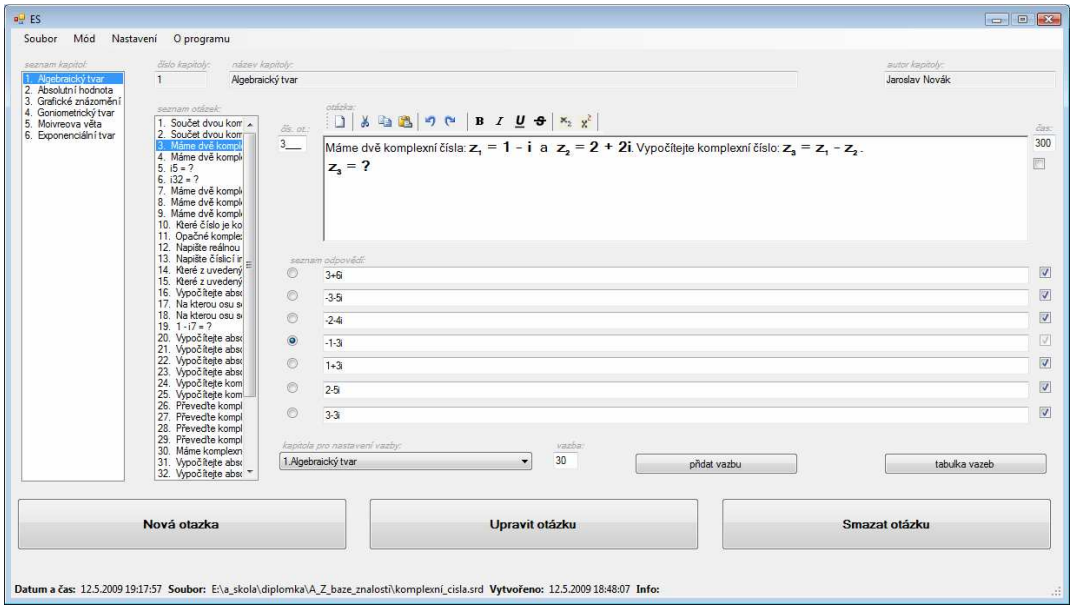
Po zadání kapitol přechází expert pomocí volby: „*Hlavní menu → Mód →* ***Zadávání otázek***“ k zadávání otázek pro testování znalostí studentů. K tomu, aby program expertovi umožnil zadávat otázky, musí být v programu zadána, alespoň jedna kapitola.

Po zapnutí módu *zadávání otázek* se expertovi ukáže obrazovka z *obrázku 6.3*. Aby mohl expert přidat novou otázku, musí vyplnit následující políčka: *číslo otázky*, *otázka*, *čas*, *vazba* a musí zadat, alespoň jednu odpověď na uvedenou otázku. Kromě vyplnění zmíněných políček, musí být označena v *seznamu kapitol* kapitola, ke které se vztahuje hodnota z políčka *vazba*.

Políčko *číslo otázky* zde funguje stejně jako při vytváření kapitol. Pořadí otázek není pro konzultaci tak důležité, jak tomu bylo při zadávání kapitol. Kromě jedné z možností nastavení rozhodovacího mechanismu, kdy jsou otázky při konzultaci položeny přesně v pořadí v jakém jsou zadány do seznamu otázek, nemají čísla otázek podstatný vliv na konzultaci.

Nejdůležitějším políčkem v tomto módu je *otázka*, kde expert zadává otázky, které pak student dostává při konzultaci. V případě, že expert chce, aby student odpověď zadával je dobré do tohoto políčka napsat kromě otázky, také nápovědu nebo návod jak odpověď zadat. Příkladem této nápovědy může být: „Zadejte

odpověď číselně v metrech“. Pro formátování textu otázky jsou zde některé standardní funkce pro textový editor: tučné písmo, kurzíva, podtržené písmo, přeškrtnuté písmo, horní a dolní index. Další užitečné funkce jsou: vyjmout, kopírovat, vložit, zpět a vpřed.



Obrázek 6.3.: Obrazovka programu při zapnutém módu „*Zadávání otázek*“.

Dalším políčkem je *čas*. Zde expert zadává čas v sekundách, který dává studentovi na to, aby na otázku odpověděl. Pod tímto políčkem se nachází ovládací rámeček, které slouží pro zapnutí funkce automatické nastavení času. Tato funkce na základě počtu znaků, které bude muset student pro zodpovězení otázky přečíst, zvolí čas automaticky. Funkce se hodí pro ladění otázek nebo pro otázky, na které by měl student znát odpověď okamžitě po přečtení, ale je nepoužitelná pro časově náročné otázky, kdy student musí například počítat.

Expert po zadání otázky a času pro zadání odpovědi, musí zadat do *seznamu odpovědí*, alespoň jednu odpověď. *Seznam odpovědí* může obsahovat 1 až 7 odpovědí. Pro zvolení správné odpovědi jsou nalevo od políček pro zadání textu odpovědí tlačítka. Vždy je zde označena jedna správná odpověď. Napravo od políček textu odpovědí se nachází ovládací rámečky, které slouží pro volbu, zda danou

odpověď přiřadit k otázce nebo ne. Program neumožňuje odebrat ze *seznamu odpovědí* správnou odpověď. Pokud chce expert odebrat odpověď, která je označena jako správná, musí nejprve označit jinou odpověď jako správnou, pak lze již tuto odpověď ze seznamu odebrat. Na konzultaci má zásadní vliv, zda do *seznamu odpovědí* expert zadá pouze jednu nebo více odpovědí. V případě, že expert zadá pouze jednu odpověď, student při konzultaci musí odpověď napsat. Zde je na expertovi, aby této situaci přizpůsobil jak otázku, tak samotnou odpověď. Při konzultaci uvidí student počet znaků, které má zadat a políčko pro zadání odpovědi mu nedovolí zadat více znaků, než má správná odpověď. Pokud expert přidá do *seznamu odpovědí* více, než jednu odpověď student správnou odpověď při konzultaci pouze vybere ze seznamu.

Poslední políčko, které expert při zadávání otázek vyplňuje je *vazba*. Do tohoto políčka zadá číslo od 0 do 99, které vyjadřuje vazbu mezi touto otázkou a kapitolou označenou v *seznamu kapitol*. Podle této vazby později program vybírá otázky a počítá jak dobře student jednotlivé kapitoly umí.

Po přidání otázky do *seznamu otázek* pomocí tlačítka *Nová otázka*, může expert doplnit další vazby mezi otázkou a jinými kapitolami. K tomu slouží tlačítko *Přidat vazbu*. Toto tlačítko po stisknutí vloží mezi aktuálně označenou kapitolu v *seznamu kapitol* a označenou otázku v *seznamu otázek* vazbu o hodnotě, která je zadaná do políčka *vazba*.

tabulka vazeb:

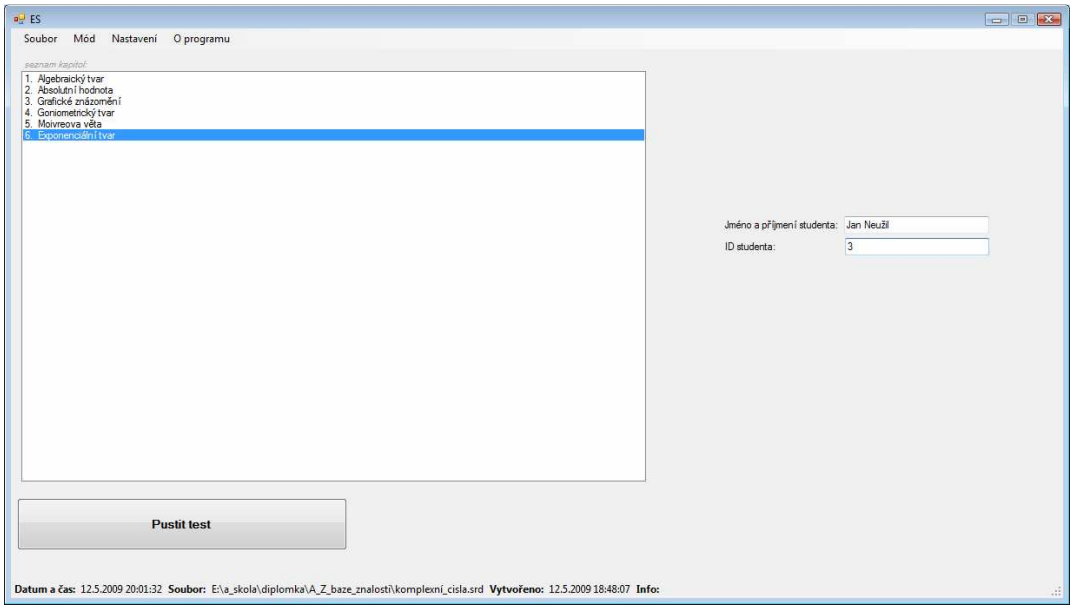
		1.Součet dvou k...	2.Součet dvou k...	3.Máme dvě kom...	4.Máme dvě kom...	5.i5 = ?
▶	1.Algebraický tvar	20	20	30	60	80
	2.Absolutní hodn...	0	0	0	0	0
	3.Grafické znázor...	0	0	0	0	0
	4.Goniometrický t...	0	0	0	0	0
	5.Moivreova věta	0	0	0	0	0
*	6.Exponenciální t...	0	0	0	0	0

Obrázek 6.4.: Obrazovka programu po stisknutí tlačítka *Tabulka vazeb* v módu *Zadávání otázek*.

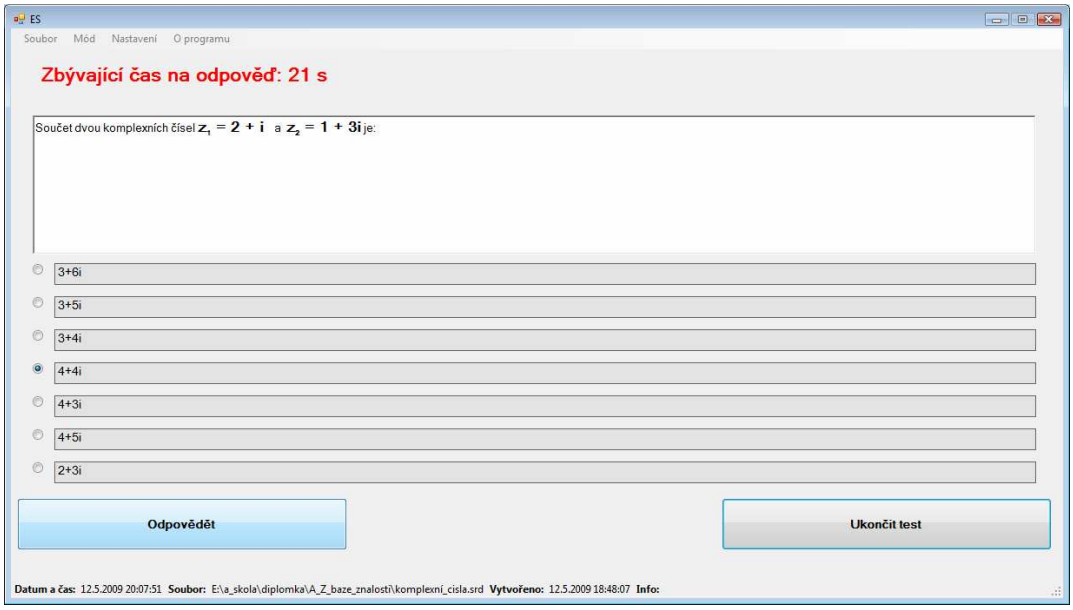
Pro přehlednější editaci vazeb mezi kapitolami a otázkami se zde nachází tlačítko *Tabulka vazeb*. Po stisknutí tohoto tlačítka se zobrazí kompletní tabulka vazeb. Obrazovka programu po stisknutí tlačítka *Tabulka vazeb* je zobrazena na *obrázku 6.4*. V této tabulce může expert jednoduše a přehledně editovat jednotlivé vazby mezi všemi kapitolami a otázkami. Než tabulku zavře musí vazby uložit pomocí tlačítka *Uložit vazby*.

6.3.3 Zkoušení

Pro spuštění zkoušení (konzultace) se vybere volba: „*Hlavní menu → Mód →* ***Zkoušení***“. Po zadání této volby se studentovi ukáže obrazovka, kterou můžete vidět na *obrázku 6.5*. Student zde vybere kapitolu v *seznamu kapitol* a zadá svoje identifikační údaje. Zde je pro konzultaci velice důležité, kterou kapitolu pro zkoušení student vybere. Při konzultaci mu bude program pokládat otázky tak, aby se nikdy nevztahovaly ke kapitolám, které se v *seznamu kapitol* nachází za vybranou kapitolou. V případě, že byly do *seznamu otázek* přidány takové otázky po kliknutí na tlačítko *Spustit test* se spustí konzultace. Obrazovka programu po spuštění konzultace je na *obrázku 6.6*.



Obrázek 6.5.: Obrazovka programu po spuštění módu zkoušení, kde student vybere kapitolu ze seznamu kapitol a zadává identifikační údaje.



Obrázek 6.6.: Obrazovka programu po zadání identifikačních údajů, kdy student odpovídá na otázky, které program vybere.

6.4 BÁZE ZNALOSTÍ

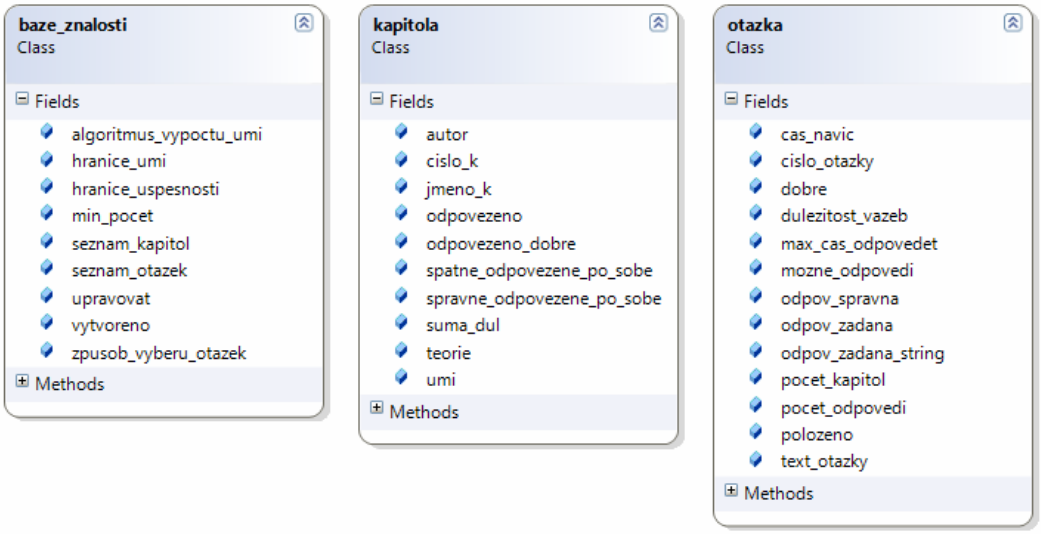
Vytvořenou a nastavenou bázi znalostí lze uložit dvěma způsoby a to binárně a nebo do XML souboru. Pokud se uloží binárně vznikne soubor s koncovkou „*.srd“, v případě uložení jako XML vznikne soubor s koncovkou „*.xml“.

