AALBORG UNIVERSITET

Institut for Elektroniske Systemer
Afdelingen for Datalogi

Fredrik Bajers Vej 7, DK-9220 Aalborg Øst

Titel:

Tænkeboks — vha. bayesianske netværk og spilteori

Projektperiode:
1. september - 22. december 1997

Semester:
Dat3/F7D

Projektgruppe:
C1-213

Projektgruppens deltagere:
Søren Andersen
Lisa Elgaard
Michael Höhle
Brian Jensen
Brian Kristiansen
Søren Rolander Mogensen

Vejleder:
Dennis Nilsson

Oplagsantal: 9

Sideantal: 112

SYNOPSIS:

Denne rapport omhandler modellering af spillet tænkeboks vha. bayesianske netværk og influensdiagrammer. Et bayesianske netværk benyttes til at modellere en simpel udgave af tænkeboks, for at opnå en forståelse af den metode, der benyttes i modelleringen vha. bayesianske netværk. Herefter opstilles et influensdiagram, for denne simple udgave, som vi benytter til at finde en optimal modstrategi, givet at vi kender modspillerens strategi. Vi foretager en formel analyse af propagering i bayesianske netværk, hvor vi kommer ind på, hvordan Hugin propagerer vha. junction træer. Ud over modellering af spillet introducerer vi emnet spilteori og benytter denne teori på den simplificerede version af spillet. Vi ser ligeledes på, hvordan vi kan benytte spilteori på en lille udvidelse af det simplificerede tænkeboks. Endeligt vil vi analysere, hvorledes vi kan håndtere udvidelser og den fulde udgave af tænkeboks. I forbindelse med den fulde udgave overvejes kompleksiteten samt mulige simplifikationer.

Forord

Denne rapport er udarbejdet i efteråret 1997 af projektgruppe C1-213 på Dat3/F7D semestret ved afdelingen for datalogi på Aalborg Universitet. Det overordnede tema for projektet er *modellering*, hvor vi har valgt at arbejde med beslutningsstøttesystemer. Projektenhedskurset er af denne grund "Beslutningsstøttesystemer 1".

Rapporten henvender sig fortrinsvist til vejleder, censor og medstuderende.

Nummerering af figurer og tabeller er på formen $x.y$, hvor x angiver, hvilket kapitel figuren er i, og y er figur- eller tabelnummeret. Litteraturhenvisninger er på formen [<forfatter>, <udgivelsesår>] eller [<forfatter>, <udgivelsesår>, s. <side>], hvor <forfatter>, <udgivelsesår> henviser til en bog eller artikel fra litteraturlisten på side 109. Indeks er i teksten angivet med fed skrift, der hvor definitionen af ordet findes, og samtlige indeks er opstillet i indekslisten på side 111.

Notation mht. bayesianske netværk er hentet fra [Jensen, 1996], mens grafteoretisk notation er i overensstemmelse med [Chartrand og Oellermann, 1993].

Vi takker Sven Vestergaard, adm. direktør for Hugin Expert A/S, der har stillet en fuld version af Hugin til PC til rådighed for os.

Søren Andersen	Lisa Elgaard
Michael Höhle	Brian Jensen
Brian Kristiansen	Søren Rolander Mogensen

Indhold

Kapitel 1

Indledning

Denne rapport afspejler projektarbejdet for projektgruppe C1-213 ved Aalborg Universitet i efteråret 1997. Med projektet ønsker vi at få et indblik i beslutningsstøttesystemer. Specielt ønsker vi at modellere spillet tænkeboks vha. bayesianske netværk og influensdiagrammer samt implementere disse i modelleringsværktøjet Hugin.

Da vi ikke tidligere har stiftet bekendtskab med Hugin, ønsker vi gennem projektet at opnå færdigheder i anvendelse deraf. Vi ønsker at have en forståelse for, hvad der sker i det underliggende system, når vi vælger at propagere.

For at få en god kvalitet af vores modeller er det nødvendigt med en nærmere analyse af spillet tænkeboks. Til denne analyse anvender vi spilteori, der er en matematisk teori, hvormed vi kan finde de bedste strategier for deltagerne i et spil. Vores håb er, at vi kan finde en strategi til spillet tænkeboks, hvormed vi kan forvente at vinde i det lange løb.

1.1 Spillet tænkeboks

I denne rapport modelleres forskellige versioner af spillet tænkeboks. Versionerne tager alle udgangspunkt i det oprindelige spil, hvorfor det er nødvendigt at fastlægge vores opfattelse af spillets regler.

Tænkeboks er et terningspil for mindst 2 spillere. Spillet begynder med, at de n spillere har fire terninger hver, som de slår med, således at kun spilleren selv ser slaget.

En spiller foretager en **melding** ved at byde på hvor mange ens terninger, han tror, der er af et bestemt eller ubestemt antal øjne i alt på bordet. Værdien for budene er som følger:

$$m \cdot \text{ens} < m \cdot 1 < m \cdot 2 < m \cdot 3 < m \cdot 4 < m \cdot 5 < m \cdot 6 < (m+1) \cdot \text{ens}$$

Det vil f.eks. sige, at tre treere er mindre end tre firere, og tre ens er mindre end fire ens.

I tænkeboks er alle ettere jokere, dvs. at en etter kan have værdien af en vilkårlig side på terningen. Viser alle t terninger for en spiller i et slag $1, 2, 3, .., t$, har spilleren $t + 1$ jokere. Et sådan slag kaldes en **jokerserie**.

En melding er opfyldt, hvis der mindst er det antal ens, der blev budt. Bliver der f.eks. meldt $4 \cdot 3$, og der er fem treere i spillet, da er meldingen opfyldt.

Spillet startes ved, at en vilkårlig spiller foretager sin melding. Herefter byder spillerne op efter tur, ved at melde en højere melding, end den forrige spiller meldte eller ved at "løfte". Når en spiller løfter, betyder det, at alle spillere skal vise, hvad de har slået, og det undersøges, om den sidste melding, før der blev løftet, er opfyldt. Hvis meldingen er opfyldt, har den spiller, der løftede, tabt. Ellers har spilleren, der foretog meldingen, tabt.

Taberen fra sidste runde begynder næste runde, og alle spillere, på nær taberen, lægger én terning fra. Taberen af spillet er den spiller, der som den sidste, har en eller flere terninger tilbage.

1.2 Oversigt af rapport

Det komplette tænkeboks er så kompliceret at modellere, at vi tager udgangspunkt i en meget simplificeret form af tænkeboks. Dette simple spil analyseres vha. bayesianske netværk og spilteori.

Vi introducerer bayesianske netværk i Kapitel 2. Hernæst opstiller vi et generelt netværk for en simplificeret version af spillet tænkeboks, og foretager en analyse af dette netværk. Ud fra denne analyse opstilles et influensdiagram for spillet set ud fra en enkelt spillers synspunkt.

Da vi gennem hele projektforløbet benytter programmet Hugin, ser vi i Kapitel 3 nærmere på, hvad der foregår bag programoverfladen. Vi ønsker at vide, hvordan evidens propageres gennem netværket, og definerer i den anledning junction træer, som benyttes af Hugin til propagering.

I Kapitel 4 opstilles de indledende definitioner til spilteori, specielt defineres Nash ligevægte. Den opstillede teori benytter vi i Kapitel 5 til at analysere det simplificerede tænkeboks, som vi beskrev i Kapitel 2.

For at kunne vurdere, hvordan udvidelser af den simple udgave påvirker spillet, og hvordan det skal håndteres af spilteorien, udvider vi i Kapitel 6 den simple version med en ekstra side på terningen.

Efter at have modelleret den simplificerede version af tænkeboks vha. influensdiagrammer, overvejer vi i Kapitel 7, hvorledes disse kan anvendes til at beskrive en fuld version af tænkeboks. Desuden undersøges, hvilken kompleksitet en sådan model vil få.

Kapitel 2

Model af simpel tænkeboks

I dette kapitel vil vi kort gennemgå de elementære elementer i bayesianske netværk og influensdiagrammer. Ønskes der en grundigere og mere formel introduktion, henvises til [Jensen, 1996].

Først simplificerer vi reglerne for det oprindelige tænkeboks og konstruerer et bayesiansk netværk, der beskriver spillets forløb. Dernæst opbygges et influensdiagram, således at spillet ses fra én spillers synsvinkel.

Begrundelsen for at se på en simplificering frem for den fulde udgave af tænkeboks er, at det gør det lettere at overskue problemstillingerne så som opbygning af model og opstilling af sandsynligheder. Erfaringerne fra den simple model kan senere benyttes i opbygningen af en større model for spillet.

2.1 Bayesianske netværk

Når der skal opstilles et **bayesiansk netværk** for et spil som tænkeboks, er det vigtigt, at modellen er så realistisk som mulig. Dette gøres ved at se på de aktører, der er involverede, samt hvilke handlinger de kan udføre. Konsekvenserne af handlingerne og deres indflydelse på andre handlinger skal ligeledes modelleres i netværket. Aktørernes handlinger indbygges i netværket vha. **variable**, der beskriver deres **tilstande**.

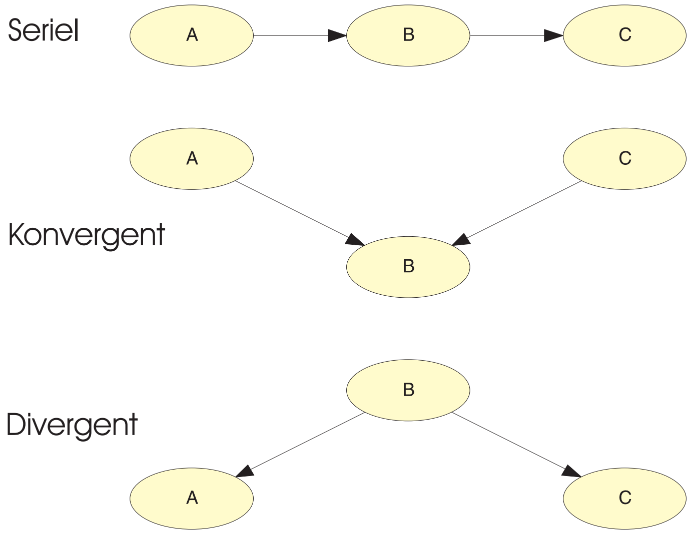
Variablene i bayesianske netværk er alle **stokastiske variable**. De variable, der ikke kan observeres direkte, men som ønskes bestemt, kaldes **hypotesevariable**. Variable, der kan observeres, kaldes **informationsvariable**. Ud over disse to typer af variable findes også **mellemvariable**, der benyttes af forskellige årsager, f.eks. til at afspejle uafhængigheder i et netværk eller mindske antallet af forældre til en knude, hvilket mindsker kompleksiteten af det bayesianske netværk.

Det bayesianske netværk benyttes ved, at en eller flere informationsvariable observeres, og deres tilstand indsættes som **evidens** i netværket. En variabel siges at være **instantieret**, når der er indsat evidens på den. Herefter opdateres sandsynlighederne for de variable, der afhænger af den indsatte evidens. Denne process kaldes **propagering**[1].

