UNIVERSITEIT GENT

Studiefiche

Vanaf academiejaar 2022-2023

Compilers (E018520)

Wegens Covid19 kan mogelijk afgeweken worden van de onderwijs- en evaluatievormen. Dergelijke afwijkingen zullen via Ufora worden gecommuniceerd.

Cursusomvang *(nominale waarden; effectieve waarden kunnen verschillen per opleiding)*

Studiepunten 6.0 **Studietijd 180 u** **Contacturen** 45.0 u

Aanbodsessies en werkvormen in academiejaar 2022-2023

A (semester 2)	Engels	Gent	hoorcollege	21.25 u
			werkcollege: geleide oefeningen	7.5 u
			practicum	13.75 u

Lesgevers in academiejaar 2022-2023

De Sutter, Bjorn	TW06	Verantwoordelijk lesgever

Aangeboden in onderstaande opleidingen in 2022-2023	**stptn**	**aanbodsessie**
Educatieve Master of Science in de wetenschappen en technologie (afstudeerrichting informatica)	6	A
Master of Science in Computer Science Engineering	6	A
Master of Science in de industriële wetenschappen: informatica	6	A
Master of Science in de informatica	6	A
Master of Science in de ingenieurswetenschappen: computerwetenschappen	6	A
Uitwisselingsprogramma informatica (niveau master)	6	A

Onderwijstalen

Engels

Trefwoorden

contextvrije grammatica, afleidingsboom, parsers, abstracte-syntaxbomen, semantische analyse, basisblokken, instructieselectie, registertoewijzing, dataverloopanalyse, controleverloopanalyse, codegeneratie, optimalisatie, ondersteuning managed talen, geheugensanering, pijplijnen, inroosteren, technieken voor hoog-niveautalen, just-in-time-compilatie

Situering

Een computer werkt met machinetaal terwijl een programmeur schrijft in een hogere programmeertaal. Compilers vormen de verbinding tussen de programmeur en de computer. Een compiler zal het bronprogramma op een correcte en efficiënte manier vertalen naar machine-instructies. Dit gebeurt in twee grote stappen. in het front-end gedeelte worden lexicale en parsing technieken gebruikt om het programma om te vormen naar een intermediaire voorstelling. In het back-end gedeelte worden verschillende optimisatietechnieken en codegeneratietechnieken toegepast om het programma zo goed mogelijk af te beelden op de onderliggende architectuur. In dit vak worden de theoretische basisconcepten gekoppeld aan de implementatie van een wereldwijd gebruikte compiler en aan een aantal recente ontwikkelingen in het domein.

Inhoud

- Inleiding: Datastructuren voor compilers, Overzicht van fasen in de compilatie
- Lexicale analyse: Tokens en reguliere expressies, Eindige automaten
- Parsing: Contextvrije grammatica, Predictieve en LR-parsers

- Abstracte syntax: Semantische acties, Abstracte syntaxboom
- Semantische analyse: Symbooltabellen, Typechecking
- Activatierecords: Stapelvensters en locale variabelen, Doorgeven van parameters
- Intermediaire code: Voorstellingsboom, Vertaling
- Basisblokken en traces: Canonische bomen, Voorwaardelijke sprongen
- Instructieselectie: Algoritmen voor instructieselectie
- Liveness analyse en registerallocatie: Levensduurvergelijkingen, Graafkleuring en registertoewijzing
- Datastroomanalyse: Datastructuren voor datastroomvoorstelling, Datastroomalgoritmen
- Lusoptimisaties: Reduceerbare grafen, Lustransformaties, Controleverloopanalyse
- Codegeneratie: pijplijnen en inroosteren
- Optimalisatie van geheugengebruik: prefetching, lustransformaties
- Ondersteuning voor hoge-niveautalen: geheugensanering, polymorfisme, klassenhiërarchieën
- Just-in-time compilatietechnieken
- Capita selecta van state-of-the-art compilertechnieken

Begincompetenties

- Kennis van progammeertalen, programmeren in C
- Basiskennis van computerarchitectuur (de software-hardware-interface)

Eindcompetenties

1 De betekenis van de verschillende fasen in een compiler begrijpen.
2 Inzicht hebben in de constructie van automata voor de generatie van lexicale analyzers.
3 Het gebruik van verschillende parsertechnieken (o.m. LL, LR) beheersen.
4 De software voor de generatie van lexicale analyzers en grammaticale parsers gebruiken.
5 Typechecking inbouwen.
6 Een abstracte syntaxboom interpreteren.
7 Controle- en datastroomgrafen opbouwen.
8 Afhankelijkheidsanalyse gebruiken voor levensduur-analyse en code-optimisatie.
9 Instructieselectie-algoritmen vergelijken en toepassen.
10 Registers toewijzen.
11 Intermediaire voorstellingen hanteren voor de generatie van machine-onafhankelijke code.
12 Luseigenchappen analyseren en transformaties toepassen voor code-optimisatie.
13 Dataverloop analyseren en code transformeren voor optimalisatie.
14 Code inroosteren en lussen pijplijnen.
15 Technieken zoals prefetching and tiling toepassen om geheugentoegangen te optimaliseren.
16 Constructies in hoog-niveautalen ondersteunen, zoals geheugensanering, polymorfisme, en overerving
17 Just-in-time compilatietechnieken begrijpen.

Creditcontractvoorwaarde

Toelating tot dit opleidingsonderdeel via creditcontract is mogelijk mits gunstige beoordeling van de competenties

Examencontractvoorwaarde

Dit opleidingsonderdeel kan niet via examencontract gevolgd worden

Didactische werkvormen

Hoorcollege, practicum, werkcollege: geleide oefeningen

Toelichtingen bij de didactische werkvormen

In de hoorcolleges wordt de theoretische onderbouw en interne werking van compilers besproken.
De studenten krijgen voor de practica zeven programmeeropdrachten die ze deels tijdens begeleide sessies (online of in de laptopklassen), deels daarbuiten moeten uitvoeren. Ze moeten hun software, alsook soms een klein verslag, indienen. Ze werken in groepjes van twee. Samenwerking met andere groepen is niet toegestaan. Na het indienen van de werkstukken kunnen de studenten uitgenodigd worden tot een kort evaluatiegesprek waarop ze over hun werkstukken ondervraagd worden om te checken of ze de beoogde competenties verworven hebben. De meeste practica handelen om onderdelen van LLVM, een van de meest gebruikte compilers ter wereld.
In de geleide oefeningen wordt de werking van basisalgoritmes ingeoefend.

Leermateriaal

- Modern compiler implementation in C, Andrew W. Appel with Maia Ginsburg, Cambridge

University Press, ISBN 052158390X, LCCN QA76.73.C15.A63, prijs: 51 Euro
- Slides hoorcolleges, te koop aan productiekost VTK

Referenties

- Modern compiler implementation in C, Andrew W. Appel with Maia Ginsburg, Cambridge University Press, ISBN 052158390X, LCCN QA76.73.C15.A63
- LLVM Manual, http://llvm.org/docs/

