Automatické testování domácích úloh z programování

Tomáš Tichý *
tichy@math.cas.cz

Petr Vilím*
vilim@kti.mff.cuni.cz

Abstrakt This paper describes an automatical tester of homeworks from programming which we developed at MFF UK. The automatical tester cannot fully suplly the teacher, but can make his/her job easier by testing the homeworks on the test datas. Students send their homeworks to the tester by an e-mail and the tester responds the results in the same way. The results are also available on automatically generated web pages. Another web interface for teachers allows them to create new homeworks, browse received solutions and change assigned points. In the time we write this paper the tester have been used for two years, during the last year for two different lectures.

Tento článek popisuje automatický testovač domácích úloh který jsme vyvinuli na MFF UK. Automatický testovač samozřejmě nemůže plně nahradit učitele, ale může mu značně zjednodušit práci otestováním úlohy na testovacích datech. Studenti zasílají svá řešení testovači e-mailem, stejným způsobem jim pak testovač oznámí výsledek testu. Kromě toho si studenti mohou své výsledky přečíst na automaticky generovaných webových stránkách. Učitelé mají možnost pomocí dalšího webového rozhraní zadávat úlohy, prohlížet řešení studentů a měnit jejich bodové ohodnocení. V době vzniku tohoto článku se testovač používal dva roky, poslední rok pro výuku dvou různých předmětů.

1 Úvod

Problémem výuky programování v prvních ročnících na MFF UK je poměrně velké zaměření na teorii a zanedbávání praxe. Na cvičeních se většinou programuje „na sucho“ na tabuli, cvičící nahrazuje kompilátor i interpreter. Tento model výuky je vhodný pro programy, u kterých je hlavní myšlenka, avšak zvláště studenti neinformatického zaměření tak téměř nepřijdou do styku s počítačem. U praktického testu pak mají často velké problémy i ti studenti, kteří na cvičení excelovali, protože se vůbec nevyznají ve vývojovém prostředí.

Jako cvičící programování v jazyku Pascal jsme si tento problém uvědomovali. Řešením je dávat studentům domácí úlohy, které budou muset řešit prakticky a řešení budou odevzdávat na disketě, nebo lépe posílat e-mailem. Při tomto přístupu ale narazíme na následující problémy:

- Studentů je mnoho, cvičící budou zahlceni e-maily.
- O nalezených chybách se student dozví až za několik dní, kdy už si většinou moc nepamatuje, jak jeho řešení vypadalo.
- Opravování spousty jednoduchých domácích úkolů je zdlouhavá a nudná práce, proto bychom určitě ze svého cíle dříve nebo později slevili.

K naprogramování testovače domácích úloh nás inspiroval testovač úloh na programátorské soutěži ACM. Náš testovač funguje takto:

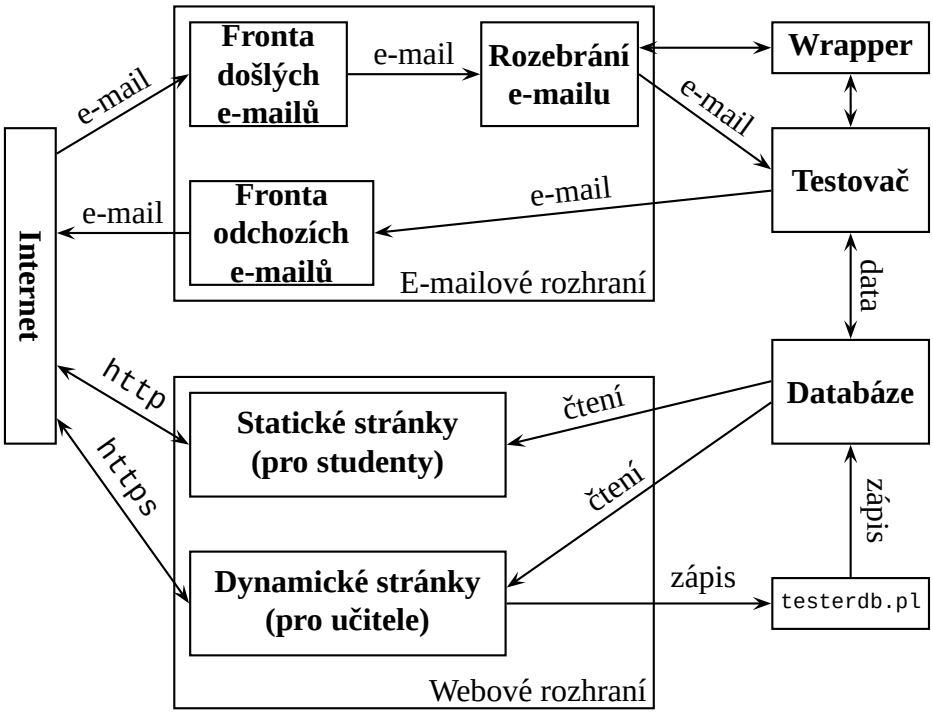
* Matematicko-fyzikální fakulta, Univerzita Karlova v Praze