[1] I Kapitel 3 findes en formel gennemgang af propagering.

Man siger, at der eksisterer en afhængighed mellem to variable i et bayesiansk netværk, hvis der er en pil imellem disse. Men faktisk er det de manglende pile, der er interessante, da det er dem, der beskriver de betingede uafhængigheder imellem variablene. For at finde disse uafhængigheder skal vi se på, hvordan variablerne er forbundet i netværket.

To variable siges at være **serielt forbundet**, hvis der er en vej, med minimum en mellemliggende variabel, fra den ene variabel til den anden. Hvis de to variable har et fælles barn, siges de at være **konvergent forbundet**. Og hvis to børn har en fælles forælder, siges de at være **divergent forbundet**. De forskellige typer af forbindelser ses på Figur 2.1.



Figur 2.1: Seriel, konvergent og divergent forbindelse i et bayesiansk netværk.

Serielt forbundne variable er uafhængige givet evidens på en mellemliggende variabel. Konvergent forbundne variable er uafhængige givet, at der ikke er evidens på det fælles barn eller en af dets efterkommere. Og divergent forbundne variable er uafhængige givet evidens på deres fælles forældre.

For enhver variabel V_i i netværket angiver en pil fra en af dens forældre, V_p, til variablen den betingede sandsynlighed, $P(V_i|V_p)$. Hvis variablen V_i indeholder to tilstande, A og B, og variablen V_p ligeledes indeholder to tilstande, C og D, vil den betingede sandsynlighed være på formen:

$$P(V_i|V_p) = \begin{array}{|c|cc|} \hline & A & B \\ \hline C & 0.5 & 0.5 \\ D & 1 & 0 \\ \hline \end{array}$$

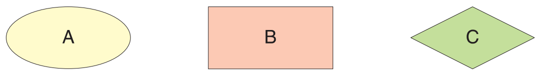
Tabellen læses som, at givet V_p er i tilstand C, da er der 50% sandsynlighed for, at V_i er i hhv. tilstand A eller B. Hvis V_p er i tilstand D, da er V_i i tilstand A med 100% sandsynlighed.

Har en variabel mere end én forælder, er det ikke nok kun at kende den betingede sandsynlighed fra hver af pilene, der fører til variablen. Dette skyldes, at disse sandsynligheder ikke giver information om, hvorledes interaktionen mellem forældrene påvirker variablen. For at opnå dette er det nødvendigt at kombinere sandsynlighederne fra de enkelte pile med information om interaktionen. For en variabel V_i specificeres derfor sandsynligheden $P(V_i|pa(V_i))$, hvor $pa(V_i)$ er mængden af V_i's forældre.

Disse sandsynligheder skal ligesom modellen beskrive den virkelige verden i så stor overensstemmelse som muligt. Derfor bliver sandsynlighederne estimeret ud fra data og eksperters erfaringer. Herefter kan modellen benyttes til at indsætte evidens på informationsvariablene, således at denne evidens kan propageres gennem netværket, hvorefter sandsynlighederne for hypotesevariablene givet denne evidens kan aflæses.

2.2 Influensdiagrammer

Influensdiagrammer er en videreudvikling af bayesianske netværk, hvor modellen betragtes ud fra én beslutningstagers synspunkt. Hvor bayesianske netværk kun indeholdt stokastiske variable, indeholder influensdiagrammer tre typer af knuder: stokastiske variable, beslutningsvariable og nyttefunktioner. Den grafiske repræsentation af de tre knudetyper ses på Figur 2.2.



Figur 2.2: Grafisk repræsentation af knudetyper: A er en stokastisk variabel, B en beslutningsvariabel og C en nyttefunktion.

Beslutningsvariable svarer til, at beslutningstageren skal foretage en beslutning. Enhver pil, der fører til en beslutningsvariabel, indikerer, at tilstanden af den variabel, pilen kommer fra, er kendt, når beslutningen skal træffes.

Nyttefunktioner angiver den forventede nytte, beslutningstageren vil få, som funktion af tilstandene i variablene i dens forældremængde. De pile, der fører til en nyttefunktion, angiver, at funktionen afhænger af den variabel, pilen kommer fra.

Beslutningstageren kan benytte influensdiagrammet til at afgøre, hvilke beslutninger der skal vælges. De beslutninger, beslutningstageren vil vælge, er dem, der maksimerer den forventede nytte. Disse kan findes ud fra de nytte- og sandsynlighedsfunktioner, der er specificeret i influensdiagrammet.

2.3 Beskrivelse af spillet ved hjælp af bayesianske netværk

I dette afsnit analyserer vi en simplificeret udgave af spillet tænkeboks. Vi opstiller en model ud fra, hvordan spillet kan opfattes af en observatør, dvs. en person der ikke deltager i spillet. Herefter simplificerer vi modellen ud fra generelle overvejelser om det simple tænkeboks. Formålet med dette bayesianske netværk er, at vi kan anvende det som udgangspunkt til et influensdiagram.

Reglerne i den simple udgave af tænkeboks er grundlæggende de samme som beskrevet i afsnit 1.1. Forskellen i forhold til den fulde udgave af tænkeboks består i, at dette spil kun har to spillere, benævnt Arne og Bente, med hver én terning, der kun har to sider. Spillet indeholder ingen jokere, og der kan ikke meldes m ens. Det vil sige, at meldingerne kan være 1, 2, 2·1, 2·2 eller løft. Det simplificerede tænkeboks har følgende form:

Aktører: Arne og Bente.

Handlinger: Arne og Bente har to forskellige typer af handlinger, de kan udføre. Enten kan de slå med en terning, eller også kan de foretage en melding.

Af ovenstående kan det ses, at vi behøver en variabel, AS, til Arnes slag, en variabel, BS, til Bentes slag og et antal variable til deres meldinger. Af meldingerne for spillet ses det, at det højeste antal gange, Arne og Bente kan melde i et spil, er hhv. tre og to, nemlig med følgende meldingssekvens:

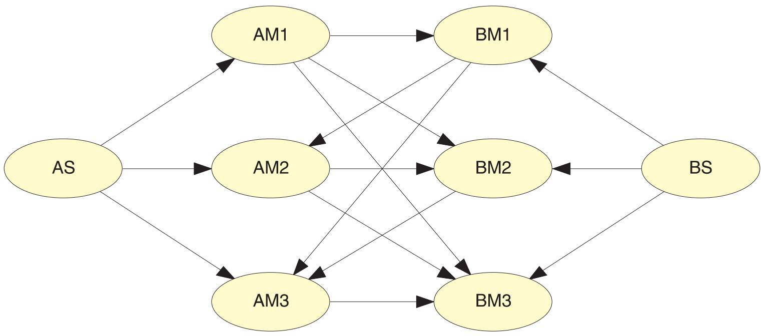
$$\begin{array}{ccccccccc} \text{A} & & \text{B} & & \text{A} & & \text{B} & & \text{A} \\ 1 & \rightarrow & 2 & \rightarrow & 2\cdot 1 & \rightarrow & 2\cdot 2 & \rightarrow & \text{løft} \end{array}$$

Med fem variable til håndtering af meldingerne tillades det imidlertid ikke, at Bente begynder spillet. Dette problem kan løses ved, at vi tillader Arne at melde 0, hvormed Bente skal starte. Det vil sige, at hvis Arne melder 0, skal denne melding ikke tages i betragtning i de efterfølgende meldingsrunder, og Bente skal i sin første melding kun melde ud fra sit eget slag. Med denne udvidelse er det nødvendig med seks variable til håndtering af meldingerne.

Vi har til det simple tænkeboks i første omgang konstrueret en model, der beskriver den virkelige verden så præcist som mulig. Derfor påvirker variablene hinanden som vist på Figur 2.3, hvor Arnes første melding er AM_1, anden melding er AM_2 og tredje melding er AM_3. Bentes meldinger bliver således BM_1, BM_2 og BM_3.

Tilstandende i variablene er vist i Tabel 2.1.

Som det ses af Figur 2.3, har vi valgt, at alle Arnes meldinger har indflydelse på alle Bentes efterfølgende meldinger. Dette er valgt, da Bente på denne måde kan sammenligne Arnes tidligere meldinger med den aktuelle melding og derved afgøre,



Figur 2.3: Bayesiansk netværk for modellen.

Variabel	Tilstande
AS	1, 2
AM_1	0, 1, 2, 2·1, 2·2
AM_2	2, 2·1, 2·2, løft
AM_3	2·2, løft
BS	1, 2
BM_1	1, 2, 2·1, 2·2, løft
BM_2	2·1, 2·2, løft
BM_3	løft

Tabel 2.1: Tilstandene for variablene i den simple model for spillet tænkeboks.

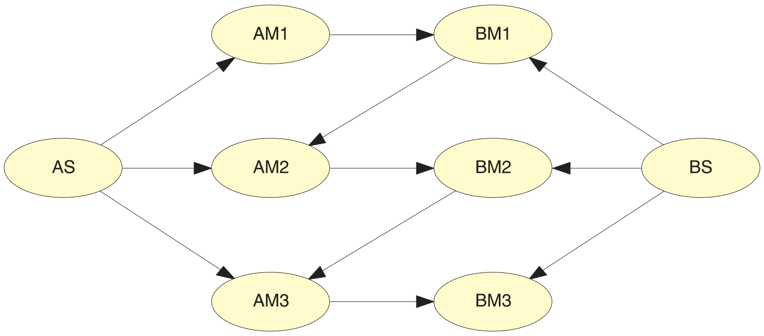
om der er uoverensstemmelse mellem hans meldinger. I den simple model er dette imidlertid overflødigt, hvilket kan ses af følgende eksempel: Hvis Arne begynder spillet og melder 1 i første meldingsrunde, så er den mindste mulige melding i hans anden meldingsrunde 2·1. Dette er ensbetydende med, at Bente løfter uanset, hvad hun har slået. Grunden til dette kan ses af følgende overvejelser:

$\mathbf{AM_2 = 2 \cdot 1, BS = 1}$: Bente ved med sikkerhed at der ikke er 2·2, da hun selv sidder med en etter, derfor vil hun tabe, hvis hun melder dette.

$\mathbf{AM_2 = 2 \cdot 1, BS = 2}$: Bente vinder, hvis hun løfter, da der ingen mulighed er for, at Arnes melding eksisterer.

Det eneste tidspunkt, Bente skal overveje at tage AM_1 i betragtning og melde andet end løft ved BM_2, er, når hun begynder spillet, dvs. AM_1 er 0. Da vi indførte denne melding, blev det netop fastslået, at denne melding ikke skulle tages i betragtning i de efterfølgende meldingsrunder af spillet.

Med ovenstående to eksempler som begrundelse kan modellen på Figur 2.3 simplificeres til modellen på Figur 2.4. I dette bayesianske netværk har variablene for spillernes meldinger færre forældre, hvilket er en stor fordel, da dette reducerer antallet af sandsynligheder, der skal specificeres.