První způsob uložení je do binárního souboru. Zde se expertovi nabízí dvě možnosti, kdy si může vybrat, zda bázi znalostí uloží jako „bázi znalostí“ nebo pouze jako „test“. Pokud se rozhodne pro první možnost, soubor lze znovu otevřít a bázi znalostí dále ladit. Student může tento soubor použít pro konzultaci, kde se ve vlastním zájmu dozví jak jsou na tom jeho znalosti. Pokud expert zvolí možnost „uložit jako test“ po otevření tohoto souboru, lze provést pouze konzultaci již nastavené báze znalostí a bázi již nelze upravovat. Možnost uložit jako test je zde pro vyučující, kteří chtějí program použít jako nástroj pro zkoušení studentů.

Druhou možností uložení báze znalostí je do XML souboru. Tato možnost je zde pro případné nastavení báze znalostí v textovém režimu bez používání tohoto

programu. Použít uložení jako XML je nevhodné pro zkoušení studentů, protože ze souboru lze jednoduše vyčíst správné odpovědi na pokládané otázky.

Pro ukládání dat báze znalostí používám rámcovou reprezentaci znalostí. Což v programování znamená ukládat data pomocí tříd. Jsou zde definovány 3 třídy, které tvoří základní strukturu programu. První je třída *kapitola*, další je *otazka* a třetí je třída *baze_znalosti* do které se uloží seznamy vytvořených tříd *kapitola* a *otazka* a další informace o nastavení báze znalostí.



Obrázek 6.7.: Grafické znázornění tříd vytvořených v programu C#.

6.4.1 Třída kapitola

V třídě kapitola se ukládají zadaná data do políček: *číslo kapitoly* (`public int cislo_k`), *název kapitoly* (`public string jmeno_k`), *autor* (`public string autor`) a *teoretický text* (`public string teorie`). Kromě zmíněných proměnných obsahuje třída kapitola další, které se při volání konstruktoru nastaví automaticky bez přičinění experta. První z nich je hodnota vyjadřující jak student danou kapitolu umí v rozsahu 0 až 1. Na začátku o vědomostech studenta není nic známo, proto hodnotu nastavuji na 0,5 (`public double umi = 0.5`). Další hodnota v třídě kapitola je proměnná počítající počet otázek, které byly v konzultaci položeny (`public int odpovezeno = 0`). Následující proměnná je (`public int odpovezeno_dobre = 0`)

a počítá kolik otázek bylo při konzultaci odpovězeno dobře. Další dvě vyjadřují počet otázek, které student odpověděl za sebou správně nebo špatně (`public int spravne_odpovezene_po_sobe`) a (`public int spatne_odpovezene_po_sobe`). Poslední proměnnou třídy je součet všech vazeb otázek položených při konzultaci na tuto kapitolu (`public int suma_dul = 0`).

Proměnné `umi`, `odpovezeno`, `odpovezeno_dobre`, `spatne_odpovezene_po_sobe`, `spravne_odpovezene_po_sobe` a `suma_dul` se používají při výpočtech rozhodovacích mechanismů, které budou popsány později v kapitole 6.5.

6.4.2 Třída otázka

Stejně jako třída kapitola i tato třída ukládá zadané informace o jednotlivých otázkách do proměnných. Do proměnné (`public int cislo_otazky`) se ukládá číslo otázky, které zde nemá žádnou funkci a slouží pouze k seřazení otázek v seznamu a jako index seznamu. Další proměnnou třídy otázka je (`public string text_otazky`), kde se ukládá text otázky, který uvidí student pokud bude položena při konzultaci. Expert dále nastaví odpovědi na tuto otázku a ty se uloží do pole možné odpovědi (`public string[] mozne_odpovedi = new string[7]`). Expert může zvolit počet odpovědí v rozsahu 1 až 7. Tento počet se ukládá také do proměnné třídy otázka (`public int pocet_odpovedi`). Jedna z odpovědí musí být označena jako správná. Číslo této odpovědi se uloží do proměnné (`public int odpov_spravna`). Expert u otázky nastavuje také čas v sekundách (`public int max_cas_odpovedet`), který má student vymezený na odpověď. Velice důležitou proměnnou třídy otázka je seznam důležitosti vazeb otázky na jednotlivé kapitoly (`public List<int> dulezitost_vazeb = new List<int>()`). Index seznamu vazeb odpovídá indexu seznamu kapitol. V případě, že se expert rozhodne jednu kapitolu smazat, automaticky se smaže i hodnota vazby v seznamu u každé otázky.

Třída *otazka* používá pro svou funkci i další proměnné, které expert zadat nemůže. Například k inicializaci seznamu vazeb se používá proměnná (`public int pocet_kapitol`), kde se uloží aktuální počet vytvořených kapitol. Další proměnné

třídy *otazka* se nastavují až při konzultaci. Pokud je otázka při konzultaci použita do proměnné (`public int polozeno`) se uloží 1. Než je otázka položena je v proměnné hodnota 0. Pokud na tuto položenou otázku student odpoví, jeho odpověď se uloží do jedné z proměnných (`public int odpov_zadana`) nebo (`public string odpov_zadana_string`). Pro uložení zvolené odpovědi jsou potřeba dvě proměnné, protože v případě, že byla expertem uvedena pouze jedna možná odpověď, pak student zadává text odpovědi a ta se uloží jako text. V případě, že expert k otázce přiřadil více odpovědí pak student vybere jednu z nich a číslo vybrané odpovědi se ukládá jako číslo. Potom co student odpoví na otázku, uloží se do proměnné (`public int cas_navic`) čas, který student pro odpověď nevyužil. Poslední proměnná (`public int dobre`) vyjadřuje, zda student odpověděl na tuto otázku dobře, pak má proměnná hodnotu 1, pokud odpoví špatně zůstává jí hodnota 0.

6.4.3 Třída baze_znalosti

Tato třída je hlavní a obsahuje dva seznamy proměnných. V jednom jsou všechny vytvořené kapitoly (`public List<kapitola> seznam_kapitol`) a ve druhém všechny vytvořené otázky (`public List<otazka> seznam_otazek`). V třídě *baze_znalosti* je uložena informace, zda se soubor uložil jako test nebo jako báze znalostí pro další ladění báze. Informace je uložena v proměnné (`public bool upravovat`). Pokud se soubor uloží jako test do proměnné se uloží hodnota `false` a po otevření souboru lze pustit pouze konzultaci. V případě, že se soubor uloží jako báze znalostí v proměnné je hodnota `true` a po otevření lze bázi dále ladit. Další proměnná třídy *baze_znalosti* umožňuje uložit zvolený algoritmus (`public int algoritmus_vypoctu_umi`) pro výpočet hodnoty proměnné *umi* pro jednotlivé kapitoly. Hodnoty kterých tato proměnná nabývá budou popsány v kapitole 6.5.2. Další proměnná (`public int zpusob_vyberu_otazek`) také ovlivňuje rozhodovací mechanismus. Určuje pořadí otázek, které se budou pokládat. Důkladnější popis proměnné uvedu v kapitole 6.5.1. Další proměnná (`public int hranice_uspesnosti`) určuje na kolik procent musí student v testu uspět, aby jeho

konzultaci program vyhodnotil jako úspěšnou. Proměnná (`public int min_pocet`) má znovu vliv na rozhodovací mechanismus. Některé způsoby výběru otázek, které při konzultaci nepoloží všechny otázky ji využívají jako hranici pro minimální počet otázek, které budou z každé zkoušené kapitoly položeny. Další proměnnou (`public double hranice_umi`) znovu využívá funkce pro výběr následující otázky, ale na rozdíl od předchozích má u každého algoritmu jinou funkci. Funkce proměnné (`public double hranice_umi`) jsou podrobněji popsány v kapitole 6.5.3. Do poslední proměnné (`public string vytvoreno`) se ukládá při uložení třídy do souboru aktuální datum a čas.

6.5 ROZHODOVACÍ MECHANISMUS

Rozhodovací mechanismus tvoří u tohoto expertního systému dvě funkce a jedna proměnná. První z funkcí slouží pro výběr otázky při konzultaci a druhá podle zvolené odpovědi vypočítá nové hodnoty proměnné *umi* u jednotlivých kapitol. Poslední části rozhodovacího mechanismu je proměnná, která určuje hranici mezi úspěšnou a neúspěšnou konzultací v jednotlivých kapitolách.

6.5.1 Funkce pro výběr otázky

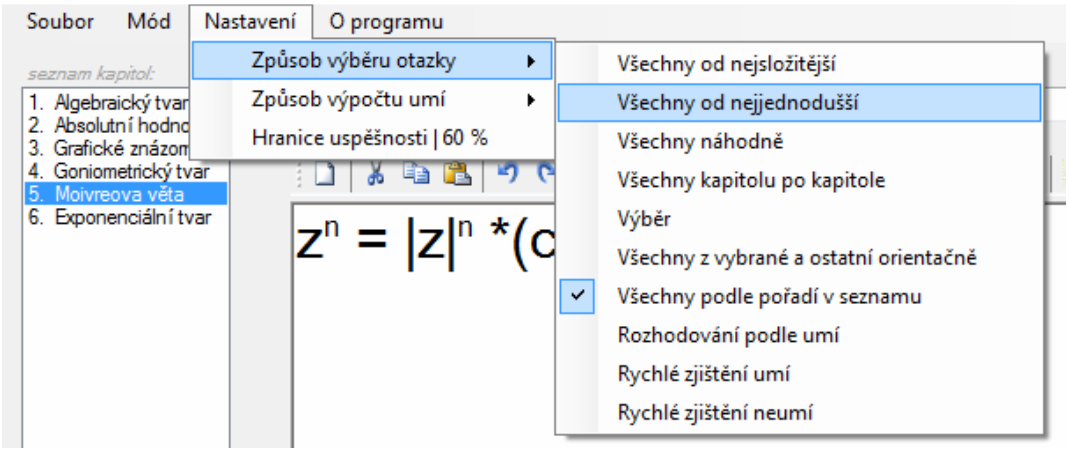
Tato funkce má hlavičku:

`int fce_vyber_otazky(int mod_vyberu, int vybrana_kapitola_fce)` a podle vstupních parametrů vybere vždy následující otázku při konzultaci. Prvním parametrem je proměnná (`int mod_vyberu`), podle ní funkce pozná, který ze způsobů výběru otázek má použít. Způsob výběru lze nastavit v: „*Hlavní menu → Nastavení → Způsob výběru otázky*“. Druhým parametrem je proměnná `int vybrana_kapitola_fce`, která funkci říká, do které kapitoly může nejvýše vybírat otázky. V této proměnné je uložený index vybrané kapitoly pro konzultaci a jestliže má otázka vliv na kapitolu s vyšším indexem nelze ji pro konzultaci použít.

Výstupní hodnota této funkce je typu `int` a znázorňuje index otázky, která bude v konzultaci následovat. Výstupní hodnota se vždy nachází v rozsahu od *-1* do (*počet otázek – 1*). V případě, že je výstupní hodnota od *0* do (*počtu otázek – 1*),

slouží jako index otázky, která se po volání této funkce položí v konzultaci. Může zde nastat situace, že funkce již nenajde žádnou vhodnou otázku a pak na výstup pošle hodnotu -1. Jestliže je výstupem této funkce -1 zbytek programu pozná, že se má konzultace ukončit.

V následujícím textu se nachází popis všech možností výběru otázek, které jsou zobrazeny na *obrázku 6.8.*



Obrázek 6.8.: Znázornění způsobu výběru otázek v programu.

Všechny od nejsložitější

Jestliže expert nastaví: „*Hlavní menu → Nastavení → Způsob výběru otázky → **Všechny od nejsložitější***“ pak se `mod_vyberu = 1` a znamená to, že se budou postupně volit nejsložitější otázky. Algoritmus v tomto případě počítá pouze s vazbami, které souvisí s kapitolami nacházejícími se v pořadí do vybrané kapitoly. Sečte tedy všechny vazby na zkoušené kapitoly u každé otázky. Následně z tohoto výběru odebere všechny otázky, které mají jakoukoliv nenulovou vazbu na kapitoly nacházející se za vybranou. Nakonec hledá maximální hodnotu součtu vazeb na zkoušené kapitoly u otázek, které nebyly dosud v konzultaci položeny. Pokud nalezne takovou otázku pošle na výstup index této otázky. Expert tuto možnost

nastaví v případě, že chce, aby se při konzultaci student co nejrychleji dozvěděl jak jsou na tom jeho znalosti a zároveň budou položeny všechny otázky.

Příklad výběru otázek pro algoritmus výběru otázek „*Všechny od nejsložitější*“:

Tabulka 6.1.: Znázorňuje vzájemné vazby mezi kapitolami A, B, C, D, E, F, G a otázkami 1 až 15, pro výběr otázek: „*Všechny od nejsložitější*“.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	50	40	20		10					10		30		10	
B			20	30	10	40		20		10	10	30			
C		30		70		30		30	40		20	20	20		40
D			50		80	20			50	30					
E							60			30	30	60		30	20
F								20	10		10		50		
G											30		20	30	70
∑	50	70	90	100	100	90	60			80		140			

Přehled vazeb se nachází v *tabulce 6.1*. Před zahájením konzultace student vybral kapitolu E. Protože otázky 8, 9, 11, 13, 14 a 15 mají nenulové vazby na kapitoly F nebo G nemohou být v konzultaci položeny. Následně algoritmus hledá nejdůležitější otázky. První je tedy položena otázka 12, která má součet vazeb na zkoušené kapitoly 140. Po otázce 12 následují otázky: 4, 5, 3, 6, 10, 2, 7 a 1.

Všechny od nejjednodušší

Další možností nastavení výběru otázek: „*Hlavním menu* → *Nastavení* → *Způsob výběru otázky* → ***Všechny od nejjednodušší***“ pak se `mod_vyberu = 2`. Tento způsob výběru následující otázky je velice podobný jako způsob „*Všechny od nejsložitější*“. Algoritmus znovu bere v potaz otázky, které mají pouze nulové vazby na kapitoly, které se nacházejí za vybranou. Sečte vazby na zkoušené kapitoly a hledá mezi nimi minimální nenulovou hodnotu (což je rozdíl oproti předchozí verzi, kde se vybere maximální hodnota). Při této volbě se v konzultaci nejprve položí nejjednodušší otázky a obtížnost se bude postupně zvyšovat.

Příklad výběru otázek pro algoritmus výběru otázek „*Všechny od nejjednodušší*“:

Tabulka 6.2.: Znázorňuje vzájemné vazby mezi kapitolami A, B, C, D, E, F, G a otázkami 1 až 15, pro výběr otázek: „*Všechny od nejjednodušší*“.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	50	40	20		10					10		30		10	
B			20	30	10	40		20		10	10	30			
C		30		70		30		30	40		20	20	20		40
D			50		80	20			50	30					
E							60			30	30	60		30	20
F								20	10		10		50		
G											30		20	30	70
∑	50	70	90	100	100	90	60			80		140			

Pro přehlednost je zde použito stejné nastavení vazeb jako v předchozím příkladu a student pro konzultaci vybral znovu kapitolu E. Vazby jsou nastaveny podle tabulky 6.2. Nejprve se provede stejně jako v minulém případě, povinné odstranění otázek: 8, 9, 11, 13, 14 a 15, které by nemohl vybrat žádný algoritmus kvůli vazbám na kapitoly, které student ještě neumí nebo nechce zkoušet. Pak tento algoritmus pracuje obráceně než předchozí. Otázky vybírá od nejjednodušší. Nejdříve vybere otázku číslo 1, protože má nejmenší vliv na zkoušené kapitoly. Následuje 7, 2, 10, 3, 6, 4, 5 a12. Otázky se nepoloží v přesně opačném pořadí, než v minulém případě pouze kvůli otázkám 3 a 6 (4 a 5), které mají stejný součet vazeb. To je dáno tím, že algoritmus vybírá nový index otázky podle podmínky, kde hledá menší součet vazeb (v minulém případě větší). Funkce tedy při stejném součtu vazeb pošle na výstup první otázku, kterou najde. Kromě této maličkosti nemá pořadí nastavených otázek v tomto případě, žádný vliv na konzultaci.

Všechny náhodně

Třetí možností způsobu výběru otázek při konzultaci je: „*Hlavní menu → Nastavení → Způsob výběru otázky →* ***Všechny náhodně***“. Tento algoritmus nejprve rozdělí otázky na dvě skupiny. V první skupině jsou otázky, které nemohou být položeny kvůli nenulovým vazbám na kapitoly nacházející se za vybranou kapitolou

pro konzultaci. V druhé skupině jsou otázky vhodné pro tuto konzultaci. Z této skupiny otázek se pak pomocí generátoru náhodných čísel vybírají postupně všechny otázky. Pro tento případ není třeba uvádět příklad. Výběr otázek nemá žádné stanovené pravidla. Tato volba se využije pro zkoušení, kdy není vhodné, aby student předpokládal, ke které kapitole otázka patří. Konkrétním příkladem může být přímá a nepřímá úměra, kde je hlavním úkolem studentů rozpoznat o který typ příkladu se jedná a nebylo by tedy vhodné, aby student dostal nejprve příklady pouze na přímou úměru a až po nich by následovaly příklady na nepřímou úměru.

Všechny kapitolu po kapitole

Expert může zvolit také: „*Hlavní menu → Nastavení → Způsob výběru otázky →* ***Všechny kapitolu po kapitole***“. V tomto případě algoritmus prochází postupně všechny zkoušené kapitoly a vybírá z nich nejjednodušší otázky. Na první pohled je algoritmus podobný s volbou „*Všechny od nejjednodušší*“. Rozdíl je v tom, že u algoritmu „*Všechny kapitolu po kapitole*“ se položí vždy nejprve všechny otázky z první kapitoly, které mají pouze nulové vazby na kapitoly pozdější i když patří ke zkoušení. Po položení všech otázek vztahujících se pouze k první kapitole se začnou pokládat další otázky, které se mohou vztahovat, buď pouze ke druhé kapitole nebo k druhé a první kapitole. Při této volbě se tedy nemůže stát, že bude při konzultaci položena mezi prvními otázka, která má pouze malou vazbu například na poslední zkoušenou kapitolu, což se u volby „*Všechny od nejjednodušší*“ stát mohlo.

Příklad výběru otázek pro algoritmus výběru otázek „*Všechny kapitolu po kapitole*“:

Algoritmus vybírá nejjednodušší otázky tak, aby neskákaly v kapitolách, ale aby byly i kapitoly zkoušené postupně jak byly zadané. Vazby k tomuto příkladu jsou uvedeny v tabulce 6.3. Nejprve se tedy vybere otázka 1, která jako jediná nemá vazby na jinou kapitolu než A. Při hledání otázek pro kapitolu B se žádná nenajde, protože všechny z výběru otázek mají vazbu nebo vazby i na další kapitoly. Algoritmus tedy přejde ke zkoušení kapitoly C. Následně najde dvě vhodné otázky 2 a 4. Podle součtu vazeb je otázka 2 jednodušší a proto je položí v tomto pořadí. Následuje zkoušení kapitoly D. Algoritmus nalezne tři vhodné otázky a to 3 ($\sum$vazeb =

90), 5 (∑vazeb = 100), a 6(∑vazeb = 90). Nejprve je položena otázka 3, pak otázka 6. Zde se znovu projeví malý vliv pořadí zadání otázek. Pak je položena otázka 5, která má větší součet vazeb než dvě předchozí. Tím končí zkoušení kapitoly D a začne zkoušení kapitoly E. Zde jsou otázky 7 (∑vazeb = 60), 10 (∑vazeb = 80), a 12(∑vazeb = 140). V napsaném pořadí jsou otázky také položeny. U tohoto algoritmu je velice důležité pořadí kapitol, které expert zadal. Na otázce 7 je velice dobře vidět rozdíl mezi tímto algoritmem a algoritmem „*Všechny od nejjednodušší*“, kde byla díky součtu vazeb otázka 7 položena jako druhá. Zde byla položena až jako třetí od konce kvůli vazbě na poslední zkoušenou kapitolu E.

Tabulka 6.3.: Znázorňuje vzájemné vazby mezi kapitolami A, B, C, D, E, F, G a otázkami 1 až 15, pro výběr otázek: „*Všechny kapitolu po kapitole*“.

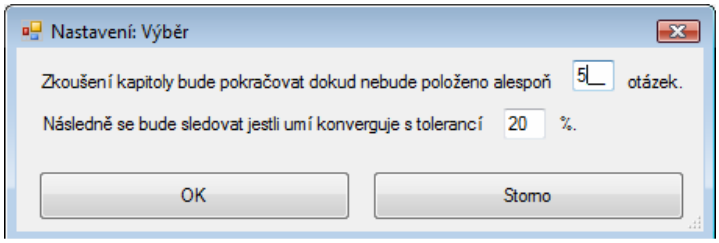
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	50	40	20		10					10		30		10	
B			20	30	10	40		20		10	10	30			
C		30		70		30		30	40		20	20	20		40
D			50		80	20			50	30					
E							60			30	30	60		30	20
F								20	10		10		50		
G											30		20	30	70
∑	50	70	90	100	100	90	60			80		140			

Výběr

Další volbou kterou si může pro výběr otázek expert zvolit je: „*Hlavní menu → Nastavení → Způsob výběru otázky →* ***Výběr***“. Pro tento algoritmus je nutné zadat minimální počet otázek a číslo v procentech. Minimální počet otázek algoritmu říká kolik se minimálně musí studentovi z dané kapitoly položit otázek. Druhé číslo v procentech vyjadřuje rozmezí, ve kterém se musí hodnota *umi* pohybovat po odpovědění na minimální počet otázek. Pokud jsou splněny obě tyto podmínky konzultace přejde ke zkoušení další kapitoly. Algoritmus jsem pojmenoval „*výběr*“, protože vybere prvních *n* otázek *(n = minimální počet otázek)* v celém spektru důležitosti na aktuálně zkoušenou kapitolu. Pokud expert tedy zadá minimální počet otázek 5, algoritmus hledá otázky, které mají vazby co nejblíže hodnotám 0, 25, 50,

75 a 100. Pokud se hodnota *umi* po odpovědění na tyto otázky neustálila v nastaveném rozmezí, bude konzultace pokračovat tak, že algoritmus vždy najde otázku, která má vazbu co nejblíže aktuální hodnotě *umi*. Po splnění i této druhé podmínky se přejde ke zkoušení další kapitoly.

U tohoto algoritmu tedy nelze předem odhadnout, jak bude celá konzultace probíhat. Do expertem nastaveného minimálního počtu otázek jsou otázky předem dané. Ostatní otázky pak závisí na úspěšnosti studenta. Na následujícím *obrázku 6.9.* vidíte obrazovku nastavení tohoto algoritmu.



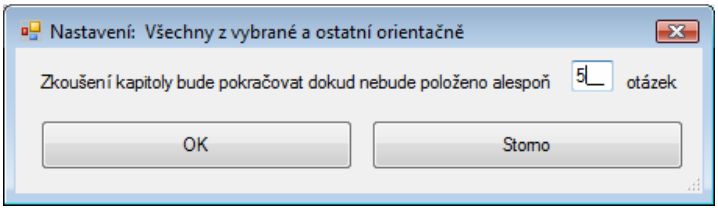
Obrázek 6.9.: Obrazovka nastavení algoritmu „*Výběr*“.

Všechny z vybrané ostatní orientačně

Expert může zvolit také volbu: „*Hlavní menu → Nastavení → Způsob výběru otázky →* ***Všechny z vybrané ostatní orientačně***“. Expert pokud chce zvolit tento způsob výběru otázek nastavuje počet otázek, které mají být položeny z ostatních kapitol. Konzultaci tedy ovlivní počet, který expert zadá. V případě, že by expert zadal například 3, budou se postupně z každé kapitoly pokládat 3 nejdůležitější otázky podle vazeb na tuto kapitolu. Takto budou položeny postupně vždy 3 nejdůležitější otázky ze všech kapitol až po vybranou kapitolu a orientačně se zjistí jak student umí minulé kapitoly. Z vybrané kapitoly se pak položí všechny otázky, aby se důkladně prozkoušela.

Algoritmus samozřejmě hlídá, zda nemá otázka vliv na kapitoly za vybranou. Pokud má otázka nenulovou vazbu na některou z kapitol nacházející se za vybranou kapitolou nebude při konzultaci položena. Při této volbě může nastat případ, že

nebude k dispozici potřebný počet otázek. V takovém případě se položí, alespoň dostupné otázky.



Obrázek 6.10.: Obrazovka nastavení algoritmu „*Všechny z vybrané a ostatní orientačně*“.

Příklad tohoto algoritmu pro výběr následující otázky:

Tabulka 6.4.: Znázorňuje vzájemné vazby mezi kapitolami A, B, C, D, E, F, G a otázkami 1 až 15, pro výběr otázek: „*Všechny z vybrané a ostatní orientačně*“.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	50	40	20		10					10		30	20	90	80
B			20	30	10	40		20		10	10	30	20		
C		30		70		30		30	70		20	20			40
D			50		80	20			50	30			20		
E							60			30	30	60		30	20
F								20	10						
G															70
∑	50	70	90	100	100	90	60			80	60	140	60	120	

Pokud jsou vazby nastaveny jako je tomu v tabulce 6.4. a expert zvolil minimální počet otázek z každé kapitoly 3, algoritmus začne zkoušet orientačně kapitoly A,B,C a D. Najde tedy nejdůležitější otázku pro kapitolu A, což je otázka 14, následují otázky 1 a 2. Algoritmus pak přejde ke zkoušení kapitoly B, protože minimální počet otázek ke kapitole A již byl položen. Nejprve najde otázku 6, pak otázku 4 a 12. Ke kapitole C najde algoritmus pouze otázku 11. Všechny ostatní otázky k této kapitole již byly položeny v rámci zkoušení kapitol A a B. Dále je na řadě kapitola D. Položí se tedy otázky 5, 3 a 10. Poslední kapitolou ke zkoušení je kapitola E, která bude prozkoušena důkladně. Algoritmus nyní hledá nejjednodušší

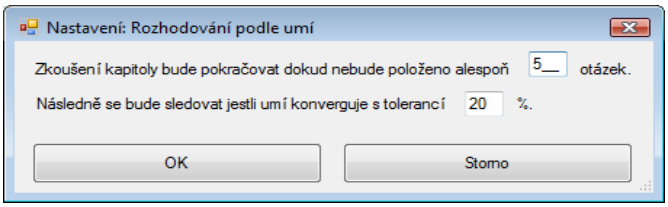
otázku pro tuto kapitolu, což je jediná dosud nepoložená otázka 7, která má vazbu na kapitolu E. V této konzultaci nebyly položeny otázky 8, 9 a 15, které mají vazby na kapitoly F nebo G, ale nebyla položena ani otázka 13. Otázka 13 nebyla položena, protože algoritmus ke každé kapitole na kterou má otázka 13 vazbu našel, alespoň 3 důležitější otázky.

Všechny podle pořadí v seznamu

Expert může zvolit také volbu: „*Hlavní menu → Nastavení → Způsob výběru otázky →* ***Všechny podle pořadí v seznamu***“. V tomto případě má expert pořadí otázek při konzultaci přímo pod kontrolou. Algoritmus položí otázky ve stejném pořadí v jakém byly seřazeny při jejich zadávání. Algoritmus pouze testuje, zda nemá otázka nenulovou vazbu na kapitolu nacházející se za vybranou kapitolou pro zkoušení. V případě, že pro zkoušení je vybrána poslední vytvořená kapitola budou položeny všechny otázky, které mají nejméně jednu nenulovou vazbu. V případě, že má otázka nenulovou vazbu na kapitolu nacházející se za vybranou kapitolou, otázka se jednoduše přeskočí. Tuto volbu expert nastaví v případě, že mu nevyhovuje ani jeden z ostatních algoritmů.

