Posudek bakalářské práce

Matematicko-fyzikální fakulta Univerzity Karlovy v Praze

Autor práce	Tomáš Pavlín
Název práce	Komponování hudby pomocí programovacího jazyka
Rok odevzdání	2017
Studijní program	Informatika
Studijní obor	IPSS
Autor posudku	Jan Hajič jr. Oponent
Pracoviště	Ústav formální a aplikované lingvistiky

K celé práci	lepší	OK	horší	nevyhovuje
Obtížnost zadání	X			
Splnění zadání		X		
Rozsah práce *... textová i implementační část, zohlednění náročnosti*		X		

Řešitel vytvořil programovací jazyk *Composing* pro nedeterministické skládání hudby pomocí bezkontextových gramatik a umožnil generování skladeb v žánrech definovaných v jazyce *Composing* netechnickému publiku pomocí grafického rozhraní *SongMaker*. Konkrétní hudební prostředky a syntaktické struktury, do kterých mají být uspořádány, se definují v šablonách, které poté umožňují generovat „nekonečnou“ hudbu daného žánru. Nedeterminismus v kombinaci s elegancí bezkontextových gramatik je výtečně zvolená funkcionalita, a na základě své zkušenosti s hudební kompozicí nemám pochyb, že se jedná o potenciálně přínosný nástroj.

Autor prokázal programátorskou zdatnost. Vytvořil funkční komplexní aplikaci kombinující několik úloh: navrhl programovací jazyk, implementoval pro něj interpret, vytvořil příslušné GUI *SongMaker* a systém parametrizace hotových šablon přes toto rozhraní. Dodal též příkladové šablony. Mám však jisté pochybnosti, zda autor i přes veškerou snahu dostatečně vychází vstříc svému zamýšlenému netechnickému publiku. Práce zjevně nebyla konzutována se skladatelem.

Za nejslabší bod práce považuji dokumentaci. Uživatelská dokumentace GUI SongMaker je precizní a úplná, dokumentace jazyka *Composing* však neodpovídá na poměrně základní otázky potenciálních uživatelů. Vývojová dokumentace sice popisuje architekturu aplikace, neobsahuje však popis datového modelu a není popsáno, jak základní metody pro manipulaci s hudbou nad tímto datovým modelem operují, což je jádro činnosti programu. Také bych rád viděl vývojovou dokumentaci přiloženou k softwarovému dílu, nejen v textu práce. Referenční dokumentace jednotlivých tříd a metod pak chybí úplně.

Za nejsilnější bod práce považuji návrh aplikace: dobrou volbu funkcionality, kvalitní technickou analýzu, dobře navrženou architekturu aplikace a vhodně vybrané a integrované technologie. Celý komplexní systém plní zadání a uspokojivě funguje jako dobrý proof-of-concept.

K obhajobě mám následující dotazy:

1. Jak bych v jazyce *Composing* vytvořil šablonu pro generování kánonu?
2. Jak je reprezentována skladba během intepretace zdrojových souborů v jazyce *Composing*?
3. Grafické rozhraní SongMaker je nástroj na generování skladeb z hotových šablon spíše než grafické rozhraní k programování v jazyce *Composing*. Jak by si řešitel představoval grafické rozhraní pro tvorbu samotných žánrových šablon?

Textová část práce

	lepší	OK	horší	nevyhovuje
Formální úprava *... jazyková úroveň, typografická úroveň, citace*		X		
Struktura textu *... kontext, cíle, analýza, návrh, vyhodnocení, úroveň detailu*		X		
Analýza	X			
Vývojová dokumentace			X	
Uživatelská dokumentace			X	

Práce je psaná jasně, přehledně, má logickou strukturu a splňuje náležitosti odborného stylu.

V úvodu autor cituje z excelentního článku (Edwards, 2011) pouze příklad s Mozartem, ačkoliv článek dodává mnoho dalších čerstvějších příkladů algoritmických postupů v kompozici. Nedeterminismus například souvisí s prací Johna Cage či Iannise Xenakise (Stochastic Music Programme, v Edwardsově článku poměrně detailně popsané).

Kapitola 2 je obdobně krátká. (Vzhledem k tomu, že práce je především implementační a ne výzkumná, není toto zásadní výtka.) Autor správně poznamenává, že v současnosti je v aplikacích algoritmické kompozici nejsilnější přístup Live Coding, úplně však chybí reflexe komponování pomocí strojového učení, které především v posledních letech výrazně postupuje: viz Magenta (Google) či FlowMachines (Sony). Mimo jiné by se totiž jazyk *Composing* dal využít pro generování stylově konzistentních syntetických trénovacích dat pro tyto metody.

Kapitola 3 je poněkud nepřesná v zavádění hudební terminologie a je otázka, zda z ní čtenář, který by danou terminologii neznal, získá dostatečné znalosti. V pojmu ,,rytmus" např. chybí informace, že délka not se měří v *dobách*, a chybí popis vztahu mezi rytmem a ,,hodinkovým" časem u pojmu tempo. Pro jaké čtenáře je tato kapitola vlastně určena?

Popis ovládání GUI SongMaker v kap. 4 je vyčerpávající a přehledný. Popis jazyka *Composing* v kapitole 5 však i přes velmi vhodně zvolenou funkcionalitu ukazuje značné rezervy v porozumění potřebám uživatelů. Dobře krok za krokem vysvětluje používání jazyka, avšak končí u triviální klavírní struktury melodie/doprovod, a vůbec nemluví o vícehlasu či kompozici pro komorní soubory a orchestry, která je například pro zamýšlené uživatele z řad tvůrců hudby do filmů či počítačových her klíčová. Také kupříkladu není popsáno, jak se tvoří trioly (běžný rytmický prostředek, kdy se délka noty dělí nikoliv na dvě, nýbrž na tři stejně dlouhé úseky). Bylo by třeba podobných návodů několik, pro různé hudební textury. Nabízí se pak také oddělení uživatelské dokumentace mimo bakalářskou práci, a v práci ponechat pouze přesný popis jazyka s jedním či dvěma příklady.

