Posudek diplomové práce

Matematicko-fyzikální fakulta Univerzity Karlovy

Autor práce	Ladislav Láska		
Název práce	Scalable link-time optimization		
Rok odevzdání	2017		
Studijní program	Informatika	**Studijní obor**	Diskrétní modely a algoritmy

Autor posudku	Jan Hubička	**Role**	vedoucí
Pracoviště	Informatický ústav Univerzity Karlovy		

Práce se věnuje otázce optimalizace celých programů. Při klasickém modelu překladu je program rozdělen do množství kompilačních unit, které jsou překládány zvlášť. Tento postup ale limituje možnosti inter-procedurální optimalizace a proto moderní překladače nabízí alternativu, kde se celý program optimalizuje během procesu linkování známý jako link-time optimization, LTO.

LTO sebou přináší množství implementačních problémů. Jednou z hlavních je nutnost minimalizovat dobu překladu a upravit algoritmy inter-procedurální analýzy tak, aby pracovaly efektivně na programech čítající miliony funkcí.

Řešitel se dokázal zorientovat ve složitém prostředí produkčního překladače GNU Compiler Collection a zvolil si práci na points-to analýze. Tato analýza má za úkol určit, pro každý ukazatel v programu, na které paměťové lokace může ukazovat. Protože velké programy (například Firefox) obsahují miliony lokací i ukazatelů, je nutné efektivně uložit velké množství množin, které dohromady mohou obsahovat řádově 10^{12} prvků. GCC Tuto optimalizace (ve velmi slabé podobě) implementuje, ale vypnutá pro všechny úrovně optimalizace z důvodů nepřiměřených nároků na dobu překladu i použitou paměť.

Je patrné, že přesná řešení pro points-to analýzu nejsou prakticky zajímavá a proto je nutné přistoupit k aproximaci. Klasický aproximační algoritmus vychází z práce Bjarne Steensgaarda a používá k reprezentaci datovou strukturu union-find. Toto elegantní řešení však bylo dlouhou dobu zatíženo patentem a také vede ke ztrátě přesnosti kterou nelze pomocí nějakého parametru ovlivnit.

Jako alternativu řešitel navrhl použití upravených Bloomových filtrů. Tato datová struktura je kompaktní a jak se podařilo ukázat, vhodná k implementaci základních operací nutných pro implementaci algoritmu points-to analýzy. Navíc přesnost analýzy lze nastavit. Práce dokládá praktickou použitelnost tohoto řešení při překladu přohlížeče Firefox (jednoho z největších open-source programů) a kontroluje konzervativnost výsledků oproti přesnému řešeni implementovanému pomocí bitmap. Věřím, že toto řešení se podaří prosadit do oficiálního vydání GCC v dalším vývojovém

cyklu a využít toto řešení i pro další optimalizace založené na bitmapách (například ipa-reference). Pomůže tak vyřešit letitý problém, který blokuje řešení mnoha dalších problému v oblasti.

Implementační část práce tedy považuji za velmi dobrou a inovativní. Použití Bloomových filtrů v points-to analýze je sice tématem několika publikovaných článků, ale tyto implementace nebyly nikdy prakticky testovány a nasazeny v produkčním prostředí. Implementace analýzy v GCC se od učebnicového algoritmu points-to velmi liší, protože musí řešit množství detailů nutných ke korektnímu překladu skutečných programů. Proto nebylo nasazení filtrů v GCC jednoduché.

Výhrady mám ale k textové části. Je vidět, že některé části vznikaly na poslední chvíli a jsou nepřehledné. Práce by zasloužila lepší prezentaci samotné datové struktury. Měření na prohlížeči Firefox jsou sice vypovídající a průkazná, ale bylo by vhodné je ověřit i na dalších testech a zpracovat lépe kvalitu výsledné aproximaci v závislosti na velikosti filtrů a v provnání se Steensgaardovým algoritmem. I přes tyto nedostatky doporučuji práci přijmout jako diplomovou.

Práci doporučuji k obhajobě.

Práci nenavrhuji na zvláštní ocenění.

V Leeds dne 24. 1. 2017

Podpis: