MAI/Opt
Stefan Engevall

Version 0.9
2024-01-11

Kursinformation TAOP62, Optimeringslära fortsättningskurs 6 hp, VT1-2023

1. Kursmål & innehåll

1.1 Mål med kursen

Inom optimeringslära behandlas matematiska teorier och metoder som syftar till att analysera och lösa beslutsproblem som uppkommer inom teknik, ekonomi, medicin, etcetera. Kursen ger, tillsammans med grundkursen, en bred orientering om optimeringslära, med inriktning mot grundläggande teori och metoder för diskreta optimeringsproblem i ändlig dimension, samt en inblick i dess tillämpning för att analysera praktiska optimeringsfrågeställningar. Efter fullgjord kurs skall studenten:

1. kunna redogöra för viktiga klasser av optimeringsproblem och kunna klassificera optimeringsproblem utifrån deras egenskaper, som till exempel i nätverk eller diskreta problem
2. kunna modellera matematiska modeller av enkla optimeringsproblem
3. kunna redogöra för grundläggande begrepp, som till exempel optimalitetsvillkor, svag och stark dualitet, samt giltiga olikheter
4. ha kunskap om och kunna använda grundläggande teori för några vanliga typer av optimeringsproblem, som till exempel dualitetsteori för linjära (nätverks)problem, och ha kännedom om och kunna utnyttja optimalitetsvillkor, som till exempel Bellmans ekvationer, för att avgöra optimalitet för ett en föreslagen lösning
5. kunna redogöra för olika grundläggande algoritmer och kunna sammanfatta principerna bakom algoritmerna för att lösa några vanligt förekommande typer av optimeringsproblem, som till exempel trädsökning för diskreta problem
6. kunna utnyttja relaxeringar, och speciellt Lagrange-dualitet, för att approximera optimeringsproblem, samt kunna stänga in optimalvärden med hjälp av optimistiska och pessimistiska uppskattningar
7. kunna använda vanligt förekommande optimeringsprogramvara för att lösa standardmässiga optimeringsproblem
8. ha viss kunskap om praktiska tillämpningar av optimeringsproblem.

1.2 Förkunskaper

Optimeringslära grundkurs, t.ex. TAOP52.

1.3 Kursinnehåll

Kursen omfattar följande områden:

- Nätverksoptimering: Problem med nätverksstruktur, linjärprogrammering med heltalsegenskap, billigaste vägar, flöden i nätverk, minkostnadsflödesproblem, heltals problem med grafstruktur.
- Heltalsprogrammering: Optimeringsmodeller med diskreta variabler, lösningsmetoder baserade på trädsökning, plansnittning, heuristiker och metaheuristiker.

- Lagrangerelaxation: Grundläggande teori och principer för lösningsmetoder baserade på Lagrangerelaxation. Fokus på hur skattningar kan genereras och tillämpningar på nätverksproblem och heltalsproblem.
- Dynamisk programmering: Problemformulering, optimalitetsprincipen, tillämpningar på lagerhållningsproblem och resursallokeringsproblem.

2. Administration & Organisation

2.1 Kurshemsida

Kursens använder LISAM för senaste info, kursmaterial, mm.

2.2 Lärare/Föreläsare

Stefan Engevall	Examinator, Kursansvarig, Medverkar på samtliga kursmoment Kontaktinformation: E-post: stefan.engevall@liu.se, Telefon: 073-6569326, Rum: Inget permanent rum på Valla. Vid mötesbehov, kontakta först via e-pst eller telefon
Lukas Eveborn	Lektioner (a-klass), Labbar
Nils Åsman	Lektioner (b-klass), Labbar
Jenny Enerbäck	Lektioner (c & ii-klass), Labbar
Gunnar Lidestam	Lektioner (d-klass), Labbar
Anton Hänström	Lektioner (e-klass), Labbar
Maggie Hellström	Lektioner (f-klass), Labbar
Nils-Hassan Quttineh	Redovisningar, Labbar, (Ev. Föreläsningar, Ev. Lektioner)

2.3 Disposition

Kursen omfattar 6 hp = 160 h, och en uppskattning kan vara följande fördelning av tidsanspråk:
Schemalagt, totalt 67 h/student:

