LUNDS UNIVERSITET
Lunds Tekniska Högskola

Kursplan för läsåret 2010/2011
(Genererad 2010-06-28.)

OPTIMERANDE KOMPILATORER EDA230

Optimising Compilers

Antal högskolepoäng: 7,5. **Betygsskala:** TH. **Nivå:** A (Avancerad nivå). **Huvudområde:** Teknik. **Undervisningsspråk:** Kursen ges på svenska. **Valfri för:** D4, D4is, D4pv, E4, E4pv, Pi4. **Kursansvarig:** Univ.lektor Jonas Skeppstedt, Jonas.Skeppstedt@cs.lth.se, Inst f datavetenskap. **Förkunskapskrav:** EDAF05 Algoritmer, datasrukturer och komplexitet ellerEDA027 Algoritmer och datastrukturer. **Prestationsbedömning:** Tentamen är skriftlig. Ett projekt ingår i kursen. Detta skall fullgöras inom en månad efter kursens slut. Slutbetyget bestäms av betygen på den skriftliga tentamen och ett frivilligt extra projekt. **Övrigt:** Obligatoriska moment: Övningar, datorlaborationer och projektuppgift. **Hemsida:** http://cs.lth.se/eda230.

Syfte

Kursens syfte är att studenterna skall förstå möjligheter och begränsningar hos moderna optimerande kompilatorer, och därigenom veta vad de behöver optimera för hand och vad kompilatorer kan optimera automatiskt.

Mål

Kunskap och förståelse
För godkänd kurs skall studenten

- förstå vilka algoritmer och datastrukturer som används vid implementation av moderna optimerande kompilatorer, samt
- förstå vilka språkkonstruktioner som begränsar optimeringsmöjligheter.

Färdighet och förmåga
För godkänd kurs skall studenten

- kunna analysera benchmarkprogram för att se vilka optimeringsalgoritmer som är lämpliga att implementera, samt
- kunna implementera givna optimeringsalgoritmer på SSA form, kontrollera att de är korrekt implementerade, samt mäta prestandaeffekter.

Värderingsförmåga och förhållningssätt
För godkänd kurs skall studenten

- kunna på egen hand sätta sig in i en vetenskaplig artikel i ämnet, och presentera denna

muntligt för de andra studenterna,
- kunna bedöma ungefär hur lång tid det kan tänkas ta att implementera den presenterade algoritmen, samt bedöma om det kan anses vara värt arbetet, samt
- kunna bedöma om en optimeringsteknik kan vara lämplig att använda för hand i de fall kompilatorn inte kan göra det automatiskt.

Innehåll
Kontrollflödesanalys, dataflödesanalys, beroendeanalys, aliasanalys, elimination av redundans, optimering av loopar, optimering av proceduranrop, registerallokering, schemaläggning av instruktioner, optimering för objektorienterade språk, optimering av minneshierarkin samt vektorisering för processorer med SIMD instruktioner, t.ex. AltiVec och CELL.

Litteratur
Skeppstedt, J: An Introduction to the Theory of Optimising Compilers with Performance Measurements.