– Student pošle e-mail obsahující zdrojový kód řešení jeho domácího úkolu.
– Z emailu se vybere zdrojový kód, přeloží a spustí na testovacích datech.
– Pokud se program chová správně na veřejných i tajných testovacích datech, jsou studentovi započteny body. Veřejná testovací data jsou součástí zadání.
– Testovač oznámí výsledek testování v odpovědi na příchozí e-mail.
– Dotazy na nejasnosti v řešení lze zaslat cvičícímu prostým odpovězením na tento e-mail.
– Přidělené body se objeví také na stránkách testovače, studenti tak spolu mohou soutěžit.

Je zřejmé, že testovač nemůže nahradit při kontrole učitele, protože testuje pouze funkci programu a ne elegantnost řešení. Nicméně cvičící má možnost si řešení prohlédnout dodatečně. Velkou výhodou je interaktivita – o výsledku testování se student dozví během několika málo minut (velkou roli zde hraje čas potřebný na doručení e-mailu).

2 Návrh systému

Testovač je provozován na počítači s OS Linux, jednotlivé části jsou napsány přímo jako shellové skripty nebo v jazyku Perl, pro webové rozhraní se používá HTTP server Apache. Návrh systému je schématicky nakreslen na obrázku 1, jednotlivé moduly jsou popsány v následujících odstavcích:



Obrázek 1: Schéma systému

E-mailové rozhraní

Studenti mohou svá řešení posílat na adresu svého cvičícího, nebo přímo testovači. Do předmětu (*subject*) vyplní svůj univerzitní *login*, název úlohy a mohou si zde také vybrat

kompilátor (případně interpreter), pomocí kterého bude úloha testována. Např. pro jazyk Prolog jsou k dispozici tři různé interpretery. Samotné řešení je pak možné napsat přímo do těla mailu, nebo ho poslat jako přílohu.

Na straně testovače je e-mail přijat skriptem, který jej zařadí do fronty pro sekvenční zpracování, aby nedocházelo k dočasným přetížením. Po vyjmutí z fronty je e-mail rozdělen na jednotlivé části (hlavičku, tělo a přílohy) a z nich jsou vybrána potřebná data pro testování, tj. *login* studenta, identifikátor úlohy, zdrojový kód úlohy a případně zvolený překladač. Samotný test pak zapíše výsledky do databáze, ze které je pak sestavena odpověď na příchozí e-mail, která je zařazena do fronty na odeslání.

Při testování úlohy je třeba dbát na bezpečnost – na počítači je spuštěn zcela neznámý program, což znamená velké riziko. Proto jsou úlohy testovány v bezpečně odděleném prostoru, což zaručuje modul nazvaný bezpečnostní *wrapper*. V tomto prostředí jsou také dekódovány e-maily (dekódovací software ukládá přílohy do souborů podle jmen z e-mailu).

Databáze

Veškerá zadání, testovací data, řešení atd. musí být uložena v perzistentní paměti počítače – databázi. Protože velká část testovače je napsána shellu a kvůli jednoduššímu spravování jsme nezvolili SQL ale pouze souborový systém. Nevýhodou tohoto řešení je méně příjemná práce s databází ve webových rozhraních.

Webové studentské rozhraní

Webového rozhraní pro studenty se počítá s poměrně velkým množstvím přístupů. Proto nejsou stránky tvořeny dynamicky z databáze, ale jsou vytvořeny předem a přepisují se po změnách v databázi (samozřejmě pouze změněná část z nich). Pro tvorbu stránek jsme použili jazyk Perl, pro větší konfigurovatelnost má každá stránka svou šablonu (*template*).

Studenti mají k dispozici tyto stránky:

- Seznam zadání úloh. (Používá se jako *server side include* do jiných stránek.)
- Pro každou úlohu stránku zadání, která obsahuje také veřejnou část testovacích dat a ukázkový výstup na těchto datech.
- Tabulku studentů se seznamem jejich vyřešených příkladů a bodovým ziskem. Studenti jsou také rozděleni do jednotlivých výukových kruhů.
- Stránka studenta, kde nalezne veškeré informace, které mu testovač poslal e-mailem (kdy odeslal kterou úlohu, jak dopadlo testování atd.).

Webové učitelské rozhraní

Toto rozhraní vzniklo teprve nedávno. Dovoluje pracovat s testovačem i učitelům, kteří nemají přímý přístup k databázi testovače.

Na rozdíl od studentského rozhraní jsou stránky tvořeny dynamicky. K tvorbě stránek byl opět použit jazyk Perl, pro zrychlení také modul `modperl` pro HTTP server Apache. Protože učitelé mohou pomocí tohoto rozhraní měnit databázi testovače, kladli jsme velký důraz na bezpečnost. Stránky pro učitele jsou proto přístupné pouze pomocí bezpečného šifrovaného protokolu `https` a každý učitel se musí prokázat svým heslem.

Pomocí tohoto rozhraní učitelé mohou:

- Přidávat a měnit zadání úkolů včetně testovacích dat, maximálního bodového ohodnocení apod.

– Prohlížet si řešení jednotlivých studentů přímo na stránkách (se zvýrazněním syntaxe) nebo je stahovat.
– Měnit přidělené body za příklady.
– Přidělit studentovi body navíc, nebo je naopak ubrat.

Samotný webový server nemá práva měnit databázi testovače, když to chce udělat, musí použít speciální program `testerdb.pl` a prokázat se mu jménem a heslem učitele.

Testovač úloh

Vstupem testovače je několik souborů (přílohy e-mailu), z nichž některý obsahuje zdrojový kód zasílaného řešení. Z principiálních důvodů je problém rozpoznat zdrojový kód od dalších příloh, proto testovač vyzkouší zkompilovat všechny.

Testovač podporuje několik programovacích jazyků, pro každý jazyk může být dáno několik kompilátorů případně interpeterů. V současnosti jsou podporovány jazyky Pascal (GNU Pascal, Free Pascal), Prolog (GNU Prolog, SWI Prolog, SICStus) a Haskell (GHC, HUGS). Testovač lze snadno rozšířit o další jazyky a kompilátory/interpretery napsáním tří jednoduchých skriptů: pro kompilaci, spuštění programu na daném vstupu a porovnání získaných výsledků se vzorovým výstupem. Tento návrh umožňuje použití kompilátorů i interpreterů, či zadávání úloh s nejednoznačným řešením. Každý programovací jazyk má svůj implicitní kompilátor/interpreter.

Zdrojový kód je zkompilován vybraným kompilátorem (implicitním nebo zvoleným uživatelem, musí se ale jednat o kompilátor jazyka daného v zadání úlohy).

Testování programu se provádí s pomocí zmíněných skriptů příslušejících danému kompilátoru. Napřed je test proveden na vstupech zveřejněných jako součást zadání. Do protokolu je zaznamenán výsledek a případné rozdíly mezi získaným výstupem vzorovými výsledky. Bylo-li toto testování úspěšné na všech veřejných vstupech, následuje testování na tajných vstupech, o výsledku tohoto testování se uživatel dozví pouze zda byl test úspěšný nebo ne.

Je zde samozřejmě kladen velký důraz na bezpečnost, proto jsou veškeré kritické operace (kompilace, spouštění programu i porovnávání výsledků) prováděny s pomocí bezpečnostního wrapperu a časovými limity.

Bezpečnostní wrapper

Bezpečnostní wrapper je část testovače značně nezávislá na ostatních. Úlohou tohoto modulu je spustit daný program (testovanou úlohu či program na zpracování e-mailů), ale znemožnit mu přístup k většině dat na počítači (program nesmí mít možnost číst nebo dokonce měnit soubory „do kterých mu nic není“). Wrapper odděluje a znemožňuje jakoukoliv nežádoucí interakci spuštěného programu s ostatními částmi testovače a jinými programy a daty v rámci operačního systému.

Wrapper má adresář se svou vlastní instalací operačního systému. Před spuštěním programu se změní kořenový adresář (programem `chroot`) právě na tuto vlastní instalaci. Testování je navíc prováděno pod jiným uživatelem. Program má oprávnění zapisovat pouze do adresáře vyhrazeného pro svou práci.

Současnou implementaci nepovažujeme za zcela ideální a pracujeme na lepším řešení. Zvažovali jsme použití již existujících programů, například `subterfugue`, avšak žádný z nich není považován za dostatečně bezpečný.

3 Současný stav a další plány

První verze testovače vznikla spontánně z výukových důvodů a byla dále zdokonalována podle aktuálních potřeb a nápadů. V souvislosti s použitím testovače pro další vyučované předměty se ukázala potřeba podpory více programovacích jazyků a potřebnost učitelského webového rozhraní. Proto jsme napsali novou (současnou) verzi testovače, ve které jsme použili zkušenosti s předchozího provozování testovače.

Naše plány dalšího vývoje můžeme shrnout takto; některé jsou méně a jiné více významné:

- dotáhnout vývoj bezpečnostního wrapperu
- vytvořit softwarový balíček pro snadnou instalaci a použití
- umožnit vytváření sad úloh pro různé skupiny studentů
- rozšířit nabídku úloh
- umožnit různé způsoby bodového ohodnocování
- odevzdávání úloh přes webový formulář

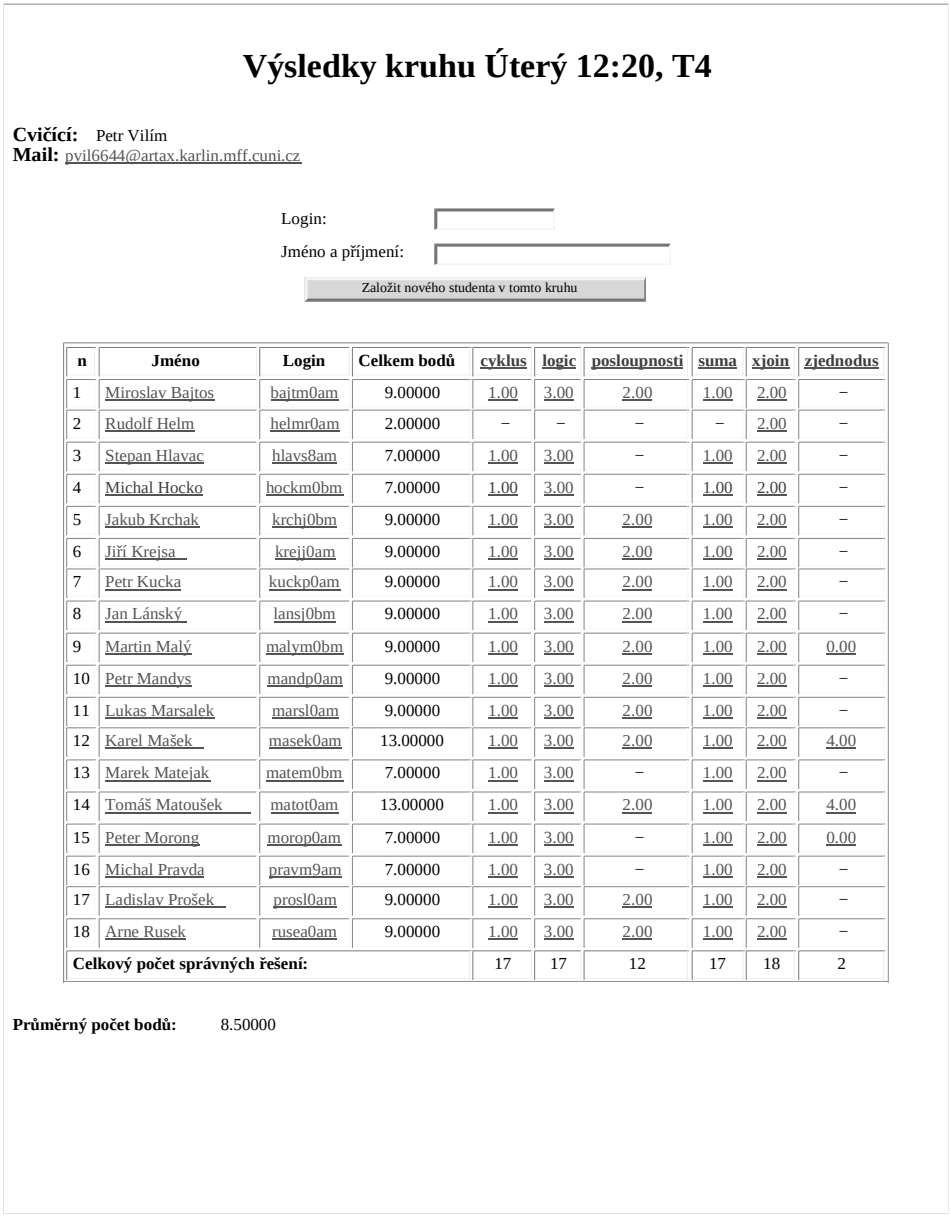
Současná verze testovače není připravena k distribuci, proto ani není nikde volně ke stažení, nicméně v případě zájmu o použití testovače pro potřeby Vaší výuky nás můžete kontaktovat.