- Seminarier: 16 h
- Inspelade Föreläsningar 16 h
- Lektioner: 22 h
- Utförande Laborationer: 4 h
- Redovisningar Laborationer: 4 h
- Tenta: 5 h

Icke Schemalagt: Totalt 93 h/student:

- Teoriinhämtning, förberedelser för, och avrapportering av, laborationer: 19 h
- Teoriinhämtning, egen räkning, inför tenta: 74 h

2.4 Organisation

Kursen bedrivs i form av inspelade seminarier, föreläsningar, lektioner, samt datorlaborationer med tillhörande muntliga redovisningar. Momenten beskrivs utförligare nedan. En kursplanering för kursen, där planerat innehåll för varje tillfälle finns angivet, finns på LISAM senast från och med kursstart.

Formellt sett har kursen inga obligatoriska tillfällen, men det är obligatoriskt att genomföra redovisningar av projekt/labbar – och det underlättar för alla inklusive dig själv – att genomföra dem vid schemalagda tillfällen. Tentan är självklart också obligatorisk att genomföra. Det är synnerligen lämpligt att deltaga på schemalagda labbtider, då detta är de enda tider som man kan få handledning i datorprogrammen som behöver användas. Självklart är övriga undervisningsmoment också högst lämpliga att aktivt deltaga på, för att nå den förståelse som krävs för att klara labbar och tentor.

2.4.1 Zoom-rum/TEAMS

För kursen finns ett zoom-rum uppsatt, som tills vidare är öppet utan lösenord: https://liu-se.zoom.us/j/65739414880. Information om när och hur zoom-rummet används följer senare, beroende på eventuella behov som uppstår.

TEAMS-kanalen för kursen kan också komma att användas.

2.4.2 Seminarier

Seminarierna liknar till stor del vad som är föreläsningar i andra kurser, men djupare teoriinnehåll gås igenom på det som i denna kurs benämns (inspelade) föreläsningar. Det innebär att seminarier i huvudsak ägnas åt att beskriva problemtyper som studeras inom kursen, att gå igenom lösningsstrategier och algoritmer som behövs för att lösa problemen som studeras. Algoritmer exemplifieras med lösning av vissa problem eller delar av problem på tavlan. Viss grundläggande nödvändig teori för lösningsstrategier och algoritmer gås också igenom på seminarierna.

Seminarierna kräver i normalfallet varken specifika förberedelser eller aktivt deltagande, även om det är fördelaktigt att åtminstone översiktligt sätta sig in i vad ett seminarie skall handla om (via kursboken), och att deltaga så aktivt som möjligt på seminarierna.

Eventuellt kan rekommenderade förberedelser att tillkomma under kursens gång, men seminarierna kommer *inte* att förutsätta att rekommenderade förberedelser genomförs.

PPT-underlag för seminarierna planeras att publiceras (i en preliminär version) fredag veckan innan seminariet är schemalagt, dock senast strax före seminariet ges, eventuellt i en uppdaterad version.

2.4.3 Inspelade föreläsningar

De inspelade föreläsningarna ägnas åt mer detaljerad teorigenomgång, och t.ex. förklaring eller härledning av varför en viss algoritm funkar på ett visst sätt, snarare än hur den fungerar. Illustration av hur algoritmer fungerar erhålls på seminarier och lektioner.

De inspelade föreläsningarna är schemalagda[1], men det är självklart upp till var och en när man själv tycker det är lämpligast att titta på inspelningen. Det finns dock rekommenderat före vilket tillfälle det är lämpligt att titta på en viss föreläsning.

Om frågor på innehållet i en föreläsning uppstår, hänvisas till att fråga i samband med annan undervisning (seminarier, eller lektioner/labbar som Stefan eller NisseQ håller), eller mailkontakt med Stefan. Man kan självklart också fråga lektionsassistenterna, även om de inte nödvändigtvis är pålästa på all bakomliggande teori, utan de fokuserar på problemlösning och teori/algoritmer som krävs för att lösa problemen.

[1] Det är också därför det kan ligga en föreläsning parallellt med andra kursmoment. Den som just då deltager i ett annat kursmoment, förväntas hitta andra tillfällen när man kan titta på föreläsningarna.

