Rozšířený Abstrakt

Implementace plug-in modulu pro výukový systém Moodle

Bc. Jan Hybš, doc. Ing. Jiřina Královcová, Ph.D.

Abstrakt

Práce se zabývá problematikou testování dovedností studentů v předmětech zabývající se programováním. Dovednosti studentů jsou testovány prostřednictvím pedagogem definovaných algoritmických úloh, na které studenti zasílají svá řešení.

Cílem této práce je vytvořit modul pro systém Moodle, který bude umožňovat specifikaci těchto úloh a odevzdání řešení na tyto úlohy. Tento modul by měl do jisté míry zautomatizovat proces testování dovedností studentů.

V rámci práce byl vytvořen systém CoDiAna, který se skládá ze dvou částí. První část je modul pro systém Moodle sloužící pro správu algoritmických úloh a shlížení výsledků studentů. Druhou částí je exekutivní aplikace sloužící pro zpracování studenty odevzdaných řešení.

Úvod

Tato práce zasahuje do odvětví zpracování a analýzy zdrojových kódů. Cílem práce je seznámit se realizací modulů pro systém Moodle a vytvořit modul řešící problematiku správy programovacích úloh, které slouží pro otestování dovedností studentů. Systém by měl umožňovat specifikaci zadání úloh, zpracování přijatých řešení a měření hodnotících veličin. Měl by také podporovat ověření správnosti řešení porovnáním vygenerovaného výstupní souboru s referenčním výstupním souborem úlohy. Dále pak možnost testování efektivity řešení a analýzy podobnosti řešení. Systém by měl řešení studentů ohodnotit a navrhnout příslušnou známku. Další činností systému je detekce duplicitních řešení, tedy nalezení plagiátů mezi odeslanými řešeními.

Výsledný plug-in modul může být nasazen do e-learningového systému provozovaném na Fakultě mechatroniky, informatiky a mezioborových studií. Pomocí tohoto systému bude možné zjednodušit testovací proces v předmětech, které se zabývají programováním a algoritmizací.

Vhodnou strukturou systému musí být zajištěna nezávislost podporovaných programovacích jazyků při realizování algoritmických úloh. Dále je nutné důkladně prozkoumat bezpečnost systému. V systému bude nutné vytvořit opatření zabezpečující proces zpracování a testování odevzdaných řešení.

Experiment a metody

Algoritmické úlohy jsou definované zadáním a specifikací vstupního a výstupního souboru. Studenti na základně specifikace úlohy odevzdávají svá řešení v podobě algoritmu, prostřednictvím systému Moodle. Po zpracování řešení jsou naměřeny tři hodnoty, ze kterých je určen finální výsledek. Jsou to hodnoty o časové a paměťové náročnosti a informace o tom, zda je výstupní soubor studenta shodný (dle stanovených kritérií) s referenčním výstupním souborem úlohy. K určení finálního výsledku je použit následující vztah:

$$f(x) = o(x)\lceil t(x)\rceil\lceil m(x)\rceil\left(\frac{1}{2}t(x) + \frac{1}{2}m(x)\right), \; kde$$

$$o(x) = \begin{cases} 0, & chybný\ výstup \\ 1, & správný\ výstup \end{cases}, \qquad H(t) \in \langle 0{,}1\rangle, \qquad H(m) \in \langle 0{,}1\rangle \tag{1}$$