Rozhodování podle umí

Jedním z nejinteligentnějších způsobů výběru otázek je volba: „*Hlavní menu → Nastavení → Způsob výběru otázky →* ***Rozhodování podle umí***“. Zde expert znovu zadává minimální počet otázek, které bude algoritmus pokládat ke každé zkoušené kapitole a také rozmezí ve kterém musí hodnocení studenta být pro ukončení konzultace. Student od počátku dostává otázky, které jsou svou vazbou mezi otázkou a zkoušenou kapitolou co nejblíže k aktuální hodnotě *umi*.

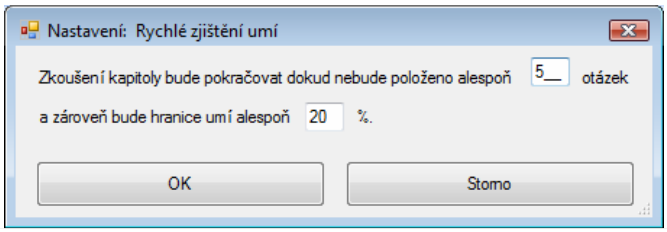


Obrázek 6.11.: Obrazovka nastavení algoritmu „*Rozhodování podle umí*“.

Na počátku je hodnota *umi* u každé kapitoly přednastavena na 50 % (0,5). Student tedy dostane nejprve otázku, která se co nejvíce blíží vazbou na zkoušenou kapitolu hodnotě 50. V případě, že na otázku odpoví špatně, hodnota *umi* se změní na 0 % a student dostane další otázku, která bude mít na zkoušenou kapitolu nenulovou vazbu, ale číslu 0 nejbližší. Stejně jako tomu bylo u algoritmu „*Výběr*“ i zde nelze přesně odhadnout jak bude konzultace probíhat.

Rychlé zjištění umí

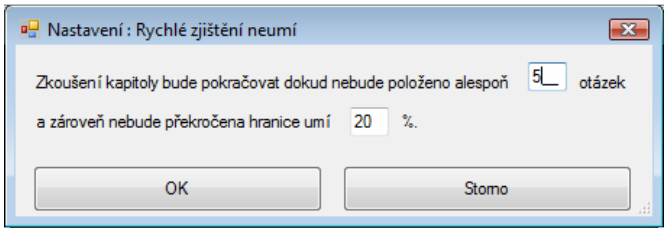
Další možností nastavení výběru otázek je volba: „*Hlavní menu* → *Nastavení* → *Způsob výběru otázky* → ***Rychlé zjištění umí***“. Pokud expert vybere tuto volbu musí nastavit dvě hodnoty pomocí nichž se tento algoritmus dále rozhoduje. První je minimální počet otázek, které budou položeny ke každé zkoušené kapitole (otázky budou položeny, alespoň ty co budou k dispozici). Druhou hodnotou je pak hranice v procentech. Pokud se student nad tuto hranici dostane v hodnocení aktuální kapitoly po minimálním počtu otázek, ve zkoušení této kapitoly se již nepokračuje a přejde se k další kapitole. V opačném případě, kdy hodnocení nepřesahuje nastavenou hranici se dále pokládají otázky, které patří k této kapitole. U tohoto algoritmu se nejprve pokládají ty nejdůležitější otázky, aby se co nejrychleji přešlo k další kapitole, pokud v testu student prokáže patřičné znalosti a hodnocení je nad hranicí určenou expertem. U tohoto algoritmu nelze jednoznačně určit jak bude konzultace probíhat pouze ze znalosti jednotlivých vazeb mezi otázkami a kapitolami. Algoritmus se rozhoduje, také podle toho jak student na jednotlivé otázky odpověděl.



Obrázek 6.12.: Obrazovka nastavení algoritmu „*Rychlé zjištění umí*“.

Rychlé zjištění neumí

Poslední možností nastavení výběru otázek je volba: „*Hlavní menu → Nastavení → Způsob výběru otázky →* ***Rychlé zjištění neumí***“. Pokud expert vybere tuto volbu musí nastavit stejně jako v předchozím případě dvě hodnoty pomocí nichž se tento algoritmus dále rozhoduje. První je minimální počet otázek, které budou položeny ke každé zkoušené kapitole (otázky budou položeny, alespoň ty co budou k dispozici). Druhou hodnotou je pak hranice v procentech, nad kterou pokud se student nedostane v hodnocení aktuální kapitoly po položení minimálního počtu otázek ve zkoušení této kapitoly se již nepokračuje a přejde se k další kapitole. Algoritmus v tomto případě vyhodnotí tuto kapitolu, jakože ji student neumí. V opačném případě, kdy hodnocení přesahuje nastavenou hranici se dále pokládají otázky, které patří k této kapitole. U tohoto algoritmu se nejprve pokládají ty nejjednodušší otázky. V případě, že student nedosáhne expertem nastavené hranice znalostí po několika prvních jednoduchých otázkách, přejde se k další kapitole. U tohoto algoritmu nelze stejně jako v minulém případě jednoznačně určit, jak bude konzultace probíhat pouze ze znalosti jednotlivých vazeb mezi otázkami a kapitolami. Algoritmus se rozhoduje také podle toho, jak student na jednotlivé otázky odpověděl.



Obrázek 6.13.: Obrazovka nastavení algoritmu „*Rychlé zjištění neumí*“.

Srovnání algoritmů pro výběr otázky

V následujícím výčtu algoritmů je popsáno kdy je který algoritmus vhodné použít:

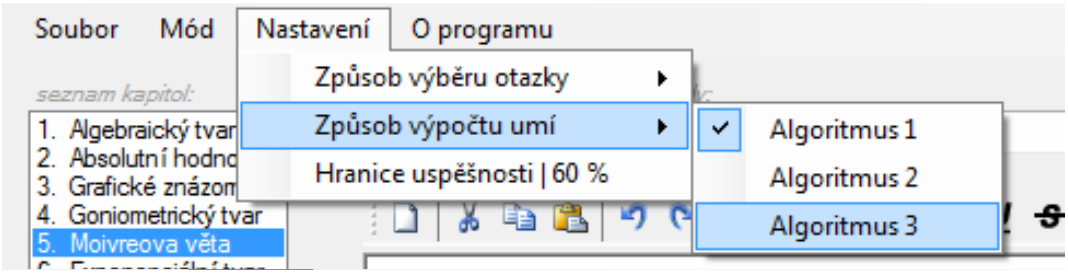
1. *Všechny od nejsložitější* – tento algoritmus je zde pro případ, že by bylo vhodné nejprve položit složitější otázky.
2. *Všechny od nejjednodušší* – je nastaven standardně, protože je praktické, aby si student při testování nejprve uvědomil jednodušší látku a později složitější.
3. *Všechny náhodně* – expert použije v případě, že není vhodné, aby student věděl ze které kapitoly je aktuálně zkoušen.
4. *Všechny kapitolu po kapitole* – je vhodný pro případ, že expert nechce, aby student dostával otázky napříč kapitolami. V tomto případě se nikdy nepoloží otázka z následující kapitoly, dokud nejsou položeny všechny otázky ze všech předchozích kapitol. Otázky se pokládají od nejjednodušší vzhledem k aktuálně zkoušené kapitole.
5. *Výběr* – je prvním algoritmem, který nepoloží všechny otázky. Vybere pouze některé napříč spektrem důležitosti vazeb. Při volbě minimálního počtu otázek 3, vybere z každé kapitoly otázky, které se vazbou nejvíce blíží 0, 50 a 100.
6. *Všechny z vybrané ostatní orientačně* – v případě, že expert chce vyzkoušet pouze jednu kapitolu důkladně. Z ostatních algoritmus položí pouze nejsložitější otázky.
7. *Všechny podle pořadí v seznamu* – tento algoritmus je zde pro případ, že chce mít pořadí otázek při konzultaci expert plně pod kontrolou nebo pokud se mu nehodí žádný z ostatních algoritmů.
8. *Rozhodování podle umí* – algoritmus studentovi pokládá vždy otázky, které jsou vazbou na aktuálně zkoušenou kapitolu nejblíže aktuální hodnotě jeho znalostí.
9. *Rychlé zjištění umí* – v případě, že expert předpokládá, že student danou problematiku umí vybere tento algoritmus. Student pak dostává první nejsložitější otázky z jednotlivých kapitol a to pouze určený počet. V případě, že jeho hodnocení odpovídá hranici určené expertem po položení minimálního počtu otázek přejde se k testování další kapitoly. V opačném případě se začnou pokládat další jednodušší otázky.

10. *Rychlé zjištění neumí* – je algoritmus opačný k předchozímu. Zde expert předpokládá, že student látku neumí. Při konzultaci se tedy nejprve položí nejjednodušší otázky a pokud student dokáže odpovědět správně, konzultace pokračuje. V případě špatných odpovědí vyhodnotí, že student kapitolu neumí a konzultace se ukončí.

6.5.2 Funkce pro výpočet hodnoty umí u vybraných kapitol

Tato funkce realizuje výpočet hodnoty proměnné *umi* pro všechny kapitoly po položení otázky. Hlavička funkce vypadá takto: `void fce_vypocet_umi(int algoritmus, int index, int fce_vybrana_kap)`. Prvním vstupním parametrem této funkce je proměnná `int algoritmus`. Podle této proměnné zjistí funkce, který z možných algoritmů má použít. Další vstupní proměnnou je `int index`, která je zároveň výstupem z předchozí popisované funkce `fce_vyber_otazky`, tedy index zkoušené otázky. Poslední vstupní proměnnou je `int fce_vybrana_kap`, která funkci říká, kterou kapitolu student vybral na začátku konzultace.

Expert může pomocí volby: „*Hlavní menu → Nastavení → Způsob výpočtu umí*“ měnit algoritmus pomocí, kterého se bude provádět výpočet hodnoty *umi* pro jednotlivé kapitoly. Expert si zde může vybrat ze třech různých algoritmů jak je vidět na *obrázku 6.14.*



Obrázek 6.14.: Znázornění výběru algoritmu pro výpočet hodnoty *umi*.

Algoritmus 1

První algoritmus je nastaven standardně a pokud ho expert nezmění, bude výpočet proveden podle tohoto algoritmu. Samotný výpočet je rozdělen na dvě části a to pro případ, že student odpověděl na otázku dobře a pro případ, že student odpověděl špatně. Vzorec pro správnou odpověď:

$$Umi_{nové} = Umi_{staré} + \left[(1 - Umi_{staré}) * \left(\frac{vazba}{suma\_vazeb} \right) \right] \quad \ldots \textbf{\textit{vzorec č. 6.1.}}$$

Pro špatnou odpověď nová hodnota počítá podle následujícího vzorce:

$$Umi_{nové} = Umi_{staré} - \left[Umi_{staré} * \left(\frac{vazba}{suma\_vazeb} \right) \right] \quad \ldots \textbf{\textit{vzorec č. 6.2.}}$$

$Umi_{nové}$... je nově vypočítaná hodnota pro kapitolu

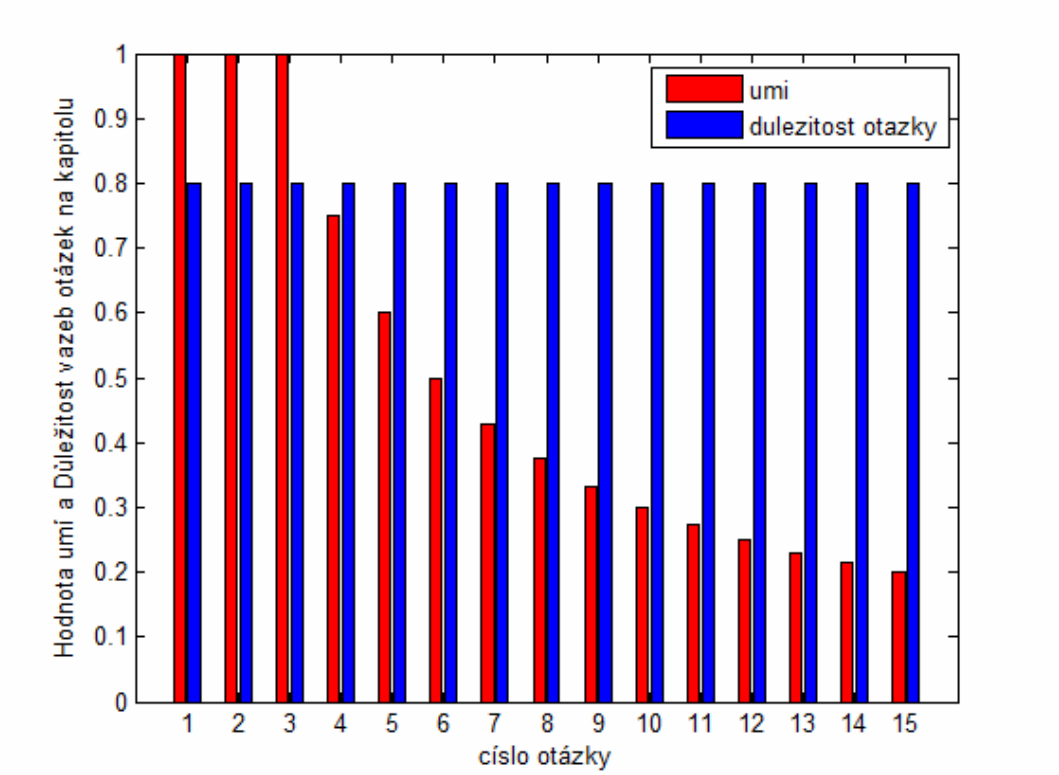
$Umi_{staré}$... je hodnota *umi* před tímto výpočtem

vazba … je vazba položené otázky nastavená expertem na aktuálně počítanou kapitolu

suma_vazeb … je součet vazeb na tuto kapitolu, které expert nastavil u všech do teď položených otázek.

Vzhledem k tomu, že při konzultaci nelze sledovat postup výpočtu algoritmu namodeloval jsem tento vzorec v programu Matlab. Z následujícího *obrázku 6.15*. lze lépe vyčíst jak vzorec funguje.