Figur 2.4: Reduceret bayesiansk netværk for modellen.

I det følgende afsnit vil vi omforme det simplificerede bayesianske netværk til et influensdiagram, således at spillet betragtes ud fra Bentes synsvinkel.

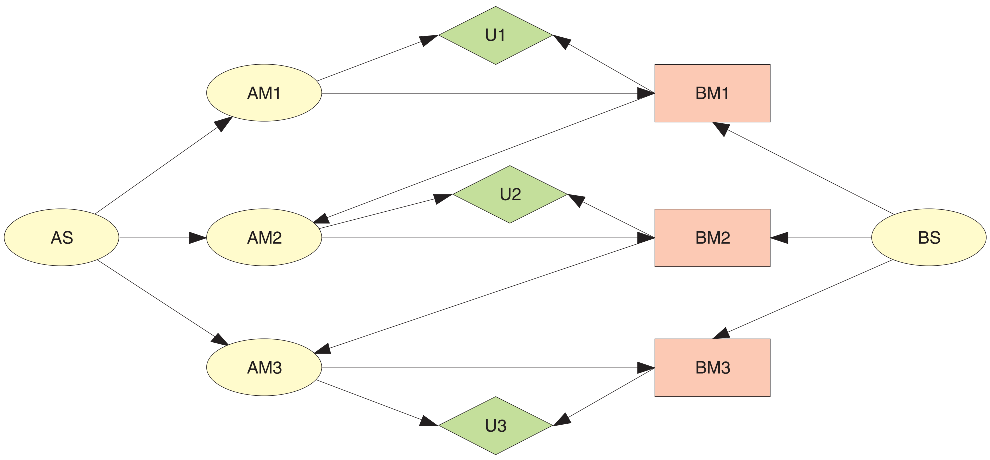
2.4 Influensdiagram for simplificeret tænkeboks

Efter at vi har analyseret spilforløbet i det simple tænkeboks, vælger vi at modellere spillet set fra Bentes synspunkt. Vi ønsker, at modellen skal foreslå de meldinger, der maksimerer Bentes forventede nytte, givet Arnes strategi samt at der er evidens på Arnes meldinger og Bentes slag. Det vil sige, at vi vil opstille et influensdiagram for spillet, hvor Bente er beslutningstageren. Bente kender ikke Arnes strategi, men hendes gæt indsættes på pilene, der fører ind i Arnes meldingsvariable. Det skal bemærkes, at et influensdiagram kan foreslå flere meldinger med samme forventede nytte.

Ved overgangen fra bayesiansk netværk til influensdiagram skal vi sikre os, at alle variable, der har pile til beslutningsvariablene, er kendte, når Bente skal melde. Modellen på Figur 2.4 bliver ændret til modellen på Figur 2.5.

Som det ses af figuren, er der ud over beslutningsvariable blevet indført nyttefunktioner mellem Arnes og Bentes meldinger. Disse nyttefunktioner skal sikre, at modellen altid vælger gyldige meldinger i BM_1, BM_2 og BM_3. Dette er gjort ved, at alle ugyldige meldinger får tildelt værdien -10, f.eks. når Arne har meldt 2, så får BM$=1$ tildelt værdien -10. Herved vælger modellen ikke disse meldinger, når der propageres, og nyttefunktionen maksimeres.

Førnævnte er dog ikke nok til, at modellen kan give et bud på, hvordan Bente skal melde. Det er nødvendigt at fortælle den, hvornår Bente vinder og taber. I tænkeboks er dette kompliceret, da det ikke kan forudses, hvornår spillet ender. Som beskrevet i reglerne for tænkeboks i afsnit 1.1 på side 9, slutter spillet, når en af spillerne vælger løft, hvorefter meldingen forinden sammenlignes med slagene, som spillerne sidder med. Det betyder i vores tilfælde, at det er nødvendigt med nyttefunktioner mellem meldinger, der følger lige efter hinanden, dvs. mellem: AM_1



Figur 2.5: Influensdiagram for modellen, hvor Bentes meldinger er beslutningsvariable.

og BM_1, BM_1 og AM_2, AM_2 og BM_2, BM_2 og AM_3 samt AM_3 og BM_3. Således bliver modellen, som vist på Figur 2.6.

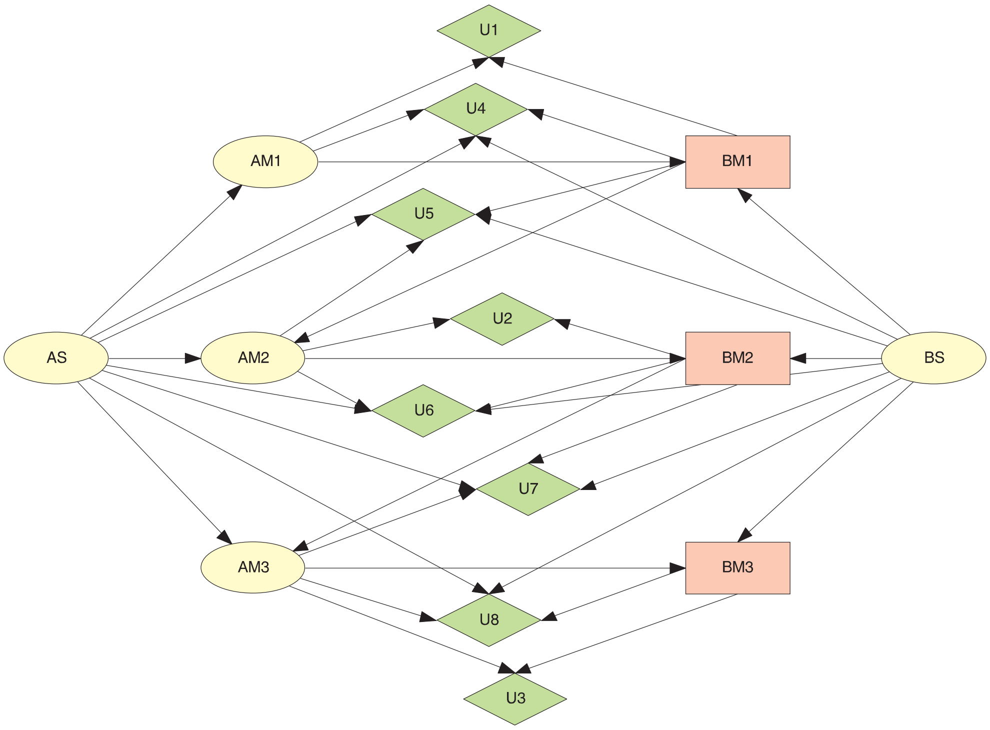
Foruden at være forbundet med de respektive meldinger, skal nyttefunktionerne desuden have input fra AS og BS, for derved at finde ud af, om meldingen er opfyldt eller ej. Hvis der løftes, og Bente vinder spillet, vælger vi at belønne det med en forventet nytte på 1, og hvis Bente taber, giver det -1. Hvis der ikke løftes, sættes den forventet nytte til 0. Nyttefunktionen mellem to meldinger sættes endvidere til 0, hvis begge meldinger er løft.

Hvis denne model bliver analyseret nærmere, kan det konkluderes, at U_1, U_2 og U_3 kan reduceres bort. Dette kan gøres ved at inkorporere disse nyttefunktioner i henholdsvis U_4, U_6 og U_8. Vi lader de nye nyttefunktioner for U_4, U_6 og U_8 være summen af de eksisterende nyttefunktioner, dvs. henholdsvis $U_1 + U_4$, $U_2 + U_6$ og $U_3 + U_8$, [Jensen, 1996, s. 143]. Herved kan modellen reduceres til Figur 2.7.

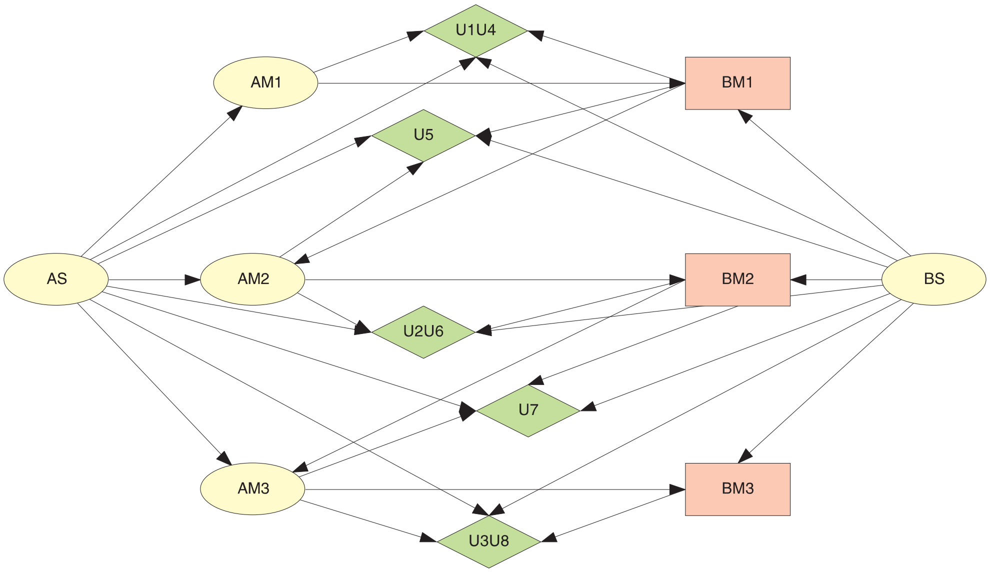
For at benytte det opstillede influensdiagram er det nødvendigt at specificere en strategi for Arne. Derfor vil vi i det næste afsnit finde en strategi i det simple tænkeboks.

2.5 Valg af strategi ud fra sandsynlighedsovervejelser

Vi vil nu finde frem til en strategi, der i influensdiagrammet kan benyttes som Arnes strategi. For at dette valg for Arnes strategi ikke skal blive fuldstændig baseret på et skøn, vælger vi først at gætte på en strategi for Bente og ud fra dette beregne sandsynlighederne for, hvilke terningslag der findes på bordet. Disse sandsynligheder anvendes til at finde en strategi for Arne.



Figur 2.6: Influensdiagram for modellen. U_1–U_3 garanterer, at Bentes meldinger er gyldige, og U_4–U_8 afgør, om Bente vinder eller ej.



Figur 2.7: Endeligt influensdiagram for modellen.

Med udgangspunkt i en simpel strategi for Bente illustrerer vi, hvorledes sandsynlighedsberegningerne skal udføres. Hvis vi skulle foretage beregninger for en bedre strategi for Bente, ville det blive for komplekst, og vi ville med fordel kunne benytte influensdiagrammet til at finde en optimal strategi for Arne.

Vi vælger at se på to af Arnes modstrategier til Bentes strategi, en deterministisk strategi og en nondeterministisk strategi. De to modstrategier finder vi vha. sandsynlighedsovervejelser for, hvilke kombinationer af terninger der er ialt, givet Bentes strategi. Grunden, til at vi har valgt at se på en nondeterministisk strategi, er, at Bente måske ellers ville kunne gennemskue Arnes strategi, hvis de spiller mange gange mod hinanden.