Vakinhoudelijke studiebegeleiding

Evaluatiemomenten

periodegebonden en niet-periodegebonden evaluatie

Evaluatievormen bij periodegebonden evaluatie in de eerste examenperiode

Schriftelijk examen, openboekexamen

Evaluatievormen bij periodegebonden evaluatie in de tweede examenperiode

Schriftelijk examen, openboekexamen

Evaluatievormen bij niet-periodegebonden evaluatie

Mondeling examen, werkstuk, verslag

Tweede examenkans in geval van niet-periodegebonden evaluatie

Examen in de tweede examenperiode is enkel mogelijk in gewijzigde vorm

Toelichtingen bij de evaluatievormen

De niet-periodegebonden evaluatie bestaat uit zeven werkstukken in de vorm van software, sommige met korte verslagen erbij, die gespreid zijn over het semester. Het verslag en elk werkstuk moet voor de start van het volgende practicum of op een opgegeven datum ingediend worden. Na het indienen van het verslag volgt er mogelijk ook een mondelinge evaluatie waarop de student ondervraagd wordt over de gebruikte methodiek en de behaalde resultaten. Voor de werkstukken werken de studenten soms in groepjes van twee. Aan de hand van een peer assessment over hun bijdrage aan het werkstuk en verslag, en eventueel na een gesprek hierover bij tegenstrijdige assessments, wordt bepaald of beide studenten al dan niet dezelfde punten krijgen.
Niet op tijd ingediende verslagen zonder geldige reden van afwezigheid (zoals een doktersbriefje) worden met een nul gekwoteerd voor dat onderdeel van de niet-periodegebonden evaluatie.
De niet-periodegebonden evaluatie verloopt gelijkaardig voor de tweede examenperiode. De student wordt dan geëvalueerd op basis van nieuwe werkstukken met een gelijkaardige belasting als in de eerste periode, maar moet die wel individueel maken. De verslagen ervan zullen een week voor het examen ingediend moeten worden. Vlak na dat schriftelijk periodegebonden examen volgt dan een mondelinge evaluatie over de werkstukken.
De periodegebonden evaluatie is schriftelijk, en verloopt deels met gesloten boek (theorie en eenvoudige oefeningen) en deels met open boek (oefeningen).

Eindscoreberekening

Verdeling van de punten: 3/5 voor de periodegebonden evaluatie en 2/5 voor de niet-periodegebonden evaluatie. Speciale voorwaarde: men moet minstens 10/20 halen op de niet-periodegebonden evaluatie en op de periodegebonden evaluatie om voor dit opleidingsonderdeel te kunnen slagen. Indien dit niet het geval is voor één of beide delen en de met de gewichten berekende totaalscore 10/20 of meer bedraagt, wordt de finale totaalscore 9/20.

Faciliteiten voor werkstudenten

Mogelijkheid tot vrijstelling van aanwezigheid met vervangende opdracht na overleg met verantwoordelijke lesgever. Mogelijkheid tot mondeling examen met schriftelijke voorbereiding op ander tijdstip binnen het academiejaar. Mogelijkheid tot feedback na afspraak tijdens en na kantooruren.

Cursusomvang *(nominale waarden; effectieve waarden kunnen verschillen per opleiding)*

Studiepunten 6.0	**Studietijd 180 u**	**Contacturen**	45.0 u

Aanbodsessies en werkvormen in academiejaar 2022-2023

hoorcollege	21.25 u
werkcollege: geleide oefeningen	7.5 u
practicum	13.75 u

Lesgevers in academiejaar 2022-2023

De Sutter, Bjorn	TWO6	Verantwoordelijk lesgever

Aangeboden in onderstaande opleidingen in 2022-2023	**stptn**	**aanbodsessie**
Educatieve Master of Science in de wetenschappen en technologie (afstudeerrichting informatica)	6	A
Master of Science in Computer Science Engineering	6	A
Master of Science in de industriële wetenschappen: informatica	6	A
Master of Science in de informatica	6	A
Master of Science in de ingenieurswetenschappen: computerwetenschappen	6	A
Uitwisselingsprogramma informatica (niveau master)	6	A

Onderwijstalen

Engels

Trefwoorden

contextvrije grammatica, afleidingsboom, parsers, abstracte-syntaxbomen, semantische analyse, basisblokken, instructieselectie, registertoewijzing, dataverloopanalyse, controleverloopanalyse, codegeneratie, optimalisatie, ondersteuning managed talen, geheugensanering, pijplijnen, inroosteren, technieken voor hoog-niveautalen, just-in-time-compilatie

Situering

Een computer werkt met machinetaal terwijl een programmeur schrijft in een hogere programmeertaal. Compilers vormen de verbinding tussen de programmeur en de computer. Een compiler zal het bronprogramma op een correcte en efficiënte manier vertalen naar machine-instructies. Dit gebeurt in twee grote stappen. in het front-end gedeelte worden lexicale en parsing technieken gebruikt om het programma om te vormen naar een intermediaire voorstelling. In het back-end gedeelte worden verschillende optimisatietechnieken en codegeneratietechnieken toegepast om het programma zo goed mogelijk af te beelden op de onderliggende architectuur. In dit vak worden de theoretische basisconcepten gekoppeld aan de implementatie van een wereldwijd gebruikte compiler en aan een aantal recente ontwikkelingen in het domein.