4 Zkušenosti z praxe

- **Opisování**
  Od počátku jsme se obávali přílišného opisování. Účinnou prevencí je upozornění na archivaci řešení a na případné dodatečné anulování výsledků. Opisování lze účinně testovat unixovým příkazem `diff`. Pokud nejsou zadané úlohy příliš těžké, opisování se moc neobjevuje a spíše se můžeme setkat se soutěživým přístupem studentů.
- **Praktické zkušenosti studentů**
  Zejména studenti prvního ročníku se tak naučí posílat e-maily včetně příloh, pracovat s Internetem, získají více zkušeností s vývojovým prostředím. Tak získají lepší předpoklady pro složení zkoušky a budoucí využití získaných znalostí.
- **Klesající bodové ohodnocení**
  Počet přidělených bodů začne po termínu odevzdání klesat. To motivuje studenty odevzdávat úlohy v termínu, ale současně jim dává částečnou volnost vlastního rozhodování. Ztracené body mohou dohnat na nepovinných příkladech.
- **Nepřesné čtení zadání**
  Občasným problémem začínajících uživatelů je nerespektování přesného zadání a jeho svévolné úpravy s cílem řešení zlepšit, například vypisováním nějakých dodatečných nebo ladících údajů (např. „zadej číslo:“).
- **Pozitivní reakce studentů**
  Interakce s testovačem je na rozdíl od interakce s učitelem okamžitá, což šetří studentům čas. Hned vědí, zda jejich řešení je správné, či v něm mají nějakou chybu. Případné triviální chyby snadno odhalí z testovačem zaslané odpovědi.

5 Příklady

Uvádíme několik příkladů, jak může vypadat webové/e-mailové rozhraní z pohledu studenta i z pohledu učitele - skutečné údaje z letošní výuky. Tyto stránky naleznete (v současnosti) na adrese `http://kiwi.ms.mff.cuni.cz/~testovac`.