Jednoznačně nejzajímavější částí jazyka *Composing* je nedeterminismus. Ačkoliv technicky přesný popis nedeterministické syntaxe v sekci 5.2 je, bylo by vhodné přidat příklad, ve kterém je několik netriviálních bloků (tj. bloků, kde se složené závorky nedají vynechat).

Nedostatky vývojové dokumentace (kap. 6 a 7) jsou také podstatné. Práce přehledně popisuje celou širokou paletu použitých technologií a architekturu aplikace a kvalitně zdůvodňuje jednotlivá návrhová rozhodnutí, avšak stejně jako v uživatelské chybí ve vývojové dokumentaci dostatečný popis základních metod `melody`, `rhythm`, či `play`, respektive vůbec popis datového modelu hudby v jazyce Composing: jak je reprezentována ,,rozeskládaná" hudba? Ve vývojové dokumentaci také chybí v popisu interpreteru způsob, kterým se implementuje nedeterminismus – stačí sotva jedna věta, avšak očekával bych u tohoto klíčového prvku explicitní popis.

Po formální stránce jsou v práci drobné nedostatky. Chybí číslování příkladů zdrojového kódu, zkratka .NET je občas psaná jako .Net, nelsabičné předložky se občas najdou na konci řádku, nedeterminizmus vs. nedeterminismus, atd. Na druhou stranu je však práce prosta pravopisných chyb. Otázkou je také volba jazyka práce – pokud je aplikace lokalizovaná do angličtiny, proč je její vývojová dokumentace pouze česky?

Implementační část práce

	lepší	OK	horší	nevyhovuje
Kvalita návrhu *... architektura, struktury a algoritmy, použité technologie*	X			
Kvalita zpracování *... jmenné konvence, formátování, komentáře, testování*		X		
Stabilita implementace		X		

Zdrojový kód obsahuje jen minimum komentářů. Chybí také vygenerovaná referenční dokumentace a ačkoliv jsou v práci integrační testy, chybí unit-testing. Na druhou stranu je struktura kódu dobrá, návrh je logický a přehledný a jmenné konvence jsou dodržovány.

Instalace proběhla, avšak odkaz na ploše slíbený v uživatelské dokumentaci vedl napřed na instalační soubor `*.msi` a až po druhém proběhnutí instalace program SongMaker našel a nabízel všechny poskytované šablony.

Software je stabilní. Jediná chyba během užívání byla nutnost po přehrání vygenerované skladby zavřít okno přehrávače, neboť jinak program odmítal příští skladbu zahrát s tím, že "MIDI soubor je ještě využíván" – zde je řešením lepší zacházení s názvy dočasných souborů.

Z uživatelského hlediska je SongMaker relativně příjemný a přehledný, o jazyku *Composing* to už platí méně. V poskytnutých žánrových šablonách navíc chybí komentáře. Pro informatiky či skladatele, kteří již mají nějakou zkušenost s podobnou technologií, nepředstavuje problém jazyk se naučit, avšak pro netechnické uživatele, na které práce také cílí, bude tvorba šablon poměrně obtížná. Práce v tomto ohledu nutně potřebuje, aby autor přešel náměstí z budovy MFF do budovy HAMU a konzultoval vývoj se studenty kompozice – zájem o takovou aplikaci jakožto tvůrčí nástroj by jistě byl.

Autor práce připravil potenciálně velmi užitečný tvůrčí nástroj, řízený nedeterminismus je výborný nápad. Na základě vlastní zkušenosti ze studia kompozice na JAMU a znalosti skladatelského prostředí bych doporučoval nástroj dále vyvíjet. Dovolím si tedy uvést několik konkrétních návrhů, které by mohly pomoci vytvořený software skutečně posunout do skladatelského světa.

1. Uživatelská dokumentace jazyka je sice precizní, avšak neúplná (viz kritika výše). Bylo by také třeba vytvořit sadu návodů, jak krok za krokem vytvářet postupně složitější skladby různého charakteru. Chybí také obrazový (tj. notový) příklad toho, jak se promítá syntaktický strom do výsledného notopisu (vhodné dodělat k fig. 3.1) – vysvětlení je dobré, avšak pro netechnické uživatele, kteří nejsou zvyklí číst formální zápisy, bude bez příslušného obrázku tento popis jen obtížně pochopitelný.
2. Pro pohodlné soužití s programem ještě chybí zabudovaná možnost vidět vygenerovanou hudbu zapsanou v notách, na jedno kliknutí: vedle tlačítka „Play" také „Show". To totiž představuje zpětnou vazbu, se kterou jsou skladatelé zvyklí rychle pracovat a bez které je naopak velmi obtížné po sobě výsledky práce zkontrolovat. Pro tuto funkcionalitu existují technologie jako LilyPond (poskytuje rendering do PDF z plaintextového formátu založeném na LaTeXu), kam lze MIDI importovat přes program `midi2ly`.
3. Také si nejsem jist některými syntaktickými volbami jazyka *Composing*. Především v českém prostředí je například poměrně nepohodlné neustále psát složené závorky. Opět: pro technické uživatele to problém není, pro netechnické je to neustálé drobné nepohodlí. Bylo by vhodné projít již hotový jazyk a najít další způsoby, jak jej dále zjednodušit.

Celkové hodnocení Velmi dobře

Práci navrhuji na zvláštní ocenění Ne

Datum

Podpis