Inhoud

- Inleiding: Datastructuren voor compilers, Overzicht van fasen in de compilatie
- Lexicale analyse: Tokens en reguliere expressies, Eindige automaten
- Parsing: Contextvrije grammatica, Predictieve en LR-parsers
- Abstracte syntax: Semantische acties, Abstracte syntaxboom
- Semantische analyse: Symbooltabellen, Typechecking
- Activatierecords: Stapelvensters en locale variabelen, Doorgeven van parameters
- Intermediaire code: Voorstellingsboom, Vertaling
- Basisblokken en traces: Canonische bomen, Voorwaardelijke sprongen
- Instructieselectie: Algoritmen voor instructieselectie
- Liveness analyse en registerallocatie: Levensduurvergelijkingen, Graafkleuring en registertoewijzing
- Datastroomanalyse: Datastructuren voor datastroomvoorstelling, Datastroomalgoritmen
- Lusoptimisaties: Reduceerbare grafen, Lustransformaties, Controleverloopanalyse
- Codegeneratie: pijplijnen en inroosteren
- Optimalisatie van geheugengebruik: prefetching, lustransformaties
- Ondersteuning voor hoge-niveautalen: geheugensanering, polymorfisme, klassenhiërarchieën
- Just-in-time compilatietechnieken

- Capita selecta van state-of-the-art compilertechnieken

Begincompetenties

- Kennis van progammeertalen, programmeren in C
- Basiskennis van computerarchitectuur (de software-hardware-interface)

Eindcompetenties

1 De betekenis van de verschillende fasen in een compiler begrijpen.
2 Inzicht hebben in de constructie van automata voor de generatie van lexicale analyzers.
3 Het gebruik van verschillende parsertechnieken (o.m. LL, LR) beheersen.
4 De software voor de generatie van lexicale analyzers en grammaticale parsers gebruiken.
5 Typechecking inbouwen.
6 Een abstracte syntaxboom interpreteren.
7 Controle- en datastroomgrafen opbouwen.
8 Afhankelijkheidsanalyse gebruiken voor levensduur-analyse en code-optimisatie.
9 Instructieselectie-algoritmen vergelijken en toepassen.
10 Registers toewijzen.
11 Intermediaire voorstellingen hanteren voor de generatie van machine-onafhankelijke code.
12 Luseigenchappen analyseren en transformaties toepassen voor code-optimisatie.
13 Dataverloop analyseren en code transformeren voor optimalisatie.
14 Code inroosteren en lussen pijplijnen.
15 Technieken zoals prefetching and tiling toepassen om geheugentoegangen te optimaliseren.
16 Constructies in hoog-niveautalen ondersteunen, zoals geheugensanering, polymorfisme, en overerving
17 Just-in-time compilatietechnieken begrijpen.

Creditcontractvoorwaarde

Toelating tot dit opleidingsonderdeel via creditcontract is mogelijk mits gunstige beoordeling van de competenties

Examencontractvoorwaarde

Dit opleidingsonderdeel kan niet via examencontract gevolgd worden

Didactische werkvormen

Hoorcollege, practicum, werkcollege: geleide oefeningen

Toelichtingen bij de didactische werkvormen

In de hoorcolleges wordt de theoretische onderbouw en interne werking van compilers besproken.
De studenten krijgen voor de practica zeven programmeeropdrachten die ze deels tijdens begeleide sessies (online of in de laptopklassen), deels daarbuiten moeten uitvoeren. Ze moeten hun software, alsook soms een klein verslag, indienen. Ze werken in groepjes van twee. Samenwerking met andere groepen is niet toegestaan. Na het indienen van de werkstukken kunnen de studenten uitgenodigd worden tot een kort evaluatiegesprek waarop ze over hun werkstukken ondervraagd worden om te checken of ze de beoogde competenties verworven hebben. De meeste practica handelen om onderdelen van LLVM, een van de meest gebruikte compilers ter wereld.
In de geleide oefeningen wordt de werking van basisalgoritmes ingeoefend.