2.5.1 Beskrivelse af den deterministisk strategi

Den deterministiske strategi skal give et entydigt svar på, hvilken melding Arne skal give. Vi vil beregne sandsynlighederne for, hvad der er på bordet, når Bentes meldinger er kendte. Ud fra disse sandsynligheder skal Arnes meldinger bestemmes. Vi antager, at terningerne er symmetriske, hvorved der er 50% sandsynlighed for 1 og 50% sandsynlighed for 2.

Vi er interesserede i at bestemme sandsynlighederne for de forskellige kombinationer af terningeslag ud fra viden om Arnes slag og Bentes estimerede strategi. Hvis Arne f.eks. har slået 2 og Bente har meldt 1, er vi interesserede i at vide, om Bente har 1, eller om hun har 2. Vi ser på sandsynligheden for, hvilke mulige kombinationer af terningeslag der kan være:

$$P(1|BM = 1, AS = 2) = P(BS = 1|BM = 1)$$

Fra Bayes formel får vi, at

$$= \frac{P(BM = 1|BS = 1)P(BS = 1)}{P(BM = 1)}.$$

Da sandsynligheden for, at Bente melder 1, er summen af sandsynlighederne for, at hun melder 1, når hun har slået hhv. 1 og 2, har vi, at

$$= \frac{P(BM = 1|BS = 1)P(BS = 1)}{P(BM = 1|BS = 1)P(BS = 1) + P(BM = 1|BS = 2)P(BS = 2)}.$$

På grund af symmetriske terninger, er $P(BS = 1) = P(BS = 2)$, og dermed

$$= \frac{P(BM = 1|BS = 1)}{P(BM = 1|BS = 1) + P(BM = 1|BS = 2)}.$$

Ud fra lignende udregninger får vi, at

$$\begin{aligned} P(2 \cdot 2|BM = 1, AS = 2) &= P(BS = 2|BM = 1) \\ &= \frac{P(BM = 1|BS = 2)}{P(BM = 1|BS = 1) + P(BM = 1|BS = 2)}. \end{aligned}$$

Det er trivielt, at

$$\begin{aligned} P(2|BM = 1, AS = 2) &= 1 \text{ og} \\ P(2 \cdot 1|BM = 1, AS = 2) &= 0. \end{aligned}$$

Det skal bemærkes, at lignende udregninger skal udføres for de resterende kombinationer af Bentes meldinger og Arnes slag.

Da vi ikke kender Bentes strategi, kan disse sandsynligheder ikke beregnes. For at estimere hendes strategi antager vi, at hun foretager overvejelserne beskrevet i Tabel 2.2.

		AS=1		AS=2	
BS	BM	AM	Resultat	AM	Resultat
1	1	$2 \cdot 1$	B Taber	2	B Taber
	2	løft	B Taber	$2 \cdot 2$	B Vinder
	$2 \cdot 1$	løft	B Vinder	løft	B Taber
2	1	$2 \cdot 1$	B Vinder	2	B Vinder
	2	løft	B Vinder	$2 \cdot 2$	B Taber
	$2 \cdot 2$	løft	B Taber	løft	B Vinder

Tabel 2.2: Hvad Arne regner med, Bente regner med, han vil melde ud fra hendes meldinger, og resultatet deraf.

I tabellen overvejer Arne forskellige strategier, for Bente, og deres udfald i det simple tænkeboks. Når Bente har slået 1, regner hun med at tabe, hvis hun melder 1. Til gengæld regner hun med at vinde ved meldingen 1, når hun har slået 2. Men hvis Bente valgte kun at melde 1, når hun havde slået 2, ville hun derved fortælle Arne sit slag, og Arne ville da kunne vinde ved at melde 2 eller løfte, afhængig af hvad han har slået. Det er altså nødvendig for Arne at regne med, at Bente alligevel vil melde 1 i en procentdel af de tilfælde, hvor hun har slået 1. Ud fra disse overvejelser og Tabel 2.2 kan en simpel strategi for Bente estimeres, se Tabel 2.3, hvor vi for simplicitetens skyld antager, at hun altid løfter på en melding, hvor hun ikke begynder meldingsrunden.

I tabellen kan det ses, at hvis Bente skal begynde spillet, dvs. Arne har meldt 0, og hun har slået en etter, så regner Arne med, at hun vil melde 1 i 20% af tilfældene,

	$AM=0$		$AM=1,2,2\cdot 1,2\cdot 2$	
$BM \backslash BS$	1	2	1	2
1	20%	50%	0%	0%
2	40%	25%	0%	0%
2·1	40%	0%	0%	0%
2·2	0%	25%	0%	0%
løft	-	-	100%	100%

Tabel 2.3: Bentes strategi, P(BM|BS), ud fra overvejelserne i Tabel 2.2 og vores "løfte" simplificering.

2 i 40% af tilfældene og $2 \cdot 1$ i de resterende tilfælde. Vi har nu fra tabellen de sandsynligheder, der skal til, for at Arne kan bestemme sandsynlighederne for de forskellige kombinationer af terningeslag.

Hvis vi går tilbage til udregningerne på side 19, kan vi se, at for at kunne bestemme sandsynligheden for en etter er det nødvendigt at aflæse følgende sandsynligheder af Tabel 2.3:

$$
\begin{aligned}
P(BM\!=\!1|BS\!=\!1) &= 0.2 \\
P(BM\!=\!1|BS\!=\!2) &= 0.5
\end{aligned}
$$

De resulterende sandsynligheder for kombinationerne af øjne, når Arne har slået 2, og Bente har meldt 1, bliver da:

$$
\begin{aligned}
P(1|BM\!=\!1, AS\!=\!2) &= 0.29 \\
P(2|BM\!=\!1, AS\!=\!2) &= 1 \\
P(2\cdot 1|BM\!=\!1, AS\!=\!2) &= 0 \\
P(2\cdot 2|BM\!=\!1, AS\!=\!2) &= 0.71
\end{aligned}
$$

I Tabel 2.4 vises de beregnede sandsynlighederne for de forskellige slag givet Arnes slag og Bentes melding, når hun følger strategien beskrevet i Tabel 2.3. Endvidere vises meldingen, Arne skal foretage ud fra disse sandsynligheder.

Arnes melding bestemmes på følgende måde. Hvis der er 0% sandsynlighed for Bentes melding, vinder Arne, når han vælger at løfte. Et eksempel på dette er, hvis Bente har meldt $2 \cdot 1$, og Arne har slået 2, så vælger Arne at løfte. Et andet kriterie for Arnes melding er, at han altid vælger den mulige melding, der ifølge tabellen har den højeste sandsynlighed. Et eksempel på dette er, hvis Arne har slået 2, og Bente har meldt 1, så viser tabellen, at der er 71% sandsynlighed for, at der er $2 \cdot 2$, mens der er 100% sandsynlighed for, at der er 2. Derfor vælger han i dette tilfælde at melde 2.

$BM \backslash AS$	1					2				
	$P(1)$	$P(2)$	$P(2{\cdot}1)$	$P(2{\cdot}2)$	AM	P(1)	P(2)	P(2·1)	P(2·2)	AM
ingen	100%	50%	50%	0%	1	50%	100%	0%	50%	2
1	100%	71%	29%	0%	2	29%	100%	0%	71%	2
2	100%	38%	62%	0%	$2 \cdot 1$	62%	100%	0%	38%	$2 \cdot 2$
$2 \cdot 1$	100%	0%	100%	0%	løft	100%	100%	0%	0%	løft
$2 \cdot 2$	100%	100%	0%	0%	løft	0%	100%	0%	100%	løft

Tabel 2.4: Sandsynligheder for hvad, der er på bordet, samt hvad Arne skal melde i den deterministiske strategi. BM = "ingen" angiver situationen, hvor Arne starter.

Givet at Bente holder sig til den estimerede strategi i Tabel 2.3, vil disse meldinger udgøre en optimal strategi for Arne.

2.5.2 Beskrivelse af den nondeterministiske strategi

I den nondeterministiske strategi skal Arne randomisere sine valg. Det vil f.eks. sige, at han, når han har slået 1, og Bente har meldt 1, nogle gange skal melde 2 og andre gange $2 \cdot 1$.

Ud fra sandsynlighederne fundet i Tabel 2.4 og egne overvejelser har vi valgt, at Arne skal melde som vist i Tabel 2.5.

$\backslash$ AS	1					2				
BM $\backslash AM$	1	2	2·1	2·2	L	1	2	2·1	2·2	L
ingen	10%	45%	45%	0%	0%	25%	50%	0%	25%	0%
1	0%	71%	29%	0%	0%	0%	58%	0%	42%	0%
2	0%	0%	62%	0%	38%	0%	0%	0%	100%	0%
$2 \cdot 1$	0%	0%	0%	0%	100%	0%	0%	0%	0%	100%
$2 \cdot 2$	0%	0%	0%	0%	100%	0%	0%	0%	0%	100%

Tabel 2.5: Sandsynligheder for Arnes meldinger i én nondeterministisk strategi.

Tabellen viser sandsynlighederne for, hvad Arne skal melde, givet at Bente følger strategien beskrevet i Tabel 2.3. Et eksempel er, at Arne har slået 1, og Bente har meldt 1, så skal Arne melde 2 i 71% af tilfældene og melde $2 \cdot 1$ i 29% af tilfældene.

Hvis Arne begynder, melder han ud fra overvejelserne, som er beskrevet for Bente i Tabel 2.2 på side 20 og Tabel 2.4. Et eksempel er, når Arne har slået 1, så skal han melde 1 i 10% af tilfældene, 2 i 45% af tilfældene og $2 \cdot 1$ i resten af tilfældene.

Vi ønsker kun at implementere én strategi for Arne i influensdiagrammet, derfor ønsker vi at undersøge, hvor meget den nondeterministiske er dårligere end den deterministiske. Hvis forskellen er lille, kunne det eventuelt betale sig, at benytte den nondeterministiske strategi, da denne er sværere at gennemskue og mere fleksibel i tilfælde af, at Bente skulle variere sin strategi.

2.5.3 Simulering af spil

Vi tester de to fundne strategier for Arne mod den estimerede strategi for Bente. Dette gøres ved at simulere 10.000 spil mellem Bente og hhv. den deterministiske og den nondeterministiske strategi. Til simuleringen benytter vi programmet, som er beskrevet i Appendiks A. Resultatet kan ses i Tabel 2.6.

Simulering	Spil tabt af Bente
Deterministisk vs. Bente	100.0%
Nondeterministisk vs. Bente	65.31%
Bente vs. Deterministisk	62.01%
Bente vs. Nondeterministisk	58.99%

Tabel 2.6: Testresultater for simulering af spil mellem den forventede strategi for Bente og hhv. Arnes deterministiske og nondeterministiske strategi.