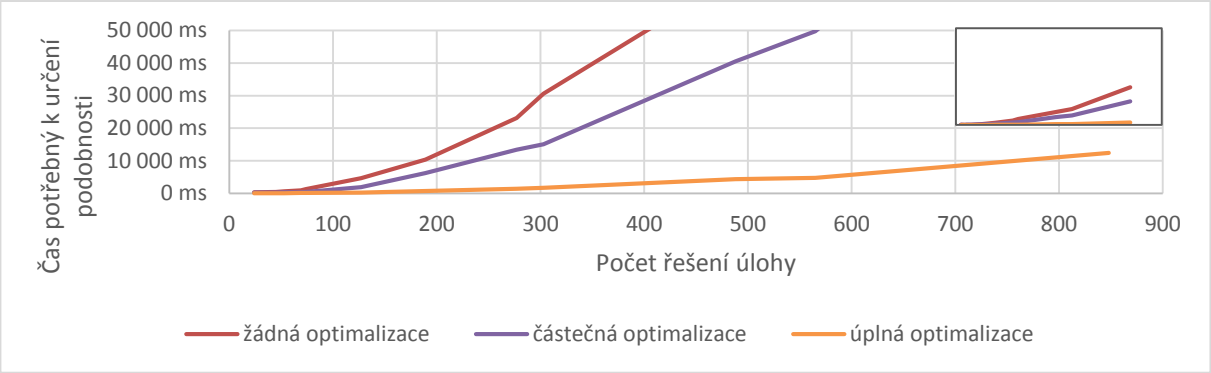
Rozšířený Abstrakt

Funkce $o(x)$ nabývá hodnot 0, pokud je výstup chybný nebo 1, pokud je výstup klasifikován jako správný. Funkce $t(x)$ určuje výsledek časové náročnosti a funkce $m(x)$ určuje výsledek náročnosti paměťové. Pokud nejsou specifikovány hodnoticí kritéria úlohy, tj. nejsou určeny hodnotící prahy, potom $t(x) = m(x) = 1$. Vztah ve výsledku popisuje průměr časové a paměťové náročnosti, pouze pokud je výstup správný za předpokladu, že žádný, ani časový, ani paměťový výsledek není nulový.

Výsledky a diskuze

V rámci této práce byl vytvořen, který je umožňuje definici programových úloh. Dále je schopen zpracovávat odevzdaná řešení v krátkém čase. Pomocí vytvořeného modulu je možné upozornit exekutivní aplikaci zasláním krátké socketové zprávy, prodleva před zpracováním je tedy minimální.

Algoritmus pro určení podobnosti odevzdaných řešení využívá optimalizačních metod, které proces syntaktické analýzy zdrojových kódů značně zrychlují. Je použit algoritmus úplné optimalizace, který využívá přepočítané výsledky analýzy kódu. Graf zobrazuje výsledky zkoumaných algoritmů. Z grafu lze vidět, že použití úplné optimalizace vede při velkém počtu řešení k velké časové úspoře.



Obrázek 1. Porovnání optimalizačních metod.

Závěr

Byl vytvořen systém CoDiAna, který umožňuje zautomatizování procesu testování dovedností studentů v předmětech s tématikou programování a algoritmizace. Pomocí tohoto systému lze vytvářet a spravovat algoritmické úlohy vytvořené pedagogem. Po zadání algoritmické úlohy mohou studenti odevzdávat svá řešení, která jsou systémem vyhodnocena – je ověřena jejich korektnost a následně navržena výsledná známka odevzdaného řešení. Výhodou systému je nezávislost na konkrétním programovacím jazyku. Systém umožňuje odevzdávat řešení v libovolném jazyce, který je serverem podporován

Výsledný systém se skládá z dvou částí. Obě části systému mohou být na různých serverech, komunikace je poté zajištěna SSH spojením. První část je modul CoDiAna systému Moodle, který slouží pro správu algoritmických úloh, odevzdávání řešení a shlížení výsledků úloh. Další částí systému je tzv. Exekutivní aplikace, která slouží pro zpracování požadavků od modulu CoDiAna. Zajišťuje kompilaci, spuštění řešení a detekci duplicitních řešení. Vhodnou strukturou aplikace je umožněno dynamické načítání rozšiřujících modulů, které přidávají podporu určitého programovacího jazyka.

Reference

[1] BARRETT, Daniel J., Richard E. SILVERMAN a Martin BLAŽÍK. SSH: Kompletní průvodce. Brno: Computer Press, 2003. ISBN 80-7226-852-X.

[2] JSch: Java Secure Channel. JCraft: Code the Craft, Craft the Code [online]. [2012] [cit. 2014-03- 12]. Dostupné z: www.jcraft.com/jsch/.

[3] Developer Documentation [online]. 2013, 2013-12-10 [cit. 2014-03-09]. Dostupné z: docs.moodle.org.