Výsledky kruhu Úterý 12:20, T4

Cvičící: Petr Vilím
Mail: pvil6644@artax.karlin.mff.cuni.cz

Login:
Jméno a příjmení:
Založit nového studenta v tomto kruhu

n	Jméno	Login	Celkem bodů	cyklus	logic	posloupnosti	suma	xjoin	zjednodus
1	Miroslav Bajtos	bajtm0am	9.00000	1.00	3.00	2.00	1.00	2.00	–
2	Rudolf Helm	helmr0am	2.00000	–	–	–	–	2.00	–
3	Stepan Hlavac	hlavs8am	7.00000	1.00	3.00	–	1.00	2.00	–
4	Michal Hocko	hockm0bm	7.00000	1.00	3.00	–	1.00	2.00	–
5	Jakub Krchak	krchj0bm	9.00000	1.00	3.00	2.00	1.00	2.00	–
6	Jiří Krejsa	krejj0am	9.00000	1.00	3.00	2.00	1.00	2.00	–
7	Petr Kucka	kuckp0am	9.00000	1.00	3.00	2.00	1.00	2.00	–
8	Jan Lánský	lansj0bm	9.00000	1.00	3.00	2.00	1.00	2.00	–
9	Martin Malý	malym0bm	9.00000	1.00	3.00	2.00	1.00	2.00	0.00
10	Petr Mandys	mandp0am	9.00000	1.00	3.00	2.00	1.00	2.00	–
11	Lukas Marsalek	marsl0am	9.00000	1.00	3.00	2.00	1.00	2.00	–
12	Karel Mašek	masek0am	13.00000	1.00	3.00	2.00	1.00	2.00	4.00
13	Marek Matejak	matem0bm	7.00000	1.00	3.00	–	1.00	2.00	–
14	Tomáš Matoušek	matot0am	13.00000	1.00	3.00	2.00	1.00	2.00	4.00
15	Peter Morong	morop0am	7.00000	1.00	3.00	–	1.00	2.00	0.00
16	Michal Pravda	pravm9am	7.00000	1.00	3.00	–	1.00	2.00	–
17	Ladislav Prošek	prosl0am	9.00000	1.00	3.00	2.00	1.00	2.00	–
18	Arne Rusek	rusea0am	9.00000	1.00	3.00	2.00	1.00	2.00	–
Celkový počet správných řešení:				17	17	12	17	18	2

Průměrný počet bodů: 8.50000

Obrázek 2: Webová stránka: Výsledky studentů v kruhu

6 Závěr

Podle našich zkušeností se testovač jednoznačně osvědčil. Splnil úlohu, pro kterou byl napsán, tj. studenti získali praktické zkušenosti s programováním přímo u počítače. To se projevilo nejen na jejich studijních výsledcích, ale také na cvičeních – programování už pro ně nebylo pouhá teorie na tabuli či v sešitě, ale začali sami pokládat praktické otázky. Programování je začalo daleko více zajímat.

Testovač se osvědčil zejména při učení se syntaxe programovacího jazyka a jednoduchých programovacích technik. Nemyslíme ale, že by byl vhodný pro výuku ve vyšších ročnících na MFF UK, kdy se klade větší důraz na logiku věci.

Zdenek Kavalir

Cvičící: Tomáš Tichý
Cvičení: Středa 9:00, E4
Celkový počet bodů: 7.00000

Domácí úkoly:

Úkol	Bodů	Max. bodů	Stav	Pokusů o odevzdání	Zkompilovatelných pokusů	Pokusů správně nad vzorovým vstupem
cyklus	1.00000	1.00000	Uznán	1	1	1
logic	3.00000	3.00000	Uznán	1	1	1
posloupnosti	–	2	Neodevzdán	–	–	–
suma	1.00000	1.00000	Uznán	3	3	1
xjoin	2.00000	2.00000	Uznán	2	2	1
zjednodus	0.00000	4.00000	NEuznán	3	3	3

Obrázek 3: Webová stránka: Informace o studentovi

1. pokus

```
*** Automatic test ***
login:    hockm0bm
task:     xjoin
language: prolog
compiler: swi
cksum:    6196982 180
time:     2001-12-12_17-09-27_uq21949

Compiling...

Trying sample test vzor_0.in ...
->Test je OK.
Trying sample test vzor_1.in ...
->Test je OK.
Trying sample test vzor_2.in ...
->Test je OK.
Trying sample test vzor_3.in ...
->Test je OK.
Trying sample test vzor_4.in ...
->Test je OK.
Running test test_0.in ...
->Test is OK.
Running test test_1.in ...
->Test is OK.
Running test test_2.in ...
->Test is OK.
Running test test_3.in ...
->Test is OK.
Running test test_4.in ...
->Test is OK.
Running test test_5.in ...
->Test is OK.
Running test test_6.in ...
->Test is OK.
Running test test_7.in ...
->Test is OK.
Running test test_8.in ...
->Test is OK.
Running test test_9.in ...
->Test is OK.

Succeessive / all tests: 15 / 15


Conclusion: task xjoin IS confirmed.
Assigned points: 2
```

Obrázek 4: Webová stránka: Automatická odpověď testovače