Leermateriaal

- Modern compiler implementation in C, Andrew W. Appel with Maia Ginsburg, Cambridge University Press, ISBN 052158390X, LCCN QA76.73.C15.A63, prijs: 51 Euro
- Slides hoorcolleges, te koop aan productiekost VTK

Referenties

- Modern compiler implementation in C, Andrew W. Appel with Maia Ginsburg, Cambridge University Press, ISBN 052158390X, LCCN QA76.73.C15.A63
- LLVM Manual, http://llvm.org/docs/

Vakinhoudelijke studiebegeleiding

Evaluatiemomenten

periodegebonden en niet-periodegebonden evaluatie

Evaluatievormen bij periodegebonden evaluatie in de eerste examenperiode

Schriftelijk examen, openboekexamen

Evaluatievormen bij periodegebonden evaluatie in de tweede examenperiode

Schriftelijk examen, openboekexamen

Evaluatievormen bij niet-periodegebonden evaluatie

Mondeling examen, werkstuk, verslag

Tweede examenkans in geval van niet-periodegebonden evaluatie

Examen in de tweede examenperiode is enkel mogelijk in gewijzigde vorm

Toelichtingen bij de evaluatievormen

De niet-periodegebonden evaluatie bestaat uit zeven werkstukken in de vorm van software, sommige met korte verslagen erbij, die gespreid zijn over het semester. Het verslag en elk werkstuk moet voor de start van het volgende practicum of op een opgegeven datum ingediend worden. Na het indienen van het verslag volgt er mogelijk ook een mondelinge evaluatie waarop de student ondervraagd wordt over de gebruikte methodiek en de behaalde resultaten. Voor de werkstukken werken de studenten soms in groepjes van twee. Aan de hand van een peer assessment over hun bijdrage aan het werkstuk en verslag, en eventueel na een gesprek hierover bij tegenstrijdige assessments, wordt bepaald of beide studenten al dan niet dezelfde punten krijgen.
Niet op tijd ingediende verslagen zonder geldige reden van afwezigheid (zoals een doktersbriefje) worden met een nul gekwoteerd voor dat onderdeel van de niet-periodegebonden evaluatie.
De niet-periodegebonden evaluatie verloopt gelijkaardig voor de tweede examenperiode. De student wordt dan geëvalueerd op basis van nieuwe werkstukken met een gelijkaardige belasting als in de eerste periode, maar moet die wel individueel maken. De verslagen ervan zullen een week voor het examen ingediend moeten worden. Vlak na dat schriftelijk periodegebonden examen volgt dan een mondelinge evaluatie over de werkstukken.
De periodegebonden evaluatie is schriftelijk, en verloopt deels met gesloten boek (theorie en eenvoudige oefeningen) en deels met open boek (oefeningen).

Eindscoreberekening

Verdeling van de punten: 3/5 voor de periodegebonden evaluatie en 2/5 voor de niet-periodegebonden evaluatie. Speciale voorwaarde: men moet minstens 10/20 halen op de niet-periodegebonden evaluatie en op de periodegebonden evaluatie om voor dit opleidingsonderdeel te kunnen slagen. Indien dit niet het geval is voor één of beide delen en de met de gewichten berekende totaalscore 10/20 of meer bedraagt, wordt de finale totaalscore 9/20.

Faciliteiten voor werkstudenten

Mogelijkheid tot vrijstelling van aanwezigheid met vervangende opdracht na overleg met verantwoordelijke lesgever. Mogelijkheid tot mondeling examen met schriftelijke voorbereiding op ander tijdstip binnen het academiejaar. Mogelijkheid tot feedback na afspraak tijdens en na kantooruren.