Eftersom de inspelade föreläsningarna kan ses vid valfritt tillfälle, kommer dessa ibland att ligga på lite obekväma tider, eller i vissa fall parallellt med andra schemalagda aktiviteter. Förväntningen är att varje student anpassar sin personliga planering så att det fungerar på bästa möjliga sätt för just hen.

Huvuddelen av de inspelade föreläsningarna kommer att göras 2024. Filmer och PPT-underlag planeras att publiceras på LISAM (i en preliminär version) fredag veckan innan seminariet är schemalagt, dock senast strax före föreläsningen hålls. Eventuellt kan vissa avsnitt av filmer inspelade under 2021 också användas.

2.4.4 Lektioner

I början av de flesta lektioner går läraren igenom en tentauppgift som relaterar till lektionsinnehållet. Detta kommer i huvudsak att gälla tentafrågor på s.k. G-nivå (se avsnitt 2.6.1). Därefter går läraren igenom några punkter som kan vara bra att tänka på när man löser uppgifterna som rekommenderas för lektionen, men oftast inte lösningar av hela uppgifter. Till lektionerna finns det cirka en inspelad lösning till någon av lektionsuppgifterna. Sådana inspelade uppgifter är avsedda att ersätta lärarledda genomgångar. Vid önskemål kan läraren gå igenom vissa fler/andra uppgifter, eller vissa delar, för hela lektionsgruppen, eller för mindre grupper på schemalagd tid.

Lektionerna är frivilliga, men det finns en mycket stor korrelation mellan de som inte går på lektioner och de som inte klarar tentan.

2.4.5 Laborationer

I kursen ingår två omgångar av laborationer, som innehåller modellering och lösning av nätverks- respektive heltalsproblem. Projekten genomförs i grupper, Team, om högst 6 studenter. All information om projekten återfinns i samband med kursstart på LISAM, där också anmälan sker. Notera att varje Team får projektuppdragen individuellt tilldelade.

Redovisning av labbar/projektarbeten sker muntligt på de tillfällen som i schemat heter Redovisning. **Närvaro vid en av redovisningarna per problemtyp är obligatoriskt.** Anmälan till vilket av respektive 2 pass per problemtyp, görs på LISAM. Vid förhinder ska examinator i förväg meddelas via e-post för att ett extra redovisningstillfälle ska erbjudas. Redovisningen kan också komma att kompletteras med individuell redovisning för examinatorn.

Inget samarbete får förekomma mellan Teamen, när det gäller uppgifterna. Det tillåtna samarbetet är i nivå med en skriftlig tenta, dvs. inget alls. Inom ett Team är det dock både meningen och uppmuntrat att samarbeta med Teamets samtliga uppgifter.

Laborationerna kan ta relativt mycket tid i anspråk, där heltalsproblemet kanske tar 60 % av tiden avsatt för momentet LAB, medan nätverksproblemen kanske tar 40 %. Underskatta inte tiden dessa moment tar. Det är också viktigt att vara väl förberedd enligt instruktionerna, vid labbtillfället som är schemalagt. De schemalagda labbtillfällena är enda tiden man kan få handledning i programmen som används för att lösa labbarna/projektuppgifterna, och är därför fundamentala för att klara uppgifterna, även om närvarokontroll inte sker.

Utanför kursens ordinarie undervisningsperiod (=VT1, 2024), sker ingen handledning relaterat till labbar, överhuvudtaget. Examination som inte färdigställts under ordinarie undervisningsperiod, kan endast ske i anslutning till kursens omtentatillfällen, efter överenskommelse med examinator.

Se även avsnitt 2.6.4 och 2.6.5, om fusk/plagiat, och generativ AI i förhållande till labmomentet.

2.5 Litteratur

Kurslitteraturen består i av följande 2 böcker:

- Lundgren, J., Rönnqvist, M., och Värbrand, P: (2008) *Optimeringslära fortsättningskurs*, 3 uppl. Studentlitteratur, ISBN 9789144053141
- Henningsson, M., Lundgren, J., Rönnqvist, M., och Värbrand, P: (2010) *Optimeringslära fortsättningskurs: övningsbok*, 2 uppl.[2], Studentlitteratur, ISBN: 9789144067605
- Errata till övningsboken, 2 uppl., som finns på LISAM, och uppdateras löpande.

