Programmering, algoritmik og matematik — en nødvendig sammenblanding?

Oplæg til IDA møde, 29. november 2004

Martin Zachariasen

DIKU

Egen baggrund

B.Sc. i datalogi 1989; Kandidat i datalogi 1995; Ph.D. i datalogi 1998.

Lektor ved Datalogisk Institut, Københavns Universitet

Flerårig erfaring med undervisning i algoritmik.

Inspiration til oplæg

- Egne indtryk/fordomme om ingeniørers programmeringskundskaber.
- Tema om matematik og datalogi i tidsskriftet *Communications of the ACM*, september 2003.

  En række artikler omhandlende emner som f.eks.:

  - Hvorfor er matematik relevant for dataloger (og programmører)?
  - I hvilke sammenhænge er matematisk/algoritmisk tankegang væsentlig for programmører?
  - Hvad er de professionelle programmørers syn på matematik/algoritmik?

Ingeniørers programmeringskundskaber...

Baseret på egne samt andre datalogers indtryk af ingeniørers programmeringskundskaber — er dybest set aldeles uvidenskabelige og ubegrundede fordomme.

\+ Gode problemløsere — får tingene til at virke.

+/– Vælger robuste fremfor originale løsninger.

– Udviklede programmer ikke altid velstruktureret, og de kan være svære at vedligeholde.

Programmering: Konkret eller abstrakt?

Programmering bliver af de fleste studerende opfattet som noget konkret og jordnært, specielt når den sammenlignes med matematikken.

Men alligevel: Programmering er 100 % abstraktion.

Påstand: Programmering er dybest set en matematisk aktivitet. Grundlæggende kompetencer udvikles derfor igennem relevant træning i matematisk tankegang og argumentation.

Programmering som matematisk aktivitet

1. Forståelse af syntaks og semantik
2. Argumentation for korrekthed
3. Argumentation for tids- og pladsforbrug

1. Forståelse af syntaks og semantik

Mange dele af et programmeringssprogs syntaks og semantik er “arvet” fra matematikken.

Syntaks: Tildeling, sammenligning, indeksering.

Semantik: Variable, mængder, kvantorer, funktioner, Boolsk algebra, rekursion.

2. Argumentation for korrekthed

Denne argumentation bør foregå på skrivebordet før programmeringen påbegyndes, men opstår ofte sideløbende i forbindelse med fejlfinding.

Matematiske værktøjer: Matematisk induktion, (løkke) invarianter.

Eksempler:

- Argumentation for korrekthed af rekursive programmer ved anvendelse af matematisk induktion.
- Argumentation for korrekthed af slutresultatet af en iteration ved anvendelse af løkke-invarianter.

3. Argumentation for tids- og pladsforbrug

Denne aktivitet er ikke altid central, men et godt kendskab til tids- og pladskompleksiteten af udbredte datastrukturer og basale algoritmer kan ofte være meget nyttig.

Matematiske værktøjer: Basale matematiske funktioner (polynomier, logaritmer etc.), udregning af summer, rekursionsligninger, kompleksitetsklasser (P og NP).

Eksempler:

- Dynamisk udvidelse af arrays (amortiseret analyse).
- Aktivitetsplanlægning (varianter med forskellig beregningskompleksitet).

Konklusioner

Gentagen træning i matematisk tankegang og argumentation giver bedre programmer:

- Programmer der udnytter sprogets konstruktioner effektivt.
- Programmer der er (mere) korrekte, dvs. fejlfrie.
- Programmer der er velstruktureret, nemme at generalisere og vedligeholde.
- Programmer der har et minimalt tids- og pladsforbrug.

Valg af det første programmeringssprog

Bør blot være et moderne og forholdsvis udbredt programmeringssprog, der indeholder alle de elementer, som er typiske for programmeringssprog.

Bemærk: Det er ikke formålet med en ingeniøruddannelse at give den studerende al den viden og de kompetencer, som det første job kræver.

Bruce et al: “One of the most important goals for a college or university education is to provide the foundation for further learning.”

Personlige anbefalinger

Traditionelle programmeringskurser bør ikke stå alene, men skal tilbydes sammen med træning i relevante matematiske færdigheder.

Disse matematiske færdigheder kan delvis opnås igennem tværgående programmeringsprojekter, der stiller krav om redegørelse for korrekthed og tidsforbrug.

Projekternes tema kan tage udgangspunkt i den aktuelle fagretning (bygning, elektro, kemi, informatik, etc.).