Ifølge reglerne for tænkeboks i afsnit 1.1 begynder taberen fra forrige spil det næste spil. Derfor antager vi, at Arne og Bente hver begynder 50% af spillene. Det viste sig, at den deterministiske strategi vandt $\frac{1}{2}(100.0\% + 62.01\%) = 81.01\%$ af spillene, mens den nondeterministiske strategi kun vandt $\frac{1}{2}(65.31\% + 58.99\%) = 62.15\%$ af spillene. Forskellen mellem den deterministiske og den nondeterministiske strategi er dermed, at den deterministiske vinder ca. 19% flere spil. Ud fra denne viden vælger vi den deterministiske strategi til Arne.

2.6 Anvendelse af strategi i influensdiagram

Vi vender nu tilbage til influensdiagrammet på Figur 2.7 på side 18, hvor vi ud fra en strategi for Arne ville finde Bentes bedste modstrategi.

Vi vælger, ud fra resultaterne i afsnit 2.5.3, at indsætte den deterministiske strategi for Arne i influensdiagrammet og finder på denne måde en strategi for Bente vha. Hugin. Da vi ønsker at modellere spillet, hvor Arne og Bente skiftes til at starte, giver vi Arne 50% sandsynlighed for at melde 0 i første melding.

En analyse af spillet vil kunne give, at visse meldinger i bestemte situationer ikke er særlig givtige. I det simple tænkeboks blev der under simplificering af det bayesianske netværk, på side 15, konkluderet, at Bente vil tabe i de tilfælde, hvor hun melder $2 \cdot 2$ og har slået 1, da Arne kun kan løfte på denne melding. Endvidere blev det konkluderet, at i de tilfælde, hvor Bente har slået 2 og melder $2 \cdot 1$, vil hun tabe spillet, da Arne på denne melding løfter, ligegyldig hvad han har slået. Disse overvejelser kan implementeres i influensdiagrammet ved at tildele disse meldinger nytten -5 i de respektive nyttefunktioner. På denne måde bliver de modstrategier, vi finder, mere robuste, hvis Arne skulle skifte strategi.

Det nye influensdiagram giver 18 deterministiske strategier for Bente, som vist i Tabel 2.7.

$AM \backslash BS$	1	2
0	$\begin{cases} 1 \\ 2 \\ 2 \cdot 1 \end{cases}$	$\begin{cases} 1 \\ 2 \\ 2 \cdot 2 \end{cases}$
1	$2 \cdot 1$	2
2	$\begin{cases} 2 \cdot 1 \\ \text{løft} \end{cases}$	$2 \cdot 2$
$2 \cdot 1$	løft	løft
$2 \cdot 2$	løft	løft

Tabel 2.7: Bentes bedste modstrategier mod Arnes nondeterministiske strategi ifølge influensdiagram på Figur 2.7 på side 18, hvor reduceringerne er blevet indført.

Ved simulering af spil mellem Arnes deterministiske strategi, fra Tabel 2.4 på side 22, mod en nondeterministisk strategi for Bente, hvor hun randomiserer ligeligt mellem valgene i Tabel 2.7, viser det sig ved simulering af 10.000 spil, at Bente vinder 75% af spillene, når Arne begynder, og 51% af spillene, når hun selv begynder.

Af ovenstående ses, at den deterministiske strategi ikke klarede sig særligt godt mod Bentes modstrategi. Dette skyldes, at Arnes antagelse om Bentes strategi ikke var særlig realistisk. Antagelsen om, at Bente løfter på alle Arnes meldinger gør, at Arne i mange tilfælde foretager en melding, som Bente i sin modstrategi kan forvandle til en sejr, ved at byde op. På baggrund heraf kan det måske bedre betale sig for Arne at benytte den nondeterministiske strategi mod Bente, da den nondeterministiske strategi trods alt er mere fleksibel, hvis Bente ikke løfter på alle Arnes meldinger.

Generelt tager hverken den fundne deterministiske eller nondeterministiske strategi højde for det videre spilforløb, da de kun er konstrueret ud fra, at Bente løfter på alle meldinger. Eksemplet illustrerer, hvor vigtigt det er at have en god antagelse om modstanderens strategi.

2.7 Opsummering

I dette kapitel så vi først på, hvilke komponenter bayesianske netværk består af, samt hvilken indflydelse disse komponenter har på resten af netværket. Da vi ønskede, at modellen givet en strategi skulle kunne finde en optimal modstrategi, blev influensdiagrammer introduceret,

Efter disse afsnit begyndte den egentlige modellering af tænkeboks. Vi simplificerede tænkeboks, således at der kun er to deltagere, Arne og Bente, der hver har en terning med to sider. Vi analyserede spilforløbet og konstruerede et bayesiansk netværk, der beskriver dette. Netværket blev analyseret, og vi fandt ud af, at nogle af pilene var overflødige.

Det bayesianske netværk transformerede vi til et influensdiagram, for på denne måde at kunne finde en bedste modstrategi for Bente, givet en strategi for Arne. Ud fra sandsynlighedsovervejelser og egne erfaringer fandt vi i afsnit 2.5 frem til to strategier for Arne, en deterministisk og en nondeterministisk strategi. Ved at simulere spil mellem en formodet strategi for Bente og de fundne strategier for Arne, fandt vi frem til, at den deterministiske var bedst, givet at Bente benyttede den formodede strategi.

Ved hjælp af det opstillede influensdiagram fandt vi i afsnit 2.6 frem til Bentes bedste modstrategi, givet at Arne benyttede den fundne deterministiske strategi. Vi simulerede 10.000 spil mellem Bentes bedste modstrategi og Arnes deterministiske strategi vha. et C-program. Det viste sig, at Bentes modstrategi vandt 51% af de spil, hvor hun startede, og 75% af spillene, hvor Arne startede.

Det ringe udbytte for Arne skyldes hans antagelse om Bentes strategi. Spørgsmålet er, om vi kunne have fundet en bedre strategi, der ikke afhænger af en antagelse om Bentes strategi. En sådan strategi vil vi senere forsøge at finde vha. spilteori.

I dette kapitel benyttede vi Hugin til at propagere evidens i bayesianske netværk og influensdiagrammer. Vi undersøger derfor i næste kapitel den matematiske baggrund for propagering.

Kapitel 3

Propagering i bayesianske netværk vha. junction træer

I dette kapitel introduceres, hvordan Hugin opdaterer sandsynlighederne i et bayesiansk netværk, dvs. hvordan initialsandsynligheder og evidens propageres rundt i netværket. Motivationen for dette kapitel er, at vi ønsker at kende de beregningstrin som Hugin udfører, når den propagerer i det bayesianske netværk. På den måde undgår vi, at Hugin virker som en magisk boks, men kommer til at virke som et værktøj, hvis funktion, vi kender. Samtidigt ønsker vi også at beskæftige os med teorien bag disse beregningstrin på et stringent plan, hvorved vi beviser at de sandsynligheder, som Hugin finder, er korrekte. Kapitlet er baseret på viden fra [Jensen, 1996], [Schafer, 1996] og [Jensen, 1997].

Propageringen foretages ikke direkte i det bayesianske netværk, i stedet bruges et junction træ for det bayesianske netværk. Derfor indfører vi først nogle indledende definitioner herunder junction træer og fortæller, hvordan et junction træ konstrueres ud fra et bayesiansk netværk. Vi vil også bevise, at vi vha. propagering i disse junction træer kan nå frem til sandsynlighederne for variablene i det bayesianske netværk. I [Schafer, 1996] beskrives op til flere arkitekturer, der fastlægger detaljerne for propageringsoperationerne. Vi vil i det følgende kun beskæftige os med Hugin-arkitektur, der i [Schafer, 1996] kaldes Aalborg-arkitekturen.

Den simple tænkeboksmodel, introduceret i Kapitel 2, benyttes i en reduceret udgave som eksempel i dette kapitel.

3.1 Opdatering af sandsynligheder i bayesianske netværk

I dette afsnit fortælles, hvordan et bayesiansk netværks repræsentation af den simultane sandsynlighed, $P(U)$, benyttes til at udtrække sandsynlighedsfordelingen for en enkelt knudes tilstande. Hertil skal vi bruge den simultane sandsynlighed.

3.1.1 Den simultane sandsynlighed

For hver variabel i det bayesianske netværk er vi interesseret i at vide, med hvilken sandsynligheder variablen kan være i dens tilstande. Det vil sige, at for hver variabel V_i i et univers $U = \{V_1, \ldots, V_n\}$ skal vi beregne $P(V_i)$. Hvis vi kender den simultane sandsynlighed, $P(U)$, kan vi vha. marginalisering beregne sandsynligheden for V_i:

$$P(V_i) = \sum_{U \setminus \{V_i\}} P(U). \tag{3.1}$$

Dette svarer til at projicere $P(U)$ fra U ned på V_i. Vi vil fremover anvende notationen i udtryk (3.1).

Repræsentation af $P(U)$ vha. et bayesianske netværk indeholder information om $P(U)$ uden direkte at indeholde tabellen for $P(U)$. Ud fra det bayesianske netværk kan den simultane sandsynlighed beregnes vha. kædereglen:

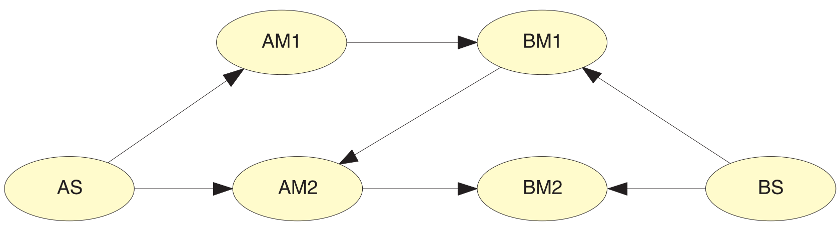
Sætning 3.1 (Kædereglen)
Lad BN være et bayesiansk netværk over universet $U = \{V_1, \ldots, V_n\}$, hvor V_i er en variabel i universet. Den simultane sandsynlighed, $P(U)$, er da produktet af alle de betingede sandsynligheder angivet i BN:

$$P(U) = \prod_{i=1}^{n} P(V_i|pa(V_i)),$$

hvor $pa(V_i)$ er mængden af forældre til V_i.

Bevis for kædereglen kan findes i [Jensen, 1996, s. 20]. Følgende eksempel viser, hvordan kædereglen kan anvendes på et bayesiansk netværk.

Eksempel 3.2
Betragt det bayesianske netværk af en reduktion af den simplificerede tænkeboksmodel fra Kapitel 2, se Figur 3.1. Denne simplificerede tænkeboksmodel indeholder kun første og anden meldingsrunde.



Figur 3.1: Bayesiansk netværk for den reducerede udgave af den simplificerede tænkeboksmodel, hvor kun første og anden meldingsrunde betragtes.