Dessutom:

- Kompletterande material (Lektionshandledningar, Laborationshandledningar, Kompletterande övningar, PPT-bilder, mm) som kan hämtas från LISAM.

Notera specifikt att gamla tentor och tillhörande lösningsförslag inte räknas som kurslitteratur, annat än de uppgifter som specifikt anges som rekommenderade uppgifter. Det innebär att övriga tentor och lösningsförslag inte kvalitetssäkras utifrån ev. oklarheter i frågeställningar eller fel i lösningsförslag; och de används av studenter på ”egen risk”. Gamla tentor och lösningsförslag läggs dock ut på LISAM som en service till studenterna.

2.6 Examination

Kursen har följande två examinationsmoment:

Moment	**Kurspoäng (hp)**	**Betyg**
Laborationer (*LAB1*)	1	U/G
Tentamen (*TEN1*)	5	U/3,4,5

2.6.1 Tentamen TEN1

Tentorna för förra kursomgången (läsåret 22/23) fick ett nytt upplägg jämfört med tidigare år. Äldre tentor är därför helt annorlunda till sin struktur. Tentorna för årets kursomgång (läsåret 23/24), kommer att följa samma upplägg som förra året. Tentafrågorna är på två nivåer, där den första nivån (G-nivån) är frågor av grundläggande karaktär där förväntningen är att man egentligen skall kunna precis allt, för att få godkänt i kursen, men för att ge utrymme för slarvfel eller tolkningssvårigheter gäller att man skall få 75% godkänt på denna del, men också en viss minimipoäng på varje område, vilket automatiskt garanterar betyg 3. För betyg 4 eller 5 krävs, förutom uppfyllda kraven för 3:a, också lösning av ytterligare uppgifter, vanligtvis av svårare eller mer omfattande karaktär, på den så kallade B -nivån. För en något utförligare beskrivning av tentaupplägg hänvisas till ett särskilt dokument för detta, i mappen för tentor, på LISAM.

För G-nivån, och för B-nivån, finns kunskapskrav specificerade i avsnitt 4. Notera dock, att vissa marginella justeringar av kunskapskraven kan ske under kursens gång, delvis beroende på om ev. något innehåll får ett justerat fokus i samband med omarbetningen av seminarier och föreläsningar. En uppdaterad kursinfo, med ev. justeringar av kunskapskraven görs tillgänglig senast vid kursens sista seminarie.

Det är tillåtet att ha med sig ett A4-blad med valfria av dig själv handskrivna anteckningar på båda sidorna.

Övningsboken, samt miniräknare är **ej** tillåtna på tentan.

[2] Upplaga 1 innehåller fler fel i facit, än upplaga 2, även om upplaga 2 också innehåller en del fel. Errata för upplaga 2 kommer att finnas tillgänglig/uppdateras; men det kommer inte att ske för upplaga 1.

Första tentamenstillfället är lördag 23 mars 2024, kl. 8-13.

2.6.2 Laboration LAB1

Kraven för momentet *LAB1*, beskrivs i avsnitt 2.4.5.

2.6.3 Övergripande bedömning

Alla delar av *LAB1* måste avklaras helt under ett och samma läsår (inklusive omtentaperioden i augusti som följer på läsåret). Om bara vissa deluppgifter avklarats och godkänts, måste man göra om hela kursmomentet (hela lab-serien), vid en senare kursomgång.

Kursen är godkänd när samtliga kursmoment som ger kurspoäng är godkända. Kursbetyget är lika med tentamensbetyget.

2.6.4 Fusk och plagiat

Eftersom en stor del av arbetet med labbar sker utan övervakning, är det viktigt att förstå vad som utgör fusk och plagiat. Plagiat är ett sätt att fuska, och är kortfattat när man använder eller lämnar in någon annans arbete (inklusive utdrag ur texter), som om det vore ens egen (t.ex., att inte ange (korrekta) referenser). Det är också att återanvända någon annans text eller presentation, ord för ord, även om du anger referens. Även bilder eller programkod följer samma generella regler som text, dvs. att använda (hela eller delar av) andras programkod eller liknande är också plagiering.