Z grafu lze odečíst, že student odpověděl první tři otázky dobře a pak všechny špatně. Pokud je pro konzultaci použit tento algoritmus, pak je výsledná hodnota po jakémkoliv počtu odpovědění otázek bez chyby vždy 1 (100 %). V případě, že vazby mají všechny stejnou hodnotu jako na *obrázku 6.15*. pak po třech dobře a třech špatně odpověděných otázkách má *umi* hodnotu 0,5 (50 %).



Obrázek 6.15.: Na grafu je vidět chování algoritmu č.1 v případě, že všechny otázky mají stejnou vazbu (důležitost otázky) na zkoušenou kapitolu. První tři otázky odpověděl student dobře a všechny ostatní špatně.

Algoritmus 2

Další algoritmus, který může expert zvolit v: „*Hlavní menu → Nastavení → Způsob výpočtu umí →* ***Algoritmus 2***“ zohledňuje obtížnost odpovědět na otázku správně. Do výpočtu je vnesena skutečnost, že se může stát, že student i bez potřebných znalostí odpověď uhodne. První co tento algoritmus udělá, že převede podle následující tabulky počet odpovědí u aktuální otázky na proměnnou obtížnost (`int obtiznost`) podle *tabulky 6.5.*

Pro výpočet obtížnosti jsem použil vzorec:

$$obtížnost = 1 - pst \qquad \ldots\textbf{.vzorec č. 6.3.}$$

pst... je pravděpodobnost toho, že student odpověď na otázku uhodne

Po zjištění obtížnosti vypočítá novou hodnotu *umi* podle následujících vzorců:

Vzorec pro správnou odpověď:

$$Umi_{nové} = Umi_{staré} + \left[(1 - Umi_{staré}) * \left(\frac{vazba}{suma\_vazeb} \right) * obtížnost \right] \quad \textbf{…vzorec č. 6.4.}$$

Pro špatnou odpověď se nová hodnota počítá podle následujícího vzorce:

$$Umi_{nové} = Umi_{staré} - \left[Umi_{staré} * \left(\frac{vazba}{suma\_vazeb} \right) \right] \quad \textbf{…vzorec č. 6.5.}$$

$Umi_{nové}$... je nově vypočítaná hodnota pro kapitolu

$Umi_{staré}$... je hodnota *umi* před tímto výpočtem

vazba … je vazba položené otázky nastavená expertem na aktuálně počítanou kapitolu

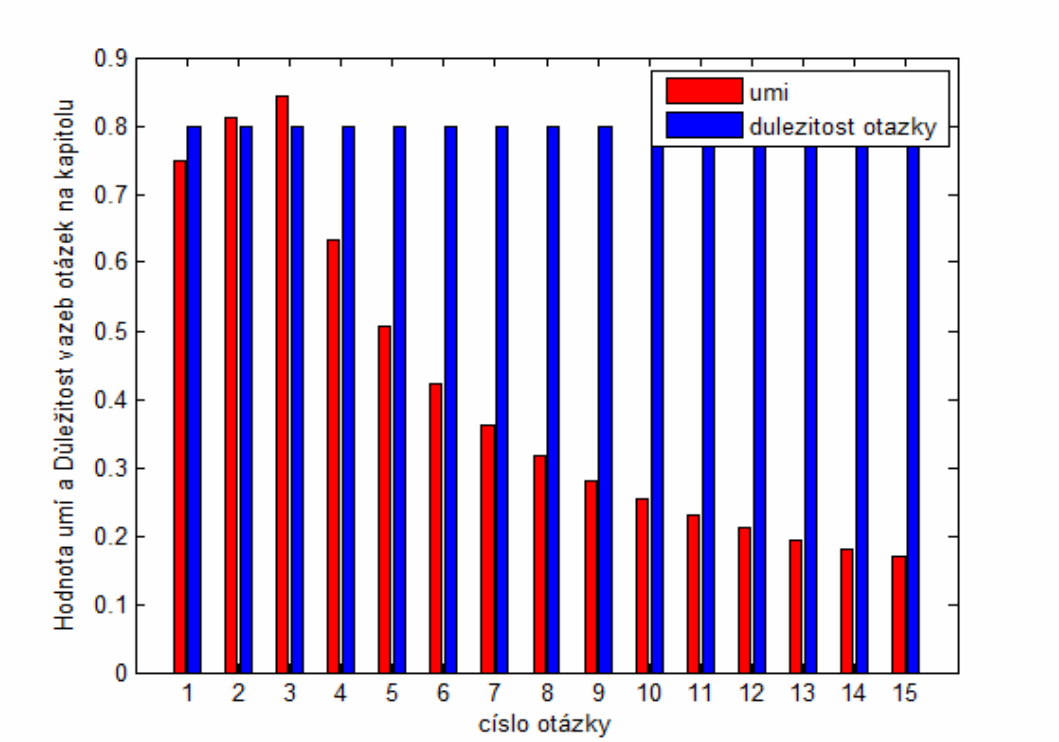
suma_vazeb … je součet vazeb na tuto kapitolu, které expert nastavil u všech do teď položených otázek.

obtížnost... udává míru obtížnosti otázky podle *vzorce 6.4.* a *tabulky 6.5*.

Tabulka 6.5.: Nastavení obtížnosti podle počtu odpovědí.

Počet odpovědí	Obtížnost
1	1
2	0,5
3	0,66
4	0,75
5	0,8
6	0,84
7	0,86

Expert vybere tuto možnost v případě, že chce při hodnocení zohlednit obtížnost vybrat správnou odpověď v závislosti na počtu možností. V případě, že student odpověď zadává (expert nastavil pouze jednu odpověď) se chová algoritmus stejně jako *algoritmus č.1*, protože je nastavena nulová pravděpodobnost toho, že by odpověď uhodl. Volit tento algoritmus je tedy zbytečné v případě, že test obsahuje pouze otázky, kde se vybere například pouze ze 3 odpovědí.



Obrázek 6.16.: Na grafu je vidět chování algoritmu č.2. v případě, že všechny otázky mají stejnou vazbu (důležitost otázky) na zkoušenou kapitolu. Obtížnost otázek je zde nastavena na 0,5 (dvě možné odpovědi), aby se co nejvíce projevil rozdíl mezi touto metodou výpočtu umí a metodou minulou.

Z grafu lze odečíst, že student odpověděl první tři otázky dobře a pak všechny špatně. Pokud je pro konzultaci použit tento algoritmus, výpočet je velice ovlivněn počtem možných odpovědí. Z *obrázku 6.16.* je vidět, že po třech správně odpověděných otázkách o dvou možnostech je zde stále uvažovaná celkem velká

pravděpodobnost toho, že student pouze náhodou vybírá správné odpovědi. Při špatné odpovědi zde není obtížnost započítána a výpočet je stejný jako u algoritmu č.1. Hodnota *umi* tedy klesá daleko rychleji, než roste při správné odpovědi. Rozdíl od algoritmu č.1. se projeví pouze při správně odpověděných otázkách. Pokud tedy student odpoví několikrát správně výsledná hodnota *umi* nebude 1 (100 %), ale bude k ní konvergovat. V případě špatných odpovědí se k hodnotě 0 (0 %) lze dostat hned po první otázce.

Algoritmus 3

Třetí algoritmus může expert nastavit pomocí volby: „*Hlavní menu → Nastavení → Způsob výpočtu umí →* ***Algoritmus 3***“. Tento algoritmus počítá hodnotu *umi* u kapitol podle následujících vzorců:

Vzorec pro správnou odpověď:

$$Umi_{nové} = Umi_{staré} + \left[(1 - Umi_{staré}) * \left(\frac{\text{vazba}}{100 * \text{odpovezeno}} \right) \right] \quad \ldots \textbf{\textit{vzorec č. 6.6.}}$$

Pro špatnou odpověď nová hodnota počítá podle následujícího vzorce:

$$Umi_{nové} = Umi_{staré} - \left[Umi_{staré} * \left(\frac{\text{vazba}}{100 * \text{odpovezeno}} \right) \right] \quad \ldots \textbf{\textit{vzorec č. 6.7.}}$$

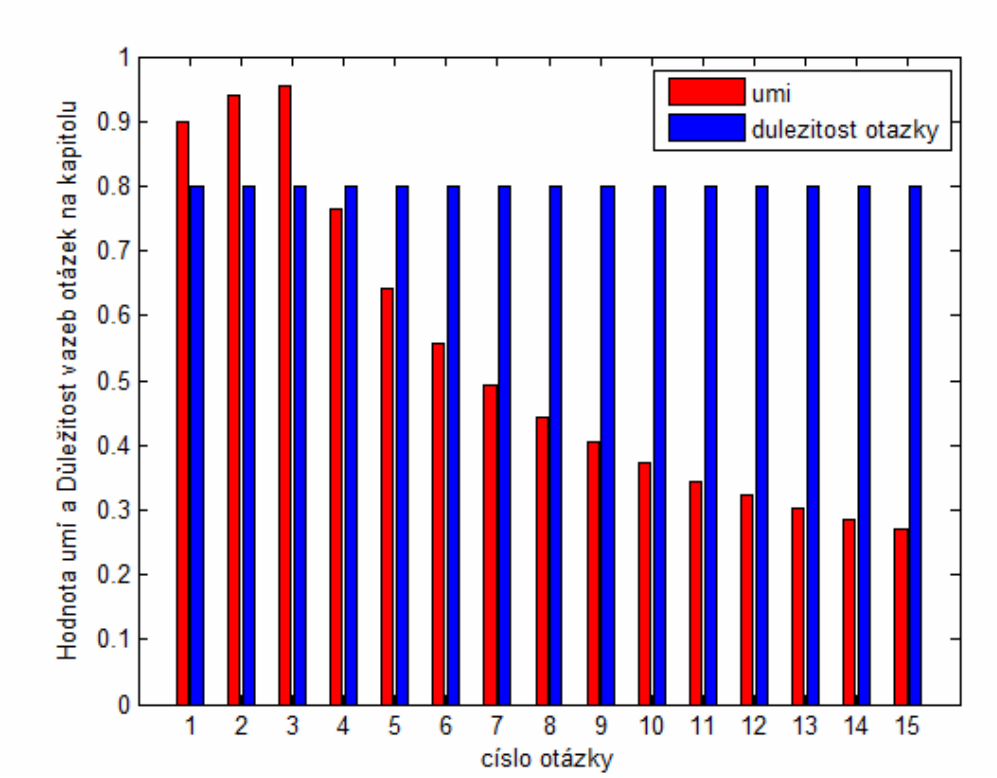
$Umi_{nové}$... je nově vypočítaná hodnota pro kapitolu

$Umi_{staré}$... je hodnota *umi* před tímto výpočtem

vazba … je vazba položené otázky nastavená expertem na aktuálně počítanou kapitolu

odpovezeno … celkový počet otázek odpověděných k této kapitole.

Na následujícím grafu je vidět jak se tento algoritmus chová v praxi.

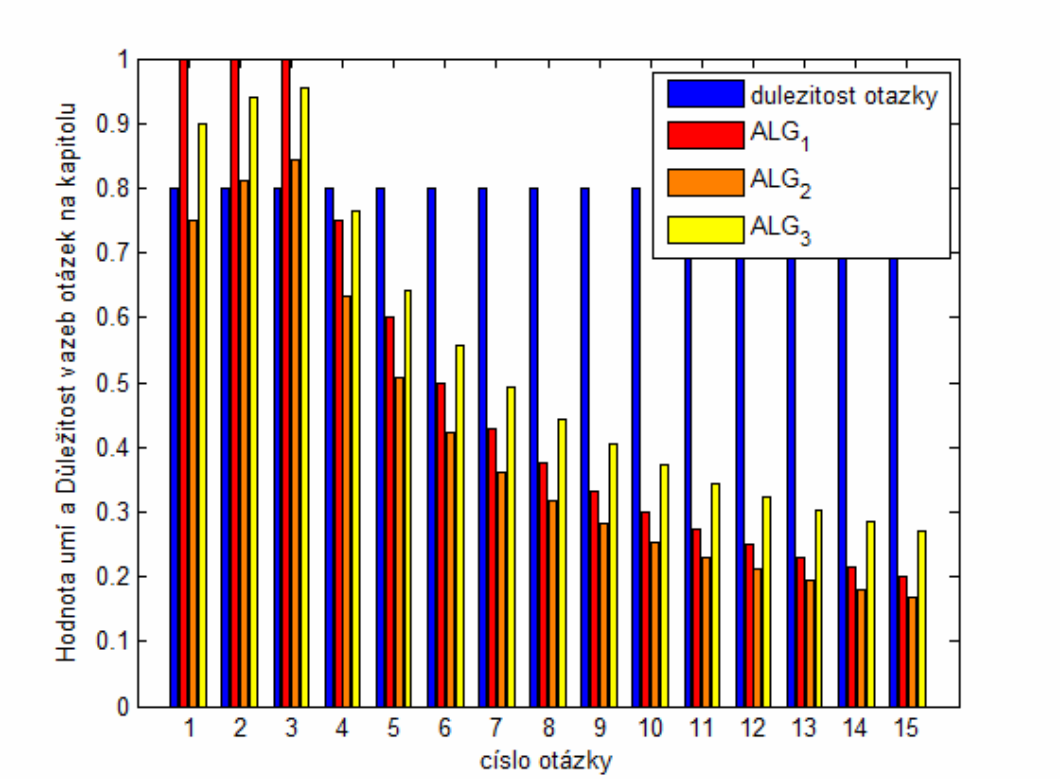


Obrázek 6.17.: Na grafu je vidět chování algoritmu č.3, v případě, že všechny otázky mají stejnou vazbu (důležitost otázky) na zkoušenou kapitolu.

Z grafu na *obrázku 6.17*. je vidět, že hodnota stejně jako u algoritmu č.2 konverguje k 1 (100 %) v případě správných odpovědí. Rychlost konvergence zde nejvíce ovlivňuje hodnota vazeb (důležitost otázky), která je stejně jako v minulých příkladech nastavená na 0,8 (80 %) u každé otázky. V případě, že chce expert vybrat tento algoritmus, měl by podle toho uvážit také důležitost jednotlivých vazeb. Jestliže nastaví všechny vazby kolem hodnot 10 až 20 % bude se hodnota *umi* měnit pouze nepatrně (v minulých případech záleželo více na poměru vazeb mezi jednotlivými otázkami).