Hermed kan vi vha. kædereglen beregne den simultane sandsynlighed, $P(U)$, hvor $U = \{BM_2, AM_2, BM_1, AM_1, BS, AS\}$:

$$\begin{aligned} P(U) &= P(BM_2|BS, AM_2)P(AM_2|AS, BM_1) \cdot \\ &\quad P(BM_1|BS, AM_1)P(AM_1|AS)P(BS)P(AS). \end{aligned}$$

Vi kan nu vha. marginalisering beregne hver knudes sandsynlighedsfordeling. Betragt f.eks. Bentes første melding i spillet, BM_1:

$$\begin{aligned} P(BM_1) &= \sum_{U\backslash\{BM_1\}} P(U) \\ &= \sum_{U\backslash\{BM_1\}} [\, P(BM_2|BS, AM_2)P(AM_2|AS, BM_1)\cdot \\ &\quad P(BM_1|BS, AM_1)P(AM_1|AS)P(BS)P(AS)\,]\,. \end{aligned} \tag{3.2}$$

$\Diamond$

Kompleksitetsovervejelser

Anvendelsen af den distributive lov gør, at udtryk (3.2) i Eksempel 3.2 kan beregnes på mange forskellige måder ved at bytte om på den rækkefølge, hvormed der skal summeres og multipliceres. Hver af måderne kan have forskellige datakompleksiteter, der har stor indflydelse på beregningstiden af alle $P(V_i)$. Tabel 3.1 viser antallet af tilstande i hver variabel i U.

Variabel	AS	BS	AM_1	BM_1	AM_2	BM_2
Antal tilstande	2	2	5	5	4	3

Tabel 3.1: Antal tilstande i hver variabel i universet, U.

Vi har, at

$$\begin{aligned} P(BM_1) &= \sum_{\{BS,AM_2,BM_2\}} P(BM_2|BS, AM_2) \sum_{AS} P(AM_2|AS, BM_1) \cdot \\ &\quad \sum_{AM_1} P(BM_1|BS, AM_1)P(AM_1|AS)P(BS)P(AS) \end{aligned}$$

har $100 + 80 + 120 = 300$ produkter og $80 + 40 + 80 + 30 + 5 = 235$ summer. Denne kompleksitetsberegning er foretaget på følgende måde. Der startes bagfra, hvilket vil sige, at vi først ser på dimensionen af tabellen, der angiver det sidste produkt. Denne dimension er $|AS| \cdot |BS| \cdot |AM_1| \cdot |BM_1| = 2 \cdot 2 \cdot 5 \cdot 5 = 100$. Det vil sige, at der bliver 100 produkter, da der skal udregnes et produkt for hver celle i tabellen.

Antallet af summationer ved marginalisering af AM_1 udregnes på følgende måde. Dimensionen af det tilbageværende efter summering af AM_1 vil være $\frac{100}{|AM_1|} = 20$.

For at komme ned på denne størrelse, er det nødvendigt at summere AM_1's fem tilstande ud for hver af de 20 celler. Det vil sige, at der for hver celle er behov for fire summationer, og dermed i alt $20 \cdot 4 = 80$ summer.

For at beregne dimensionen af produktet

$$\overbrace{P(AM_2|AS, BM_1) \cdot \overbrace{\sum_{AM_1} P(BM_1|BS, AM_1)P(AM_1|AS)P(BS)P(AS)}^{summen}}^{produktet},$$

skal det undersøges, hvilke variable i $P(AM_2|AS, BM_1)$ der ikke optræder i summen. Det er kun AM_2, der ikke er repræsenteret i summen. Da $|AM_2| = 4$ og $|AS| \cdot |BS| \cdot |BM_1| = 2 \cdot 2 \cdot 5 = 20$, bliver dimensionen af tabellen, der repræsenterer produktet, $20 \cdot 4 = 80$.

Næste skridt er at finde antallet af summationer ved $\sum_{AS}$ *produktet* og så fremdeles.

Derimod har vi, at

$$\begin{aligned} P(BM_1) \quad = \quad & \sum_{\{AS,BS,AM_1\}} P(BM_1|BS, AM_1)P(AM_1|AS)P(BS)P(AS) \ \cdot \\ & \sum_{AM_2} P(AM_2|AS, BM_1) \sum_{BM_2} P(BM_2|BS, AM_2) \end{aligned} \tag{3.3}$$

kun har $24 + 80 + 100 = 204$ produkter og $16 + 60 + 80 + 10 + 5 = 171$ summer. $\sum_{U\setminus\{BM_1\}} P(U)$ opstillet som i (3.2) har 1200 produkter og $800+300+80+10+5 = 1195$ summer.

Problemet med beregning af sandsynligheder for variable ud fra den simultane sandsynlighed er dermed, at antallet af additioner og multiplikationer vokser eksponentielt med antallet af variable i universet. Vi har endnu ikke vist en måde at udregne udtrykket i (3.2), så kompleksiteten er så lille som mulig, som den f.eks. er i (3.3). I afsnit 3.2 kommer vi nærmere ind på, hvordan dette løses med kliker i triangulerede grafer og junction træer. Men først en forklaring på, hvad der sker, når der tilføres evidens til netværket.

3.1.2 Evidens

I beregningen af $P(BM_1)$ i Eksempel 3.2 på side 28 beskæftigede vi os kun med initialstillingen, dvs. før netværket får tilført information om sikre tilstande hos nogle af variablene. En sådan information kaldes en finding.

Definition 3.3 (Finding)

Lad V_i være en variabel med n tilstande. En **finding** på V_i er en n-vektor med 0'er og 1'er, der angiver udefrakommende information om, at nogle af V_i's tilstande er umulige. Betragt det j'te element i vektoren. Et 1 indikerer, at V_i kan være i tilstand j, mens 0 fortæller, at tilstand j er umulig.

En mængde af findings $\{\underline{f}_1, \ldots, \underline{f}_k\}$ for variablene i et bayesiansk netværk siges at tilføre netværket **evidensen** $\underline{e} = \{\underline{f}_1, \ldots, \underline{f}_k\}$.

Lad V_i være en variabel med n tilstande og $P(V_i) = (x_1, \ldots, x_n)$. Antag, at V_i får evidensen $\underline{e} = \{\underline{f}_1\}$, hvor $\underline{f}_1$ fortæller, at V_i kun kan befinde sig i tilstand j og x. Vi har derfor, at $P(V_i, e) = P(V_i) \cdot \underline{e} = (0, \ldots, 0, x_j, 0, \ldots, 0, x_k, 0, \ldots, 0)$. Vi har af fundamentalreglen, at

$$P(V_i|e) = \frac{P(V_i, e)}{P(e)} = \frac{P(V_i, e)}{\sum_{V_i} P(V_i, e)}.$$

Generelt gælder følgende sætning, som vi vil anføre uden bevis.

Sætning 3.4

Lad U være et univers af variable og $e = \{\underline{f}_1, \ldots, \underline{f}_k\}$ være en række findings på U. Så er

$$P(U, e) = P(U) \cdot \underline{f}_1 \cdot \cdots \cdot \underline{f}_k,$$

og

$$P(U|e) = \frac{P(U, e)}{P(e)},$$

hvor $P(e) = \sum_U P(U, e)$.

Sætning 3.4 siger, at hvis vi kender den simultane sandsynlighed, $P(U)$, så kan evidens tilføres og en opdatering til den nye simultane sandsynlighed, $P(U, e)$, kan udføres. Men tabellen for $P(U)$ bliver som sagt hurtigt alt for stor til at kunne være direkte repræsentabel. Egentligt behøver vi ikke $P(U)$ direkte for at beregne sandsynligheder af enkelte variable i netværket. Det bayesianske netværk er allerede en smartere repræsentation af $P(U)$, der har mindre datakompleksitet. Men af beregningsmæssige årsager indføres en tilsvarende smart repræsentation i form af junction træer.

3.2 Konstruktion af junction træer

Først introduceres triangulerede grafer, hvis kliker er byggesten i et junction træ. Herefter ses på, hvilke egenskaber der knytter sig til junction træer, herunder hvorledes kompleksiteten af udtryk (3.2) i Eksempel 3.2 på side 28 bliver så lille som muligt.

3.2.1 Triangulerede grafer

En trianguleret graf defineres på følgende måde, hvor $V(G)$ og $E(G)$, er hhv. punkt- og kantmængden for en graf G.

Definition 3.5 (Trianguleret graf)

En graf G kaldes **trianguleret**, hvis der for alle kredse med længde større end 3 eksisterer en kant $xy \in E(G)$, således at xy ikke er med i kredsen samtidigt med, at knuderne x, y er det. En sådan kant kaldes en **korde**.

En graf G' kaldes en **triangulering** af G, hvis G' er trianguleret, $V(G) = V(G')$ og $E(G) \subseteq E(G')$.

Bemærk, at hvis man f.eks. har en graf med en kreds af længde 6, så bliver grafen ikke trianguleret ved, at der kommer en korde i kredsen, da korden giver anledning til en ny kreds med eksempelvis længde 5. Da denne kreds ikke har en korde, er grafen stadig ikke trianguleret.

Triangulering af en moral graf svarer til grafen udvidet med de kanter, der tilføjes ved eliminering af grafens knuder en efter en. Moral graf og eliminering defineres hermed.

Definition 3.6 (Moral graf)

Lad BN være et bayesiansk netværk. Lad M være den underliggende ikke orienterede graf af BN. I M tilføjes kanter mellem to knuder, når de i BN har et fælles barn. Grafen M kaldes **moral grafen** for BN.

Definition 3.7 (Eliminering)

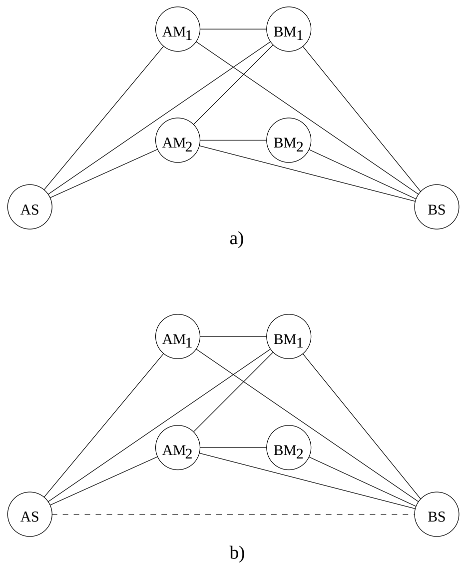
En knude x **elimineres** fra en graf ved at tilføje kanter, kaldet **fill-ins**, mellem hvert par af naboer til x, og derefter fjernes knuden x og alle kanterne til x.

Figur 3.2 indeholder moral grafen og en triangulering deraf for det bayesianske netværk på Figur 3.1 på side 28. Trianguleringen er opnået ved at udvide moral grafen med de fill-ins, der opstår ved en eliminering af knuderne i rækkefølgen BM_2, AM_2, BM_1, AM_1, BS, AS. Alternativt kunne trianguleringen opnås ved at følge Definition 3.5 og derfor i moral grafen undersøge alle kredse med længde større end 3. Dette giver, at korden mellem AS og BS skal indsættes pga. kredsen AS, AM_1, BS og AM_2, før grafen er trianguleret.