All form av samarbete mellan grupper (dvs. det som kallas Team i labbar, i förberedelser såväl som genomförande) är också fusk, liksom självklart att dela material mellan Team (såvida inte detta sker via examinator). Detta gäller även om man tar hjälp av andra personer, t.ex. tidigare studenter, eller tar del av tidigare studenters arbete.

Om tveksamhet råder, kontrollera för säkerhets skull med examinator om det är tillåtet eller inte.

Fusk vid tentamen är t.ex. all form av kommunikation med andra studenter, eller otillåtna hjälpmedel.

Misstanke om fusk rapporteras till disciplinnämnden, i enlighet med lärarnas instruktioner från universitetsledningen. Observera att om misstanke om fusk förekommer av någon eller några inom ett Team, kan det utsätta hela Teamet för anmälan till disciplinnämnden, så att disciplinnämnden får ta ställning till eventuell olika påföljd för olika medlemmar.

2.6.5 Generativ AI

Det uppmuntras att om man så önskar, använda generativ AI vid inlärningen av kursens olika moment.

För material som ligger till grund för examination gäller följande:

Att skapa matematiska eller grafiska modeller, och överföra dessa till programkod, syftar till att ge en förståelse i hur man dels angriper problem, dels formulerar modeller. Om man på något vis låter en AI hjälpa till i detta (eller utföra hela delen), gör att man går miste om en mycket viktig del i tankeprocessen som skapar lärande och förståelse, genom att man undviker olika överväganden i arbetet och eventuella diskussioner med övriga i Teamet, som kanske inte nödvändigtvis blir en del av det slutliga resultatet. Användning av generativ AI för dessa delar är helt otillåtet, och betraktas som fusk.

När det gäller de muntliga presentationerna (med tillhörande muntliga redovisningar) som skall göras av resultaten, skall utkast på presentation och eventuellt manusskrivande utföras av student(gruppen). Det är dock tillåtet att använda generativ AI för återkoppling på era presentationer (inkl. layout), och manus. Det är dock i sånt fall absolut nödvändigt att i slutet av presentationen beskriva på vilket sätt AI har använts. Det är också viktigt att man i inlämningen på LISAM, lämnar in material som det såg ut innan man skickade till AI-återkoppling. Om inget är nämnt på LISAM, och om detta är inskickat, tolkas det som att ingen AI har använts, och att om misstanke finns att AI trots allt har använts, blir detta automatiskt en misstanke om vilseledande vid examination/fusk.

2.7 Gruppkontrakt

Gruppkontrakt för labben är obligatoriskt att upprätta och lämna in på LISAM, i en särskild inlämning tillgänglig vid kursstart. Gruppkontraktet skall innefatta huruvida, och i sådant fall hur, man avser att dela upp arbetet och hur avser att dela med sig av arbetet till övriga i Teamet. Det är lämpligt att även inkludera andra samarbetsrelaterade punkter i gruppkontraktet.

2.8 Alternativ examination

Möjligheten till examination utanför ordinarie kursomgång är som delvis beskrivits i kapitlen ovan. Tentor examineras bara vid de schemalagda tentamenstillfällena. Laborationer kan examineras utanför ordinarie kursomgång, i samband med de schemalagda tentamenstillfällena, men ingen handledning sker utanför ordinarie kursomgång.

Alternativ examination (t.ex. muntlig i stället för skriftlig, eller tvärt om) är normalt endast möjligt om man har intyg från koordinator för lika villkor som styrker att alternativ examination är nödvändig.

3. Undervisningsplanering

En undervisningsplan finns i form av ett Excelark, och uppdateras kontinuerligt på LISAM. Notera att det innehåller flera flikar, och möjligheten att ”filtrera” för att se endast ett urval av rader.

4. Kunskapsområden

Nedan följer vilka kunskaper som är nödvändiga för att dels bli godkänd på tentan (3), dels för att nå ett högre betyg (4 eller 5).