Srovnání algoritmů pro výpočet umí

Na závěr je zde *obrázek 6.18*, kde je zobrazeno chování všech tří algoritmů při jedné konzultaci.



Obrázek 6.18.: V tomto grafu jsou sloučeny všechny tři předchozí verze do jednoho pro lepší srovnání toho, jak se který chová při stejných vstupních datech.

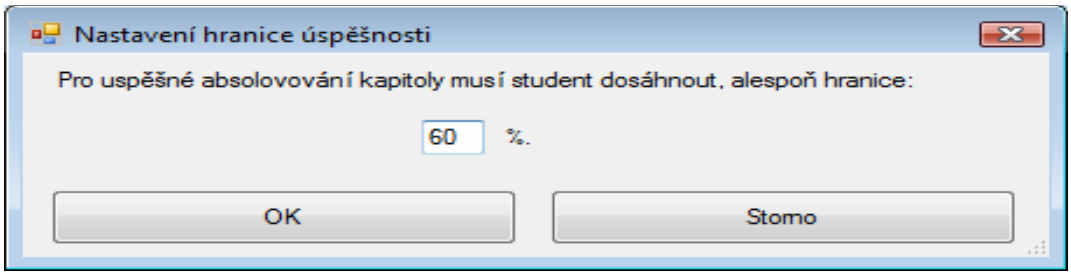
Z grafu je vidět rozdíl mezi jednotlivými algoritmy výpočtu *umi*. Vstupní data pro výpočet jsem ponechal stejná jako v předchozích případech. První algoritmus po třech správně odpověděných otázkách nepřipouští možnost, že student zkoušenou kapitolu neumí. Další dva algoritmy k hodnotě 1 (100%) pouze konvergují. U algoritmu č.2. konvergenci brzdí nastavení pouze dvou otázek, tedy pravděpodobnost 50 %, že student odpověď pouze odhadne. V případě správných odpovědí rychlost konvergence ovlivňuje hlavně počet možných odpovědí. Z *obrázku 6.18.* je dobře vidět také rozdíl mezi prvním a třetím algoritmem. První algoritmus vypočítá vždy přesnou hodnotu *umi* jakou si student zaslouží. Třetí algoritmus vždy pochybuje a k hodnotám 1 (100 %) nebo 0 (0 %) se blíží daleko pomaleji. Druhý algoritmus vždy nad studentovou správnou odpovědí pochybuje, ale při špatné odpovědi považuje za

jistotu, že student odpověď nevěděl. Díky tomu se k hodnotě 1 (100 %) blíží nejpomaleji a k hodnotě 0 (0 %) nejrychleji.

Protože je hlavním účelem programu odhalit případ, kdy student danou látku neumí, je doporučeno pro samostudium používat druhý algoritmus. V případě, že je program používán pro zkoušení studentů na známky je daleko vhodnější použít první algoritmus, který je vůči studentům spravedlivý. Třetí algoritmus je střední cestou mezi prvním a druhým.

6.5.3 Hranice úspěšnosti

Nastavení hranice úspěšnosti je poslední částí rozhodovacího mechanismu. Hranice úspěšnosti je číslo vyjadřující hodnotu v procentech, které pokud student při konzultaci dosáhne, bude u dané kapitoly program předpokládat, že ji umí. Tato hodnota lze nastavit v: „*Hlavní menu → Nastavení →Hranice úspěšnosti*“. Pokud tuto hodnotu expert při nastavování rozhodovacího mechanismu nezmění, hodnota je přednastavená na 75 %.



Obrázek 6.19.: Obrazovka nastavení hranice úspěšnosti.

6.6 VYSVĚTLOVACÍ MODUL A GENERÁTOR VÝSLEDKŮ

Poslední částí programu je textový soubor, který program vygeneruje na základě studentem provedené konzultace. Tento soubor lze uložit ve formátu „*.rtf“ pomocí tlačítka *Uložit výsledky testu*. Soubor se skládá z několika částí.

První částí souboru je jméno a ID studenta.

Další část je graf, který vykreslí úspěšnost studenta ve zkoušených kapitolách. Úspěšnost je vyjádřena v procentech od 0 do 100 % a barevně. V případě úspěšnosti 0 % je kapitola do grafu zakreslena červenou barvou. Pokud dosáhne student 100 % úspěšnosti, kapitola se vykreslí zeleně. Při ostatních výsledcích dochází k míchání těchto barev, proto při 50 % úspěšnosti vykreslí program kapitolu žlutě. Kromě procentní úspěšnosti *umi* u jednotlivých kapitol, je do grafu vepsán počet otázek položený k jednotlivým kapitolám a počet správně odpověděných.

V další části jsou vypsány všechny otázky, které byly položené při konzultaci. Otázky jsou vypsány ve stejném pořadí, jak byly položeny. Jestliže student otázku odpověděl správně je vypsána zeleně. V opačném případě je otázka vypsána červeně. Pod každou otázkou jsou vypsány odpovědi, které mohl student zvolit. Správná odpověď je zde zaznamenána vždy zeleně. Pokud student zvolil špatnou odpověď program ji označí červeně. Ostatní odpovědi jsou vypsány černě. V případě, že student odpověď zadával a zadal ji špatně, bude zde vypsána kromě správné odpovědi zeleně i jeho chybná odpověď. Pod každou otázkou je výčet čísel kapitol, ke kterým se tato otázka vztahovala a u každého čísla kapitoly je uvedena vazba nastavená expertem na tuto otázku od 0 do 99. Pokud student u otázky neuspěl, může si podle čísel kapitol pod otázkou najít odůvodnění správné odpovědi v patřičných teoretických textech. V případě, že student nestihne vybrat odpověď v časovém limitu, otázka se mu zobrazí červeně, ale výčet odpovědí neosahuje žádnou červenou odpověď. Jestliže student odpověď zadává, je zde vypsána informace o tom, že vypršel čas.

Po výčtu otázek je zde vypsáno kolik procent času z maximálního možného student využil. Dále pak, který algoritmus byl pro tuto konzultaci použit pro výběr otázek a pro výpočet hodnot *umi*. Poslední položkou této části je hodnota nastavená jako hranice úspěšnosti.

Nakonec je zde výčet teoretických textů kapitol, ve kterých se student v hodnocení nedostal na nastavenou hranici. Pokud tuto hranici expert nenastavil v položce „*Hlavní menu → Nastavení → Hranice úspěšnosti*“, standardně je nastavena na 75 %. Tyto teoretické texty studentovi slouží jako doporučení pro další studium. Ukázky souboru, který program generuje se nacházejí v *příloze č.4.*

6.7 NÁPADY PRO DALŠÍ VERZE PROGRAMU

Vytvořený program sice splňuje zadání, přesto je zde ještě spousta možností jak tento program rozšířit a vylepšit. Do další verze lze přidat:

- Automatické generování příkladů na základě expertem definovaného vzorce.
- Za velice důležité považuji, aby expert dostával zpětnou vazbu od studentů. K tomu by bylo vhodné, aby se vygenerované výsledky o provedené konzultaci odesílaly automaticky přes internet nebo ukládaly do databáze. Expert by pak na základě nejčastějších chyb mohl rozšiřovat nebo upravovat teoretický text do potřebného rozsahu.
- Pokud by se výsledky studentů ukládali do databáze, pro studenty by bylo jistě zajímavé srovnání s ostatními studenty, kteří si zkoušeli stejný test.
- Dále by bylo vhodné, aby studenti mohli hodnotit obtížnost testu, což by znovu dávalo expertovi určitou zpětnou vazbu.
- V případě, že student odpověď zadává se často může stát, že student odpoví dobře, ale program odpověď vyhodnotí jako špatnou, protože se text odpovědi přesně neshoduje s odpovědí, kterou otázce přiřadil expert. Pokud by se data z konzultací ukládaly do databáze, expert by mohl označit i ostatní správné odpovědi za správné. Studenti by pak mohli odpovědět jakoukoliv odpovědí z množiny správných.

Z uvedeného seznamu možností dalšího vývoje expertního systému vyplývá, že by bylo vhodnější program udělat jako internetovou aplikaci.

6.8 KONZULTACE S EXPERTEM

Před dokončením expertního systému jsem byl na konzultaci na ústavu fyziky, kde působí RNDr. Naděžda Uhdeová, Ph.D a RNDr. Eva Hradilová.

RNDr. Eva Hradilová se v roce 2004 účastnila projektu „Interaktivní multimediální www aplikace pro výuku fyziky“. Dalšího projektu, kterého se účastnila RNDr. Eva Hradilová se účastnila i RNDr. Naděžda Uhdeová, Ph.D a jmenoval se „Multimediální podpora teoretických cvičení z fyziky“. Společně se také

v roce 1997 podílely na projektu ohledně optimalizace přijímacího řízení na vysoké školy inženýrského typu v návaznosti na středoškolské studium. Jak RNDr. Eva Hradilová, tak RNDr. Naděžda Uhdeová, Ph.D mají několikaleté zkušenosti nejen s výukou, ale i s programy pro podporu výuky.

Potom, co jsem oběma expertkám představil tehdejší verzi programu, jsem dostal následující připomínky:

- První připomínka se týkala konzultace, kdy student zadává odpověď. Verze, kterou jsem na konzultaci měl připravenou, neobsahovala nápovědu o počtu znaků, které má student zadat. Po této konzultaci jsem tedy do pole, kde student zadává odpověď přidal otazníky, které vždy odpovídají počtu znaků odpovědi. Navíc jsem omezil počet znaků, které je možné do tohoto textového pole zadat na počet znaků správné odpovědi.
- Další požadavek na program se týkal zadávání vzorců. Vzorce by mělo jít zadávat kromě teoretického textu, také do otázky a v ideálním případě i do odpovědi. Navíc by bylo vhodné, aby program vyhodnocoval odpověď typu „*m*v*“ nebo „*m.v*“ stejně jako „*v*m*“ nebo „*v.m*“. Z části tento problém může vyřešit expert tím, že do otázky napíše poznámku typu: „*V odpovědi pište součin hvězdičkou!*“. Druhý problém s prohozením písmen ve vzorci již program řešit nedokáže a odpověď pak vyhodnotí jako špatnou.
- Další doporučení se týkalo konzultace, kdy student dostane možné odpovědi vždy v pořadí, v jakém je expert zadal. Požadavkem bylo, aby se odpovědi náhodně prohazovaly. Následně jsme se, ale shodli na tom, že tento test slouží pro jednorázové použití. Po dokončení prvního testu si student může správné odpovědi uložit a pokud bude chtít podvádět při opakování stejné konzultace, aby dosáhl lepších výsledků, bude sám proti sobě.

Obě expertky naopak ocenily:

- 10 různých algoritmů pro výběr otázek.
- 3 různé algoritmy pro hodnocení odpovědí studenta. *Algoritmus č.2* nejprve označily za špatný, protože při hodnocení správných odpovědí počítá s tím, že by student mohl bez jakýchkoliv znalostí odpověď uhodnout. Nakonec

jsme se shodli na tom, že hlavní účel programu je zjistit studentovy nedostatky a tento algoritmus tomu dodává určitou redundanci.

- Další funkce, kterou ocenily, byla možnost zadání vazeb mezi kapitolami a otázkami ve velkém rozsahu 0 až 99.

7. NÁVRH BÁZE ZNALOSTÍ

Posledním úkolem zadání diplomové práce bylo navrhnout jednoduchou bázi znalostí a na ní demonstrovat funkci programu. Báze znalostí byla vytvořena na zkoušení komplexních čísel.

7.1 POTŘEBNÉ TEORETICKÉ ZNALOSTI

V první fázi návrhu lze komplexní čísla rozdělit do šesti kapitol:

1. Algebraický tvar
2. Absolutní hodnota
3. Grafické znázornění
4. Goniometrický tvar
5. Moivreova věta
6. Exponenciální tvar

Jednotlivé kapitoly spolu navzájem úzce souvisí, což je vhodné pro demonstrování nastavení vazeb mezi otázkou a kapitolami. Ke každé z uvedených kapitol byl vytvořen krátký teoretický text.

7.2 OTÁZKY A NASTAVENÍ VAZEB

K uvedeným kapitolám na komplexní čísla bylo navrhnuto celkem 40 otázek. Některé otázky se vztahují k více kapitolám. Soubor otázek je uveden v příloze č.1. V následujících *tabulkách 7.1.* a *7.2.* se nachází konečné nastavení vazeb.

Tabulka 7.1.: Nastavení vazeb mezi otázkami a kapitolami v bázi znalostí pro zkoušení komplexních čísel. Pro otázky č.1 – 20.

kapitola \ otázka	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1.Algebraický tvar	20	20	30	60	80	80	60	20	70	50	50	30	30	60	30	0	0	0	90	0
2.Absolutní hodnota	0	0	0	0	0	0	50	50	0	0	0	0	0	0	0	50	0	0	0	30
3.Grafické znázornění	0	0	0	0	0	0	0	0	0	0	0	0	0	90	70	50	30	30	0	0
4.Goniometrický tvar	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50
5.Moivreova věta	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6.Exponenciální tvar	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Tabulka 7.2.: Nastavení vazeb mezi otázkami a kapitolami v bázi znalostí pro zkoušení komplexních čísel. Pro otázky č.21. – 40.

kapitola \ otázka	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1.Algebraický tvar	0	0	0	0	0	70	70	30	0	20	0	0	0	0	0	0	0	0	0	40
2.Absolutní hodnota	30	40	40	0	0	0	0	40	0	0	0	30	0	50	0	60	0	70	0	0
3.Grafické znázornění	0	0	0	0	0	10	10	0	0	0	0	0	0	20	40	60	0	0	0	0
4.Goniometrický tvar	50	50	50	60	70	50	50	50	50	0	40	50	40	0	0	0	40	30	40	40
5.Moivreova věta	0	0	0	0	0	0	0	0	0	0	70	60	40	0	0	0	80	0	80	0
6.Exponenciální tvar	0	0	0	0	0	0	0	0	50	80	0	0	0	50	50	50	60	40	50	40

7.3 TESTOVÁNÍ VYTVOŘENÉ BÁZE ZNALOSTÍ

Po nastavení všech vazeb byl prozatím u všech otázek nastavil čas na 300 s. Čas byl nastaven úmyslně tak velký, aby při testování měl každý možnost odpovědět

správně. Podle časů studentů co odpověděli správně, byly nastaveny časy do konečné verze báze znalostí.