Definition 3.7 og Definition 3.5 giver os, at hvis en knude, x, kan elimineres uden at tilføje fill-ins, så kan x ikke have været med i en kordeløs kreds med mere end tre kanter.

Sætning 3.8

En graf er trianguleret, hvis og kun hvis alle dens knuder kan elimineres uden at skulle tilføje fill-ins.



Figur 3.2: Moral grafen af BN er grafen i a) og en triangulering heraf er grafen i b). Indsatte korder i b) er markeret med stiplede linier.

Beviset for Sætning 3.8 kan findes i [Jensen, 1996, s. 159-161].

Desværre viser det sig, at algoritmer til triangulering af grafer er NP-komplette.

Sætning 3.9

Lad G være en graf, hvor hver knude har en værdi tilknyttet. Lad $N \in \mathbb{N}$. Størrelsen af en klike i G defineres til at være produktet af værdierne, der knytter sig til knuderne i kliken. Problemet, *"Findes der en triangulering af G med en total klikestørrelse mindre end N?"*, er NP-komplet.[Jensen, 1997, s. 10]

I tilfældet med bayesianske netværk, vil værdien, der knyttes til knuderne, være det antal tilstande, som hver variabel har. Bemærk, at for nogle tilfælde af N vil spørgsmålet dog være trivielt. For eksempel, hvis N er større end produktet af samtlige tilstande i netværket. Den maksimale klikestørrelse svarer således til dimensionen af den største tabel, vi skal bruge under udregningen af sandsynlighederne. Ved at minimere den maksimale klikestørrelse opnås, at antallet af summationer og multiplikationer ved beregningen minimeres.

På trods af, at den optimale løsning er NP-komplet, så findes der en god heuristik, kaldet **one-step-look-ahead**, der giver gode resultater [Jensen, 1997]. Heuristikken siger, at vi hele tiden skal eliminere de knuder, der ikke giver anledning til fill-ins. Giver alle knuder anledning til fill-ins, vælges næste knude, således at klikestørrelsen af den klike, som opstår ved at eliminere knuden, bliver mindst muligt.

Vi er nu klar til at definere junction træer og fortælle, hvordan et junction træ kan konstrueres ud fra et bayesiansk netværk.

3.2.2 Junction træer

Efter at en graf er blevet trianguleret tales om dens tilsvarende junction graf.

Definition 3.10 (Junction graf)
Lad G være en ikke orienteret graf og $C_1, \ldots, C_k$ være klikerne i G. Ved **junction grafen**, G^J, for G forstås grafen med $V(G^J) = \{C_1, \ldots, C_k\}$ og $E(G^J) = \{C_iC_j | C_i \cap C_j \neq \emptyset\}$.

Junction træet defineres nu ud fra junction grafen. Bemærk, at de tilknyttede separatorer er specielle for Hugin-arkitekturen.

Definition 3.11 (Junction træ)
Lad G^J være junction grafen for en ikke orienteret graf G. Et **junction træ**, J, for G er da et udspændende træ i G^J, der opfylder, at der for ethvert par af knuder $C_i, C_j \in V(G^J)$ gælder, at alle knuder på vejen mellem C_i og C_j mindst indeholder variablene i $C_i \cap C_j$.

Til hver kant, $xy \in E(J)$, er der tilknyttet en separator, der angiver fællesmængden, $x \cap y = S_{xy}$ mellem kantens knuder. Mængden af separatorer er da $S(J) = \{S_{xy} = x \cap y | xy \in E(J)\}$.

Bemærk, at vi i ovenstående definition både bruger C_i, C_j og x, y for knuder i et junction træ. Vi har valgt fremover primært at bruge x og y eller lignende for angivelse af knuder i et junction træ.

Af konstruktionen af de tilknyttede separatorer i Definition 3.11 ses, at givet to knuder $v, w \in V(J)$, gælder der for enhver kant $xy \in E(J)$ på vejen mellem u og v, at $v \cap w \subseteq S_{xy}$.

Det er ikke altid muligt ud fra en junction graf for en graf G at skabe et junction træ. Følgende sætning skaber relation mellem triangulerede grafer og junction træer.

Sætning 3.12
En ikke orienteret graf, G, har et tilsvarende junction træ, J, hvis og kun hvis G er trianguleret.

Beviset for Sætning 3.12 kan findes i [Jensen, 1996, s.161-162].

3.2.3 Fra bayesiansk netværk til junction træ

Vi vil nu beskrive, hvordan et junction træ kan konstrueres ud fra et bayesiansk netværk, BN. Følgende algoritme benyttes:

Algoritme 3.13 (Konvertering til junction træ)
Lad BN være et bayesiansk netværk over universet U.

1. Foretag trianguleringen af moralgrafen for BN ved at benytte heuristikken one-step-look-ahead.
2. Konstruer junction grafen BN^J for ovenstående triangulering.
3. Hver kant xy i BN^J tildeles vægten $|x \cap y|$. Det tilsvarende junction træ J for BN konstrueres ud fra BN^J ved at køre en algoritme til at finde et maksimalt udspændende træ i BN^J, jf. [Chartrand og Oellermann, 1993] og [Jensen, 1996].
4. Til hver separator knyttes tabellen $\underline{1}$, der er den repæsentative tabel for S_{xy}, hvor alle elementer er 1.
5. For hver $V_i \in U$ tilknyt tabellen for den betingede sandsynlighed $P(V_i|pa(V_i))$ til netop én knude i J, der indeholder $V_i \cup pa(V_i)$.
6. For hver knude, x, i J multipliceres de i trin 5 tilknyttede tabeller for de n_x betingede sandsynligeder. Resultatet kaldes den tilknyttede tabel, og bliver

$$t_x = \prod_{j=1}^{n_x} P(V_{(j)}|pa(V_{(j)})),$$

hvor $V_{(j)}$ er en anden indeksering af knuderne, da kun en delmængde af variablene i U har deres tabel tilknyttet x. Hvis $n_x = 0$, så defineres $t_x = \underline{1}$.

Følgende illustrerer, hvordan et junction træ kan konstrueres ud fra et bayesiansk netværk. Vi har her valgt at tage udgangspunkt i moral grafenen på Figur 3.2 på side 33.

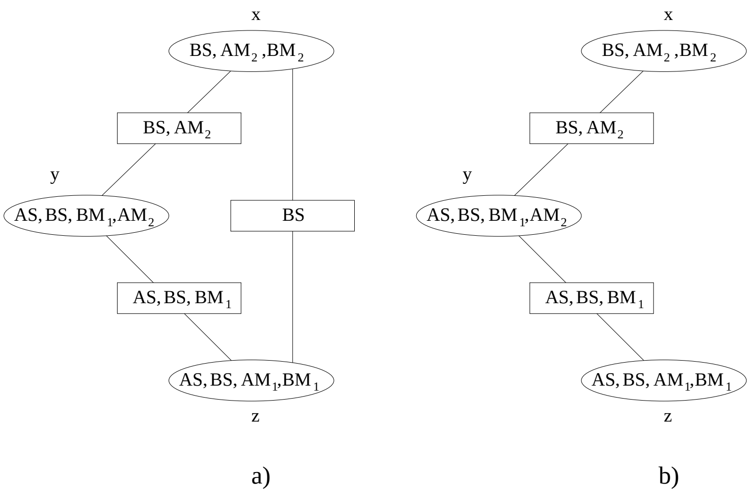
Eksempel 3.14

Lad BN være det bayesianske netværk for denne simplificerede tænkeboksmodel, se Figur 3.1 på side 28. Algoritme 3.13 bruges til at konstruere junction træet, J, ud fra BN. Moral grafen af BN og en triangulering heraf er afbilledet på Figur 3.2 på side 33. Junction grafen og et junction træ for trianguleringen af BN ses på Figur 3.3. Junction grafen, G^J, er dannet ud fra G, hvor G er grafen b) på Figur 3.2 på side 33, ved at lade klikerne i G være knuderne i G^J. Heraf fås, at $V(G^J) = \{x, y, z\}$, hvor $x = \{BS, AM_2, BM_2\}$, $y = \{AS, BS, BM_1, AM_2\}$ og $z = \{AS, BS, AM_1, BM_1\}$. Kanterne i G^J bliver som i Definition 3.10 bestemt ved at forbinde alle knuder i G^J, der har variable tilfælles. Da $|x \cap y| = 2, |y \cap z| = 3$ og $|z \cap x| = 1$ fås af Trin 3 i Algoritme 3.13, at junction træet for G^J med de tilknyttede separators bliver som b) i Figur 3.3.

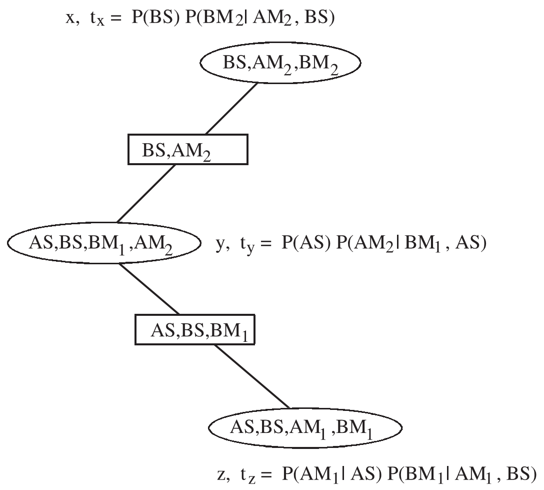
På Figur 3.4 ses junction træet fra Figur 3.3 med de tilknyttede tabeller.

Dermed er junction træet, J, konstrueret ud fra et bayesiansk netværk, BN.

$\Diamond$



Figur 3.3: a) En junction graf og b) et junction træ for BN fra Eksempel 3.2 på side 33.



Figur 3.4: Junction træ med tabeller knyttet til knuderne.

3.2.4 Konsistens

Vi forestiller os, at de enkelte knuder i en graf G har tilknyttet sandsynligheder. En knude $g \in V(G)$ kan optræde i op til flere kliker, og derfor kan g være indeholdt i op til flere knuder i J. Det er nu vigtigt, at uanset hvilken knude, der i J konsulteres, så fås samme information om sandsynlighedsfordelingen af g's tilstande. Derfor indføres konsistensbegrebet.

Definition 3.15 (Konsistens)
Lad J være et junction træ. To knuder $x, y \in V(J)$, hvor $xy \in E(J)$, siges at være **konsistente**, hvis de indeholder samme information om deres fællesmængde, S_{xy}, dvs.

$$\sum_{x \setminus S_{xy}} t_x \quad = \quad t_{S_{xy}} = \sum_{y \setminus S_{xy}} t_y.$$

Hvis det for alle kanter, xy, i J gælder, at x og y er konsistente:

$$\forall xy \in E(J) : \sum_{x \setminus S_{xy}} t_x = \sum_{y \setminus S_{xy}} t_y,$$

så siges J at være et **konsistent junction træ**. Og hvis det for alle knuder i J, $x, y \in V(J)$, gælder, at

$$\sum_{x \setminus I} t_x = \sum_{y \setminus I} t_y, \quad \text{hvor } I = x \cap y,$$

så siges J at være **globalt konsistent**.