Observera att ytterligare kunskaper än de som listas nedan; framför allt från tidigare kurser, eller från vad som bedöms som enklare moment eller problemtyper till viss del inom ramen för denna kurs, också är nödvändiga utan att det explicit nämns, och att detta får anses falla inom ramen för den sista punkten på varje område (se nedan). Som exempel på kunskaper som kan krävas på G-nivå, utan att specifikt omnämnas, är grafisk lösning av LP/HP-problem (som kan vara delsteg inom andra metoder, där examinationen ligger på förståelsen för slutsatser – även om det kan innebära att om man inte kan de enklare momenten så klarar man inte uppgiften), eller lösning av ett Heltals-problem i flera variabler med ett kappsäcksvillkor (genom sortering av variablerna). Inte heller omnämns ju ekvationslösning, addition, eller beräkning av olikheter, som är eller kan vara en del av problemlösningen som faktiskt krävs.

En tydligare mappning av kunskaper mot kursmål kommer att göras under kursens gång, 2024 (eventuellt i ett annat dokument).

4.1 Nätverk

4.1.1 Kunskaper på G-nivå

- Lösa ett billigaste-väg-problem med Dijkstras algoritm
- Identifiera kritisk väg/kritisk linje, och förstå vad den innebär, i ett projektnätverk
- Förstå och motivera möjliga och mest lämpade metoder (Bellmans ekv., Dijkstras, Fords) i givna nätverk (utan att lösa), inklusive relaterade teorifrågor
- Modellera Minkostnadsflöden med kostnader, undre gränser (>0), övre gränser (≤M).
- Modellera Minkostnadsflöden med intäkter
- Hantera modellering minkostnadsflöden där maximal kapacitet ≠ total efterfrågan (välja rätt nodstyrkor, samt ”överskottsbågar” eller ”cirkulärt flöde”
- Hantera justering av modeller för att ta hänsyn till kostnader, undre gränser eller övre gränser på flödet genom en nod
- Hantera justering av modeller där alternativa kostnader finns för samma väg.
- Identifiera om en given lösning är en tillåten baslösning eller varför den inte är det i ett MKF-problem
- Identifiera ett (unikt) basträd i ett MKF-problem, inkl. att förstå varför ett unikt basträd inte kan identifieras
- Beräkna nodpriser i en tillåten baslösning i MKF-problem (utan knorrar)
- Beräkna reducerade kostnader i en tillåten lösning i MKF-problem, och utifrån dessa, både kunna avgöra om lösningen är optimal, och om inte, vilken som skulle vara en (unik) inkommande basbåge
- Givet en inkommande basbåge i ett MKF-problem, identifiera en (unik) utgående basbåge, samt beräkna nätverkets nya flöde.
- Känslighetsanalys MKF: Kostnadskrav på en ny båge för att den skall vara intressant att använda eller inte
- Känslighetsanalys MKF: Kostnadskrav/kostnadsförändringar på en existerande båge för att den skall vara intressant att använda eller inte
- I detta ingår självklart att kunna förstå begreppen som relaterar till uppgifterna (som ETT exempel förstå begreppet nodpris)!

4.1.2 Kunskaper på högre-betygs-nivå

- Förstå och redogöra för kopplingar mellan LP-modeller och Billigaste-väg-problem respektive Minkostnadsflödesproblem
- Förstå och hantera kopplingar mellan optimalitetsvillkoren för LP-problem och Bellmans ekvationer
- Förstå och diskutera konsekvenser av heltalsegenskap för nätverksproblem
- Modellera ett ”helt” MKF-problem, inklusive justeringar som anges under grundläggande kunskaper, och med ev. ytterligare ”knorrar”
- Lösa ett billigaste-väg-problem med Bellmans ekvationer, eller med Fords algoritm
- Lösa billigaste-väg-typ-problem med alternativa målfunktioner; t.ex. min-max, multiplikativ målfunktion
- Modellera och lösa ett projektnätverksproblem