Rozhodovací mechanismus byl nastaven:

- Algoritmus výpočtu umí: algoritmus č.1.
- Způsob výběru otázky: Všechny podle pořadí v seznamu.
- Hranice úspěšnosti: 60 %

Pro testování vytvořené báze znalostí byla předložena 10-ti studentům, kteří se pokusili dosáhnout co nejlepších výsledků v co nejkratším čase:

Tabulka 7.3.: Výsledky zkoušení studentů z otázek č.1 - 20. U každého studenta je zde uvedeno, zda odpověděl dobře (**o**) nebo špatně(**x**) pro každou otázku. V případě správné odpovědi je uveden čas v [s].

student \ otázka	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Petr Svoboda	o	o	o	x	x	o	x	x	x	o	x	o	o	o	o	x	o	o	x	x
	26	49	79			14				17		10	18	44	25		17	10		
Jaroslav Horák	o	o	x	o	o	o	o	o	x	o	o	o	x	o	o	o	o	o	o	x
	10	18		59	23	16	75	23		18	5	7		29	24	19	12	6	17	
Jan Neužil	x	o	o	x	o	o	x	o	x	o	o	o	o	o	o	o	o	o	x	o
		20	13		72	35		23		17	80	17	12	94	33	19	13	3		68
Kateřina Bittnerová	o	o	x	o	o	x	o	o	x	o	o	o	o	o	o	o	o	o	o	o
	26	16		132	28		45	25		9	4	9	10	23	18	27	6	3	19	51
Jakub Navařík	o	o	o	o	o	x	o	o	o	o	o	o	o	o	o	o	o	o	x	o
	17	11	24	109	27		85	27	103	9	20	8	18	32	38	78	15	3		44
Radek Baránek	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	x	o
	31	18	24	58	38	12	78	28	79	7	8	7	10	29	33	27	7	4		12
Petr Novák	o	o	o	o	o	x	x	o	x	o	o	o	o	o	o	o	o	o	o	o
	35	24	36	75	90			63		28	27	10	28	28	87	42	40	10	71	48
Marek Matiáš	o	o	o	o	x	x	x	x	o	o	o	x	o	x	x	x	x	x	o	o
	33	12	40	49					84	20	8		46						47	79
Veronika Šeděnková	o	o	o	o	o	o	o	x	o	o	x	o	x	o	o	o	o	o	o	o
	36	15	81	152	44	32	84		60	6		12		144	38	112	13	5	22	26
Martina Míková	o	o	o	o	o	o	x	o	o	o	o	o	o	o	x	o	o	o	o	o
	23	18	17	222	9	19		83	79	7	10	12	12	31		60	46	7	39	35
dobře [-]	**9**	**10**	**8**	**8**	**8**	**6**	**5**	**7**	**5**	**10**	**8**	**9**	**8**	**9**	**8**	**8**	**8**	**9**	**6**	**8**
čas [s]	**26**	**20**	**39**	**107**	**41**	**21**	**73**	**38**	**81**	**13**	**20**	**10**	**19**	**50**	**37**	**48**	**21**	**5**	**35**	**45**

Tabulka 7.4.: Výsledky zkoušení studentů z otázek č.21 - 40. U každého studenta je zde uvedeno, zda odpověděl dobře (**o**) nebo špatně(**x**) pro každou otázku. V případě správné odpovědi je uveden čas v [s].

student \ otázka	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**
Petr Svoboda	x	x	x	o	x	x	o	x	x	x	x	x	x	x	o	x	x	x	o	o
				42			22								15				9	26
Jaroslav Horák	o	x	o	x	o	o	o	x	x	o	o	o	x	x	o	x	x	x	o	o
	40		61		47	44	12			27	44	10			9				35	11
Jan Neužil	o	x	x	o	o	o	o	o	o	o	o	o	x	o	x	x	o	o	o	o
	21			174	14	24	17	99	39	111	119	163		66			40	31	8	13
Kateřina Bittnerová	o	o	o	o	o	o	o	o	x	x	o	o	x	x	x	x	x	x	x	o
	5	21	17	67	17	53	11	51			43	7								9
Jakub Navařík	o	x	x	x	x	o	o	o	o	o	o	o	x	o	o	o	o	o	o	o
	5					25	10	54	33	21	73	21		25	21	16	28	21	9	9
Radek Baránek	o	o	o	x	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
	6	13	28		18	21	13	35	31	9	22	5	16	98	9	16	31	38	14	6
Petr Novák	x	o	x	o	o	x	x	x	x	x	o	o	x	x	x	o	o	o	x	o
		118		42	42						53	49				17	114	70		10
Marek Matiáš	o	o	o	x	x	x	x	o	x	x	o	o	x	x	x	x	x	x	x	o
	28	67	42					96			36	14								15
Veronika Šeděnková	o	o	x	x	x	x	x	o	o	o	x	o	x	x	o	x	x	x	o	o
	11	113						37	21	15		9			10				26	8
Martina Míková	o	o	o	o	o	o	x	o	o	x	o	o	o	o	o	o	o	o	o	o
	6	96	9	29	13	103		212	135		25	13	49	30	16	32	84	46	64	14
dobře [-]	**8**	**6**	**5**	**5**	**6**	**6**	**6**	**7**	**5**	**5**	**8**	**9**	**2**	**4**	**6**	**4**	**5**	**5**	**7**	**10**
čas [s]	**15**	**71**	**31**	**70**	**25**	**45**	**14**	**83**	**51**	**36**	**51**	**32**	**32**	**54**	**13**	**20**	**59**	**41**	**23**	**12**

Průměrné časy studentů z *tabulek 7.3 a 7.4* byly nakonec použity do výsledné báze znalostí. Vzhledem k tomu, že 9 z 10 studentů, kteří zde jsou uvedeni, mají již maturitu za sebou a laděná báze znalostí by měla sloužit hlavně pro testování studentů středních škol, bylo připočteno ke každému z časů 10 s. Žádný ze studentů nebyl na toto zkoušení připravený a prakticky všichni si jednotlivá pravidla pro

počítání komplexních čísel odvozovali nebo vzpomínali až při testu, takže by připravení studenti středních škol neměli mít problém s řešením v navrhnutém čase.

7.4 VYHODNOCENÍ TESTOVÁNÍ

Na závěr je zde přiložena *tabulka 7.5* s dosaženými výsledky každého studenta v jednotlivých kapitolách.

Tabulka 7.5.: Výsledky zkoušení studentů rozdělené podle kapitol. Vzhledem k tomu, že jsem hranici pro úspěšné absolvování testu nastavil na 60 %, výsledky pod 60 % jsou označeny červeně a nad 60 % zeleně.

student\kapitola	1 [%]	2 [%]	3 [%]	4 [%]	5 [%]	6 [%]	celek [%]
Petr Svoboda	46	0	66	23	24	30	31,5
Jaroslav Horák	84	42	80	54	64	47	61,8
Jan Neužil	70	59	76	83	88	79	75,8
Kateřina Bittnerová	80	67	71	75	39	9	73,2[16]
Jakub Navařík	83	85	100	67	88	100	87,1
Radek Baránek	91	100	100	93	100	100	97,3
Petr Novák	60	61	80	53	64	40	59,6
Marek Matiáš	49	39	0	47	39	9	30,5
Veronika Šeděnková	76	50	76	47	42	57	58,0
Martina Míková	82	91	80	94	100	83	88,3

Studenty, kteří se testování zúčastnili, nelze jednoduše srovnávat, protože každý z nich studuje něco jiného. Studenti Jaroslav Horák, Jan Neužil, Jakub Navařik, Radek Baránek a Martina Miková, kteří dopadli nejlépe, studují na technických vysokých školách a komplexní čísla ve škole používali daleko více, než ostatní. Kateřina Bittnerová je jediná studentka střední školy a ve škole se zatím

[16] Do celkového hodnocení nejsou započítány výsledky kapitol č.5. a č.6., protože Kateřina Bittnerová tyto dvě kapitoly ve škole neprobírala.

neučila kapitolu č.5. Moivrova věta a č.6. Exponenciální tvar komplexního čísla. V kapitolách, které uměla si vedla velice dobře. Studenti Petr Novák, Veronika Šeděnková, Marek Matiáš a Petr Svoboda nedosáhli hranice 60 %. Zatímco Veronika Šeděnková a Petr Novák ji nedosáhli pouze těsně a z každé kapitoly dokázaly vypočítat alespoň některé příklady. Marek Matiáš a Petr Svoboda si marně vzpomínali na to, co je to absolutní hodnota nebo jak se zakreslují komplexní čísla v Gaussově rovině. Tito 4 studenti již 2 až 3 roky matematiku nepoužívají.

Pro nastavení báze znalostí bylo velice důležité, že všichni studenti co se tohoto testu zúčastnili, uznali, že dosažené výsledky odpovídají jejich znalostem.

8. ZÁVĚR

Prvním úkolem diplomové práce bylo seznámit se s problematikou expertních systémů a zpracovat literární rešerši ohledně expertních systémů. V textu jsem popsal princip expertních systémů a hlavně jsem se zaměřil na architekturu expertních systémů.

Další část práce obsahuje literární rešerši na téma reprezentace znalostí pro expertní systémy. Uvedl jsem zde čtyři nejpoužívanější způsoby reprezentace znalostí, které se používají pro expertní systémy. A to pomocí predikátové logiky, produkčních pravidel, sémantických sítí a rámců. Ke každému uvedenému způsobu reprezentace znalostí jsem uvedl krátký konkrétní příklad pro reálnou představu, jak vlastně daná reprezentace znalostí pracuje.

Oproti ostatním způsobům reprezentace znalostí jsem se více zaměřil na reprezentaci znalostí pomocí rámců a uvedl jsem delší příklad reprezentace znalostí *bytu*. Z části jsem do této kapitoly přidal příklad rámce *vozidla*, který není zpracován dopodrobna, ale pouze části potřebné pro účely vysvětlení základních pojmů používaných pro popis rámců.

Hlavním cílem této diplomové práce bylo vytvoření a odladění expertního systému pro samostatné domácí studium s automatickým vyhodnocením úspěšnosti a doporučením pro další studium. Tento expertní systém má využívat rámcovou reprezentaci znalostí. Tato část práce obsahuje popis ovládání a všech funkcí tohoto expertního systému. Důkladně je zde popsán způsob ukládání znalostí pomocí rámců a všechny možnosti nastavení rozhodovacího mechanismu. Tento expertní systém jsem naprogramoval v prostředí Microsoft Visual Studio a použil jsem programovací jazyk C#. Zdrojový kód programu včetně spustitelného exe souboru se nachází v příloze č.2. Po dokončení tohoto expertního systému jsem ve spolupráci s expertkami na výuku fyziky navrhl možnosti rozšíření programu pro další verze.

Vzhledem k tomu, že dosavadním výsledkem mojí práce byl prázdný expertní systém, navrhl jsem a v tomto expertním systému vytvořil jednoduchou bázi znalostí. Vytvořená báze znalostí obsahuje 40 otázek, které se týkají komplexních čísel. Komplexní čísla jsem vybral, protože je lze rozdělit na několik kapitol (algebraický

tvar, absolutní hodnota, grafické znázornění, goniometrický tvar, Moivreova věta a exponenciální tvar), které se navzájem prolínají. Díky tomu lze dobře využít možnosti vyzkoušet více kapitol pomocí jedné otázky. Na závěr jsem tuto bázi znalostí testoval na 10-ti studentech. Výsledky testů se nachází v příloze č.4.

9. SEZNAM POUŽITÉ LITERATURY

[1] MAŘÍK V.,ŠTĚPÁNKOVÁ O., LAŽANSKÝ J., a kol.: *Umělá inteligence (1).* Akademia, Praha 1993, ISBN 80-200-0496-3

[2] MAŘÍK V.,ŠTĚPÁNKOVÁ O., LAŽANSKÝ J., a kol.: *Umělá inteligence (2).* Akademia, Praha 1997, ISBN 80-200-0504-8

[3] POPPER M.,KELEMEN J.,: *Expertné systémy*. Alfa, Bratislava 1988, ISBN 80-05-00051-0

[4] GIARRATANO J., RILEY G.,: *Expert Systems: Principles and Programming.* PSW Publisching Company, Boston 1994, ISBN 0-534-95053-1

[5] PROVAZNÍK I., BARDOŇOVÁ J.,: Expertní systémy – praktická cvičení. Vysoké učení technické v Brně, Brno 2000, ISBN 80-214-1768-4

[6] HLAVÁČ V.,: *Kognitivní robotika,* <http://cmp.felk.cvut.cz/~hlavac/Public/TeachingLectures/KognitivniRobotika.pdf>

[7] HAYES-ROTH F., WATERMAN A. D., LENAT B.D.: *Building expert systems*, Boston, Addison-Wesley Longman Publishing, 1983, ISBN:0-201-10686-8

[8] *Deterministický algoritmus – Wikipedie otevřená encyklopedie* ,[5.5. 2008], <http://cs.wikipedia.org/wiki/Deterministick%C3%BD_algoritmus>

[9] *Seznam encyklopedie – usuzování,* [5.5. 2008], <http://encyklopedie.seznam.cz/heslo/106470-usuzovani>

[10] *Kauzalita - Wikipedie otevřená encyklopedie* ,[5.5. 2008], <http://cs.wikipedia.org/wiki/Kauzalita>

[11] *Generalizace - Wikipedie otevřená encyklopedie* ,[5.5. 2008], <http://cs.wikipedia.org/wiki/Generalizace>

[12] *Heuristika - Wikipedie otevřená encyklopedie* ,[5.5. 2008], <http://cs.wikipedia.org/wiki/Heuristika>

[13] Pavlovská H.,:*Kontextové vyhledávání pro knowledge management,* VŠE Praha, 2002, <http://nb.vse.cz/~palovska/PhDthesis.pdf>

[14] *Entita - Wikipedie otevřená encyklopedie* ,[5.5. 2008], <http://cs.wikipedia.org/wiki/Entita>

[15] ANDREW D. MCGETTRICK, *The definitions of programimg languages*, Cambridge University Press, New York, 1980, ISBN 0521226317

[16] *Inkluze, inkluse – ABC.cz: slovník cizích slov*,[5.5. 2008], <http://slovnik-cizich-slov.abz.cz/web.php/slovo/inkluze-inkluse>

[17] EFRAIM TURBAN, JAY E. ARONSON, *Decision support systéme and inteligent systems*. Sixth edition, New Jersey, ISBN 0-13-08946596

[18] RONALD J.BRACHMAN, HECTOR J.LEVESQUE, *Knowledge representation and reasoning*. Morgan Kaufman publisher, San Francisco, ISBN-13:978-1-55860-932-7

[19] CHAUDHRI V.K., J.D. LOWRANCE, *Generic Knowledge-Base Editor*. <www.ai.sri.com/~gkb/welcome.shtml>

[20] JACKSON P., *Introduction to expert systems*. Reading, 1998, ISBN 0-07-909785-5

[21] *The design space of frame knowledge representation*,[9.12. 2008], <http://www.cs.umbc.edu/771/papers/karp-freview.pdf>

[22] *A frame-based peripheral nervous expert system*,[8.10.2008], <http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel5/7705/21095/00978760.pdf?temp=x>

[23] *Extensible Markup Language*,[11.12.2008], <http://cs.wikipedia.org/wiki/Extensible_Markup_Language>

10. SEZNAM PŘÍLOH

1. Seznam otázek pro navrhnutou bázi znalostí.
2. Zdrojový kód vytvořeného expertního systému.
3. Vytvořená báze znalostí na komplexní čísla.
4. Výsledky z testování vytvořené báze znalostí.