- Klara att göra hela iteration(er) i MKF-problem samt även där t.ex. basträdet inte är unikt, liksom där inkommande eller utgående basbåge inte är unik, eller t.ex. samma båge är utgående som inkommande, eller där reducerade kostnaden för en icke-basbåge är 0.
- Förstå om alternativa optimallösningar finns i ett MKF-problem, och i sådant fall hitta en sådan.
- Tillämpa det faktum att det finns en frihetsgrad i valet av ett nodpris
- Använda Fas1-metoden för Simplex för nätverk, för att hitta en tillåten lösning; både modelleringsmässigt och att genomföra en (del)iteration.
- Känslighetsanalys MKF: Kostnadskrav på en icke-basbåge för att den inte längre skall ha sitt nuvarande min- eller maxflöde
- Känslighetsanalys MKF: Kostnadskrav på en basbåge för att baslösningen skall vara oförändrad
- Känslighetsanalys MKF: Hitta nya flödet i ett nätverk om en ny båge läggs till, med en viss kostnad.
- Känslighetsanalys MKF: Hitta nytt flöde om nodstyrkor ändras i ett par noder.
- Känslighetsanalys MKF: Hitta nytt flöde och kostnadsskillnad om kapaciteten i en båge ändras (undre gräns minskar eller övre gräns ökar)
- Känslighetsanalys MKF: Totalkostnadsskillnaden i nätverket om kostnaden på en basbåge eller icke-basbåge ändras

4.2 Heltal

4.2.1 Kunskaper på G-nivå

- Formulera relationer mellan binära och linjära variabler, även med flera index
- Formulera logiska villkor mellan binära variabler, även med flera index
- Matematisk formulering av en fullständig enkel matematisk heltalsmodell (utan knorrar) – definition av variabler, formulering av målfunktion, bivillkor (inklusive krav på variabler), med variabler med max 1 index.
- Förstå om en ritad eller ritningsbar olikhet är en giltig olikhet eller inte
- Identifiera ett Gomorysnitt (heltal respektive fraktionellt) ur en simplex-tablå (utan knorrar, som t.ex. negativa koefficienter), och illustrera detta grafiskt i ursprungliga variabler
- Identifiera om en ny tablå är optimal, efter att en given ursprungstablå är given, ett tillagt Gomorysnitt är givet, och en ny tablå är given.
- I ett givet LDD-sökträd, identifiera vilka grenar som kan kapas och inte, samt identifiera pessimistiska och optimistiska skattningar för problemet
- I ett problem som löses med LDD, identifiera vilka villkor som bör genereras i en förgrening av en lösning funnen i en viss nod
- Lösa Billigaste uppspännande träd (även med knorrar, som t.ex. att en viss båge **måste ingå** i lösningen) till optimum med valfri metod av Prims eller Kruskals
- Lösa handelsresandeproblem med närmsta-granne-heuristiken
- Utföra enkel lokalsökning utan knorr i ett handelsresandeproblem
- Förstå skillnaden mellan/kopplingen mellan lokalt och globalt optimum
- I detta ingår självklart att kunna förstå begreppen som relaterar till uppgifterna (som ETT exempel förstå begreppet binär variabel)!

4.2.2 Kunskaper på högre-betygs-nivå

- Matematisk formulering av mer avancerad komplett heltalsmodell, inklusive variabler med flera index i ett (blandat) heltalsproblem
- Hantering av lagerbalansvillkor med fasta kostnader, och flera index.
- Förstå och resonera kring olika formuleringars styrka
- Förstå och resonera kring konvexa höljet av ett tillåtet område
- Identifiera och illustrera giltiga olikheter, samt förstå om en giltig olikhet är en fasett eller inte
- Identifiera ett eller flera Gomorysnitt ur en simplextablå, eventuellt med knorrar, samt lägga till ett Gomorysnitt, genomföra en iteration med duala simplexmetoden.
- Identifiera övertäckningar resp. minimala övertäckningar i kappsäcksvillkor, samt härleda giltiga olikheter ur dessa
- Kunna lyfta koefficienter i bivillkor för att generera starkare giltiga villkor
- Lösa ett problem med hjälp av LDD, och utnyttja problemspecifika egenskaper (som t.ex. om vi vet att vi ”enkelt” hittar tillåtna lösningar t.ex. med hjälp av avrundningar, eller om vi vet att optimala målfunktionsvärdet alltid är heltaligt, oavsett de lösta LP-problemens målfunktionsvärde)
- Aktivt använda kunskapen att senaste tillagda villkoret i LDD, alltid är aktivt i ”nästa” problem (vilket t.ex. gör att vi grafiskt kan lösa problem i 3 variabler).
- Använda trädsökning för att lösa eller dra slutsatser om problem med bara binära variabler (dvs. där grenarna kan fixeras till att i ena grenen är $x_j = 0$ och i andra grenen är $\boldsymbol{x_j = 1}$
- Lösa varianter av träd-problem eller handelsresandeproblem med icke-standardmässiga heuristiker (som t.ex. kan presenteras på tentatillfället, t.ex. som anpassningar på ”standard”-heuristiker).
- Lägga till subturförbjudande villkor i en TSP-lösning där subturförbjudande villkor relaxerats
- Identifiera delproblem/grenar i trädsökningsstrategi, i en TSP-lösning där subturförbjudande villkor relaxerats
- Lösa handelsresandeproblem med Närmsta insättning, Längst-bort insättning eller billigaste insättning
- Genomföra en eller ett par iterationer av Tabusökning, med givna definitioner
- Förstå och beskriva principer för genetiska algoritmer respektive simulated annealing, och kunna tillämpa dessa lösnings principer på ett mindre, enklare problem

4.3 Lagrange

4.3.1 Kunskaper på G-nivå

- Formulera Lagrangefunktionen för blandade heltalsproblem (MILP), med ett eller flera relaxerade bivillkor
- Formulera Lagrangeduala funktionen $h(v)$ för MILP, med ett eller flera relaxerade bivillkor
- Formulera alla separata Lagrangeduala subproblem då huvudproblemet är ett MILP, med ett eller flera relaxerade bivillkor
- Lösa Lagrangeduala (sub)problem när endast ett villkor har relaxerats, i ett huvudproblem som är ett MILP, för givna värden på Lagrangemultiplikatorn v
- Avgöra om en funnen lösning x, är tillåten eller inte i ursprungsproblemet i ett MILP

- Beräkna (olika) värden på $h(v)$ (för olika v) och olika lösningar $\boldsymbol{x}$ (både tillåtna och otillåtna) för att dra slutsatser om (gränser för) z^* och v^*, när endast ett villkor har relaxerats i ett MILP
- I detta ingår självklart att kunna förstå begreppen som relaterar till uppgifterna (som ETT exempel förstå begreppet LBD)!

4.3.2 Kunskaper på högre-betygs-nivå

- Samma kunskaper som på G-nivå, men kunna göra detta även för nätverksproblem (TSP, MST, BVP, MKF), och/eller när fler än ett bivillkor relaxeras.
- Förstå och tillämpa en lösningsstrategi i form av en iterativ process där Lagrangemultiplikatorerna uppdateras, enligt angivna sökriktnings- och steglängdskriterier
- Rita men även tolka och dra slutsatser av grafiska illustrationer av $h(v)$
- Förstå begreppen subgradient och subdifferential, och även hur man kan identifiera att $0 \in \partial h(\bar{v})$, samt vilken tolkning man kan göra av det
- Förstå vad icke-styrbarhet kan få för konsekvenser när man jobbar med Lagrangerelaxering som metod

4.4 Dynamisk programmering

4.4.1 Kunskaper på G-nivå

- Lösa ett formulerat partiformnings/lagersyrningsproblem eller ett kapsäcksproblem med Dynamisk programmering utan osäkerhet, med bakåtrekursion (dvs. både ”fylla i tabeller” med nödvändiga beräkningar, och göra uppnystning för att hitta optimallösning), i problem med unik optimallösning
- Förstå när och varför det är fördelaktigare att använda bakåtrekursion i stället för framåtrekursion
- I detta ingår självklart att kunna förstå begreppen som relaterar till uppgifterna (som ETT exempel förstå begreppet överföringsfunktion)!

4.4.2 Kunskaper på högre-betygs-nivå

- Rita, förstå och tolka kopplingen mellan Nätverk och DynP
- Förstå hur Bellmans ekvationer används (implicit) i lösning av DynP
- Formulera ett DynP-problem från en text, med korrekta begrepp – även andra typer av problem än rena partiformnings/lagerstyrningsproblem
- Lösa ett DynP som kan vara andra problem än rena partiformnings/lagerstyrningsproblem (t.ex. kappsäcksproblem, men även andra problemtyper), som dessutom dels kan innehålla osäkerhet, dels ha alternativa optimallösningar
- Formulera och lösa DynP-problem med mer komplexa målfunktioner