Příloha 1.: Seznam otázek pro navrhnutou bázi znalostí.

1. Součet dvou komplexních čísel $z_1 = 2 + i$ a $z_2 = 1 + 3i$ je:

- o *3+6i*
- o *3+5i*
- • *3+4i*
- o *4+4i*
- o *4+3i*
- o *4+5i*
- o *2+3i*

Otázka ke kapitolám: 1(20).

2. Součet dvou komplexních čísel $z_1 = 2 - i$ a $z_2 = 1 + 4i$ je:

- o *3+6i*
- o *3+5i*
- o *3+4i*
- o *4+4i*
- o *4+3i*
- o *4+5i*
- • *3+3i*

Otázka ke kapitolám: 1(20).

3. Máme dvě komplexní čísla: z1 = 1 - i a z2 = 2 + 2i. Vypočítejte komplexní číslo: $z_3 = z_1 - z_2$. z_3 = ?

- o *3+6i*
- o *-3-5i*
- o *-2-4i*
- • *-1-3i*
- o *1+3i*
- o *2-5i*
- o *3-3i*

Otázka ke kapitolám: 1(30).

4. Máme dvě komplexní čísla: $z_1 = 2 - 2i$ a $z_2 = 2 + 2i$. Vypočítejte komplexní číslo: $z_3 = z_1 * z_2$. z_3 = ?

- • *8*

Otázka ke kapitolám: 1(60).

5. i^5 = ?

o *i*i*i*i*
• *i*
o *-i*
o *-1*
o *1*

Otázka ke kapitolám: 1(80).

6. i^{32} = ?

o *2*
o *i*
o *-i*
o *-1*
• *1*

Otázka ke kapitolám: 1(80).

7. Máme dvě komplexní čísla: $z_1 = 2 + 2i$ a $z_2 = 2 + 2i$. Vypočítejte absolutní hodnotu: $|z_1 * z_2|$.

o *4*
o *6*
o *2*
• *8*
o *16*
o *32*
o *1*

Otázka ke kapitolám: 1(60), 2(50).

8. Máme dvě komplexní čísla: $z_1 = 1 + 2i$ a $z_2 = 2 + 2i$. Vypočítejte absolutní hodnotu: $|z_1 + z_2|$.

o *4*
o *6*
o *2*
o *3*
• *5*
o *7*
o *1*

Otázka ke kapitolám: 1(20), 2(50).

9. Máme dvě komplexní čísla: $z_1 = 1 + 2i$ a $z_2 = i$. Vypočítejte komplexní číslo $z = z_1 : z_2$.

o *4-i*

o *6-2i*
• *2-i*
o *3-7i*
o *5-i*
o *7-i*
o *1*

Otázka ke kapitolám: 1(70), 2(50).

10. Které číslo je komplexně sdružené k: z = 1 + 2i ?

o *1 + 2i*
• *1 - 2i*
o *-1 + 2i*
o *-1 - 2i*
o *žádná z předchozích možností*

Otázka ke kapitolám: 1(50).

11. Opačné komplexní číslo k číslu: z = 1 + 2i ?

o *1 + 2i*
o *1 - 2i*
o *-1 + 2i*
• *-1 - 2i*
o *žádná z předchozích možností*

Otázka ke kapitolám: 1(50).

12. Napište reálnou část komplexního čísla: z = 2i ?

• *0*

Otázka ke kapitolám: 1(30).

13. Napište číslicí imaginární část komplexního čísla: z = i^2 ?

• *0*

Otázka ke kapitolám: 1(30).

14. Které z uvedených komplexních čísel má základní argument 315° ?

o *2 + 2i*
• *2 - 2i*
o *-2 + 2i*
o *-2 - 2i*
o *žádná z předchozích možností*

Otázka ke kapitolám: 1(60), 3(90).

15. Které z uvedených komplexních čísel leží nejblíže k imaginární ose při znázornění do Gaussovy roviny?

- *1 + 2i*
- *2 - 2i*
- *-3 + 2i*
- *-2 - 2i*
- *všechny leží od imaginární osy stejně daleko*

Otázka ke kapitolám: 1(30), 3(70).

16. Vypočítejte absolutní hodnotu komplexního čísla: z = 3 + 4i ? a určete vzdálenost tohoto komplexního čísla od počátku ?

- *absolutní hodnota je 5 a vzdálenost od počátku je 25*
- *absolutní hodnota je 25 a vzdálenost od počátku je 5*
- *absolutní hodnota je 25 a vzdálenost od počátku je 25*
- *absolutní hodnota je 5 a vzdálenost od počátku je 5*
- *žádná z předchozích možností*

Otázka ke kapitolám: 2(50), 3(50).

17. Na kterou osu se v Gaussově rovině vynáší imaginární část komplexního čísla?

- *x*
- *y*
- *z*
- *žádná z předchozích možností*

Otázka ke kapitolám: 3(30).

18. Na kterou osu se v Gaussově rovině vynáší reálná část komplexního čísla?

- *x*
- *y*
- *z*
- *žádná z předchozích možností*

Otázka ke kapitolám: 3(30).

19. $1 - i^7 = ?$

- *2*
- *1+ i*
- *1- i*
- *-1*

o *1*
o *0*
o *-2*

Otázka ke kapitolám: 1(90).

20. Vypočítejte absolutní hodnotu komplexního čísla:

z = (cos (90°) + i*sin(90°))

o *0*
• *1*
o *2*
o *3*
o *90*
o *i*
o *žádná z předchozích možností*

Otázka ke kapitolám: 2(30), 4(50).

21. Vypočítejte absolutní hodnotu komplexního čísla:

z = 3*(cos (90°) + i*sin(90°))

o *0*
o *1*
o *2*
• *3*
o *90*
o *i*
o *žádná z předchozích možností*

Otázka ke kapitolám: 2(30), 4(50).

22. Vypočítejte absolutní hodnotu komplexního čísla: z =z_1/z_2 jestliže z_1 = 3*(cos (90°) + i*sin(90°)) a z_2 = 3*(cos (30°) + i*sin(30°))

o *0*
• *1*
o *2*
o *3*
o *90*
o *i*
o *žádná z předchozích možností*

Otázka ke kapitolám: 2(40), 4(50).

23. Vypočítejte absolutní hodnotu komplexního čísla: z =z_1*z_2 jestliže z_1 = 3*(cos (9°) + i*sin(9°)) a z_2 = 3*(cos (3°) + i*sin(3°))

o *0*
o *1*
o *6*
o *3*
• *9*
o *27*
o *žádná z předchozích možností*

Otázka ke kapitolám: 2(40), 4(50).

24. Vypočítejte komplexní číslo: z =z_1*z_2 jestliže

z_1 = 3*(cos (90°) + i*sin(90°)) a z_2 = 3*(cos (30°) + i*sin(30°))

o *z = 9*(cos (60°) + i*sin(60°))*
o *z = 9*(cos (90°) + i*sin(120°))*
o *z = 3*(cos (120°) + i*sin(90°))*
• *z = 9*(cos (120°) + i*sin(120°))*
o *z = 3*(cos (90°) + i*sin(90°))*
o *z = 9*(cos (120°) + i*sin(60°))*
o *žádná z předchozích možností není dobře*

Otázka ke kapitolám: 4(60).

25. Vypočítejte komplexní číslo: z =z_1/z_2 jestliže

z_1 = 3*(cos (90°) + i*sin(90°)) a z_2 = 3*(cos (30°) + i*sin(30°))

• *z = (cos (60°) + i*sin(60°))*
o *z = (cos (90°) + i*sin(120°))*
o *z = (cos (120°) + i*sin(90°))*
o *z = (cos (120°) + i*sin(120°))*
o *z = (cos (90°) + i*sin(90°))*
o *z = (cos (120°) + i*sin(60°))*
o *žádná z předchozích možností není dobře*

Otázka ke kapitolám: 4(70).

26. Převeďte komplexní číslo: z = 3*(cos (90°) + i*sin(90°)) na algebraický tvar a zadejte jeho imaginární část:

• *3*

Otázka ke kapitolám: 1(70), 3(10), 4(50).

27. Převeďte komplexní číslo: z = 4*(cos (90°) + i*sin(90°)) na algebraický tvar a zadejte jeho reálnou část:

• *0*

Otázka ke kapitolám: 1(70), 3(10), 4(50).

28. Převeďte komplexní číslo: z = (cos (30°) + i*sin(30°)) na algebraický tvar a vypočítejte jeho absolutní hodnotu:

- o *algebraický tvar je: 0,864 + 0,5i a absolutní hodnota je: 0,99*
- • *algebraický tvar je: 0,866 + 0,5i a absolutní hodnota je: 1*
- o *algebraický tvar je: 0,707 + 0,5i a absolutní hodnota je: 0,866*
- o *algebraický tvar je: 0,866 + 0,5i a absolutní hodnota je: 0,707*
- o *algebraický tvar je: 0,844 + 0,7i a absolutní hodnota je: 0,9*
- o *algebraický tvar je: 0,5 + 0,5i a absolutní hodnota je: 0*
- o *algebraický tvar je: 0,5 + 0,866i a absolutní hodnota je: 0,844*

Otázka ke kapitolám: 1(30), 2(40), 4(50).

29. Převeďte komplexní číslo: z = 3*($e^{i30°}$) na goniometrický tvar:

- o *z = 3*(cos (60°) + i*sin(30°))*
- o *z = 9*(cos (30°) + i*sin(30°))*
- o *z = 3*(cos (30°) + i*sin(45°))*
- o *z = 3*(cos (45°) + i*sin(30°))*
- o *z = 3*(cos (30°) + i*sin(90°))*
- o *z = 3*(cos (60°) + i*sin(30°))*
- • *z = 3*(cos (30°) + i*sin(30°))*

Otázka ke kapitolám: 4(50), 6(50).

30. Máme komplexní číslo v algebraickém tvaru: z = 1+i. Co bude v exponentu jestliže toto číslo převedeme na exponenciální tvar:

- o *i*0°*
- o *i*30°*
- • *i*45°*
- o *i*60°*
- o *i*90°*

Otázka ke kapitolám: 1(20), 6(80).

31. Vypočítejte absolutní hodnotu komplexního čísla: z^4, jestliže z = 3*(cos (90°) + i*sin(90°))

- o *124*
- • *81*
- o *49*
- o *90*
- o *72*
- o *80*
- o *žádná předchozí odpověď není dobře*

Otázka ke kapitolám: 4(40), 5(70).

32. Vypočítejte absolutní hodnotu komplexního čísla: z^{42}, jestliže $z = (\cos(90°) + i*\sin(90°))$

- *0*
- *2*
- *3*
- *1*
- *žádná předchozí odpověď není dobře*

Otázka ke kapitolám: 2(30), 4(50), 5(60).

33. Napište argument komplexního čísla: z^2, jestliže $z = 3*(\cos(10°) + i*\sin(10°))$

- *20*

Otázka ke kapitolám: 4(40), 5(40).

34.Jak daleko leží komplexní číslo: $z=z_1*z_2$ jestliže $z_1 = 2*(e^{i40°})$ a $z_2 = 3*(e^{i30°})$ od počátku v Gaussově rovině?

- *6*

Otázka ke kapitolám: 2(50), 3(20), 6(50).

35. Jak daleko leží komplexní číslo: $z_1 = 2*(e^{i90°})$ a od imaginární osy ?

- *0*
- *1*
- *2*
- *3*
- *4*
- *90*
- *žádná z předchozích možností*

Otázka ke kapitolám: 3(40), 6(50).

36. Jak daleko leží komplexní číslo: z^4 jestliže $z = 2*(e^{i40°})$ od počátku v Gaussově rovině?

- *16*

Otázka ke kapitolám: 2(60), 3(60), 6(50).

37. Převeďte komplexní číslo: z^4 jestliže $z = 2*(e^{i40°})$ do goniometrického tvaru:

- *$= 16*(\cos(160°) + i*\sin(120°))$*

- *= 16*(cos (120°) + i*sin(160°))*
- *= 16*(cos (160°) + i*sin(160°))*
- *= 8*(cos (160°) + i*sin(160°))*
- *= 32*(cos (160°) + i*sin(160°))*
- *= 2*(cos (160°) + i*sin(160°))*
- *= 16*(cos (120°) + i*sin(120°))*

Otázka ke kapitolám: 4(40), 5(80), 6(60).

38. Převeďte komplexní číslo: z^6 jestliže $z = 2*(e^{i10^\circ})$ do goniometrického tvaru a zadejte jeho vzdálenost od počátku v Gaussově rovině:

- *64*

Otázka ke kapitolám: 2(70), 4(30), 6(40).

39. Převeďte komplexní číslo: z^6 jestliže $z = 2*(e^{i10^\circ})$ do goniometrického tvaru. Jaký bude jeho argument v goniometrickém tvaru?

- *10*
- *45*
- *60*
- *90*
- *120*

Otázka ke kapitolám: 4(40), 5(80), 6(50).

40. Který tvar komplexního čísla je vhodný pro sčítání komplexních čísel?

- *algebraický*
- *goniometrický*
- *exponenciální*
- *jiný*

Otázka ke kapitolám: 1(40), 4(40), 6(40).