En speciel egenskab ved junction træer er, at et konsistent junction træ også er globalt konsistent.

Sætning 3.16
Et konsistent junction træ, J, er globalt konsistent.

Bevis
Givet et junction træ, J, og to forskellige knuder x og y tilhørende J. Vi skal vise, at

$$\sum_{x \setminus I} t_x = \sum_{y \setminus I} t_y, \quad \text{hvor } I = x \cap y. \tag{3.4}$$

Da x og y er knuder i et junction træ, gælder der, at der eksisterer en entydig vej fra x til y:

$$W : x = w_0, w_1, \cdots, w_n = y.$$

Eftersom junction træet er konsistent, gælder der for $0 \leq i \leq n-1$, at

$$\sum_{w_i \backslash S_{w_i w_{i+1}}} t_{w_i} = \sum_{w_{i+1} \backslash S_{w_i w_{i+1}}} t_{w_{i+1}}.$$

Vi får dermed, at

$$\sum_{w_0 \backslash S_{w_0 w_1}} t_{w_0} = \sum_{w_1 \backslash S_{w_0 w_1}} t_{w_1}, \quad \ldots, \quad \sum_{w_{n-1} \backslash S_{w_{n-1} w_n}} t_{w_{n-1}} = \sum_{w_n \backslash S_{w_{n-1} w_n}} t_{w_n}.$$

Da J er et junction træ, gælder der for I fra udtryk (3.4), at $I \subseteq w_i$, for $0 \leq i \leq n$, og derfor specielt, at $I \subseteq S_{w_i w_i + 1}$, for $0 \leq i \leq n-1$.

Da $S_{w_i w_{i+1}}$ ikke nødvendigvis er lig med $S_{w_{i+1} w_{i+2}}$ for $0 \leq i \leq n-2$, marginaliseres ned på I. Hermed fås, at

$$\begin{aligned}
\sum_{(w_0 \backslash S_{w_0 w_1}) \cup (S_{w_0 w_1} \backslash I)} t_{w_0} &= \sum_{(w_1 \backslash S_{w_0 w_1}) \cup (S_{w_0 w_1} \backslash I)} t_{w_1} \\
&\vdots \\
\sum_{(w_{n-1} \backslash S_{w_{n-1} w_n}) \cup (S_{w_{n-1} w_n} \backslash I)} t_{w_{n-1}} &= \sum_{(w_n \backslash S_{w_{n-1} w_n}) \cup (S_{w_{n-1} w_n} \backslash I)} t_{w_n}.
\end{aligned}$$

På nuværende tidspunkt er alle tabellerne marginaliseret ned på I. Da det gælder for $0 \leq i \leq n-1$, at

$$(w_i \backslash S_{w_i w_{i+1}}) \cup (S_{w_i w_{i+1}} \backslash I) = w_i \backslash I,$$

fås

$$\sum_{w_0 \backslash I} t_{w_0} = \sum_{w_1 \backslash I} t_{w_1} = \cdots = \sum_{w_n \backslash I} t_{w_n}.$$

Da $x = w_0$ og $y = w_n$ haves, at $\sum_{x \backslash I} t_x = \sum_{y \backslash I} t_y$.

□

3.3 Propagering i junction træer

Det foregående afsnit beskæftigede sig med konverteringen af bayesianske netværk til junction træer. I dette afsnit vil vi nøjere beskrive, hvorledes junction træer benyttes til opdateringer af sandsynligheder i det bayesianske netværk. Givet et bayesiansk netværk, er opgaven at beregne den marginale fordeling, $P(V_i)$, for en variabel i netværket uden at skulle beregne hele $P(U)$. Er der indtruffet evidens, e, for netværket ønskes i stedet for $P(V_i, e)$.

Et junction træ danner grundlag for rækkefølgen, hvorpå der skal marginaliseres og multipliceres for at opnå $P(V_i)$. Vi vil nu se på Hugin-arkitekturen [Schafer, 1996], der definerer operationer til beregning af $P(V_i)$ ud fra junction træet.

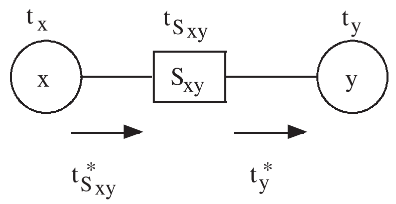
3.3.1 Hugin-arkitekturen

Grundlaget for Hugin-arkitekturen dannes af Definition 3.11 på side 34, hvor der til junctiontræet knyttes tilhørende separatorer. Det er i junction træer af denne type, at arkitekturen definerer operationer, som svarer til en del af den summering og multiplikation, der foretages i beregningen af $P(V_i)$. Det centrale i Hugin-arkitekturen er måden, hvorpå der udveksles information mellem knuderne. For junction træer under Hugin-arkitekturen defineres derfor operationen absorbering.

Definition 3.17 (Absorbering)
Lad J være et junction træ og lad $x, y \in V(J) : xy \in E(J)$. Operationen bestående af trinene

- Beregn $t^*_{S_{xy}} = \sum_{x \setminus S_{xy}} t_x$
- Tildel t_y tabellen $t^*_y = t_y \cdot \frac{t_{S^*_{xy}}}{t_{S_{xy}}}$
- Tildel $t_{S_{xy}}$ tabellen $t^*_{S_{xy}}$

kaldes en **absorbering**, hvor y absorberer fra x, se Figur 3.5.



Figur 3.5: Knuden y absorberer fra x.

En absorbering fra x til y skal fortolkes som, at x oplyser y om den information, den har om variablerne i $x \cap y$. På denne måde får y mulighed for at opdatere sin information om fællesmængden. Målet er, at x og y skal indeholde samme information om fællesmængden, dvs. at x og y skal være konsistente.

Bemærk, at divisionen med $t_{S_{xy}}$ under absorbering kun kan gennemføres uden problemer, hvis der ikke eksisterer nuller i tabellen. Da man dog gerne vil tillade sandsynligheder på nul i tabellen, udvides definitionen for division af tabeller til følgende. Divisionen af t_y med $t_{S_{xy}}$ er i orden, hvis t_y har nuller svarende til de steder, hvor $t_{S_{xy}}$ har nuller, disse steder defineres $\frac{0}{0} = 0$. En absorbering er kun mulig, hvis dette krav er opfyldt. En kant i junction træet kaldes **støttende**, hvis en absorbtion er mulig i begge retninger. Hele junction træet kaldes støttende, hvis alle dens kanter er støttende.

Der knytter sig en del egenskaber til støttende kanter, hvoraf den vigtigste er, at en støttende kant forbliver støttende under absorbering. Disse vil ikke blive diskuteret nærmere, blot henvises til [Schafer, 1996, s. 58-60].

I sammenhæng med udregning af $P(V_i)$ som et produkt af marginaliseringer, f.eks. i udtryk (3.2) på side 28, svarer en absorbering i situationen, hvor $t_{S_{xy}} = \underline{1}$, blot til anvendelsen af den distributive lov,

$$\sum_{x \setminus S_{xy}} t_x t_y = t_y \sum_{x \setminus S_{xy}} t_x.$$

Det vil sige, at absorberingen udelukkende flytter om på rækkefølgen, således at kompleksiteten af den samlede beregning reduceres.

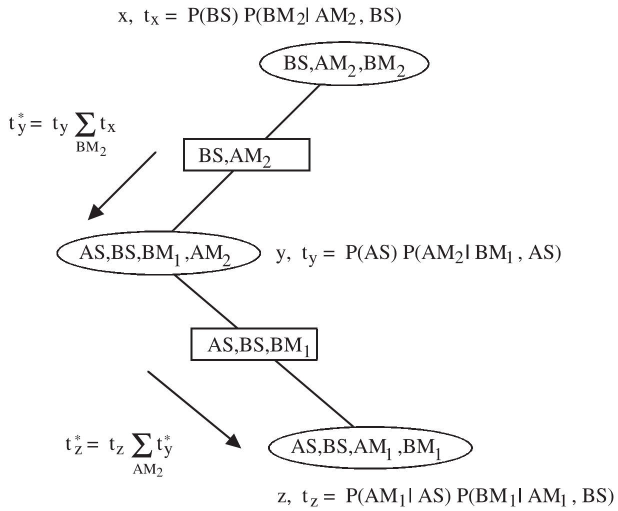
Fra Algoritme 3.13 på side 34 haves, at tabellerne for hver knude, x, i J er giver ved multiplikation af n_x betingede sandsynligheder

$$t_x = \prod_{j=1}^{n_x} P(V_{(j)} | pa(V_{(j)})),$$

hvor $V_{(j)}$ er en variabel i det bayesianske netværk over $U = \{V_1, \ldots, V_n\}$. Dette gør, at en sekvens af absorberinger i rigtig rækkefølge og retning faktiskt beregner $P(x)$ for en knude i J. Herfra kan $P(V_{(j)})$ for et $V_{(j)} \in x$ bestemmes. Beregningen svarer til at finde $P(U)$ vha. kædereglen og siden marginalisere, uden at hele $P(U)$ på noget tidspunkt beregnes. Vi vil i det følgende beskæftige os med et eksempel, der illustrerer ovenstående.

Eksempel 3.18

Betragt beregningen af $P(BM_1)$ i Figur 3.6, hvor y først absorberer fra x, hvorefter z absorberer fra y.



Figur 3.6: Junction træ for det simplificerede tænkeboks, med tilknyttede tabeller i initialstillingen, hvor separatorerne er tilknyttet $\underline{1}$ tabeller. $P(z)$ beregnes ved en sekvens af absorberinger.

Heraf fås, at

$$\begin{aligned} t_z^* &= t_z \sum_{AM_2} t_y \sum_{BM_2} t_x \\ &= P(AM_1|AS)P(BM_1|AM_1, BS) \sum_{AM_2} P(AS) \cdot \\ &\quad P(AM_2|BM_1, AS) \sum_{BM_2} P(BS)P(BM_2|AM_2, BS). \quad (3.5) \end{aligned}$$

Sammenligning med kædereglen på side 28 giver, at

$$t_z^* = \sum_{AM_2, BM_2} P(U) = P(AS, BS, AM_1, BM_1).$$

Heraf fås, at $P(BM_1) = \sum_{\{AS,BS,AM_1\}} t_z^*$. Bemærk, at vi på intet tidspunkt i udregningen har hele $P(U)$ repræsenteret.

$\diamond$

Kompleksiteten af (3.5) i ovenstående eksempel er $24 + 80 + 100 = 204$ produkter og $16 + 60 + 80 + 10 + 5 = 171$ summer. Dette er samme kompleksitet, som vores hidtil bedste bud fundet i udtryk (3.3) på side 30.

Gennem Eksempel 3.18 har vi vist sammenhængen mellem absorbering i junction træet og beregning af sandsynligheder vha. kædereglen. Det er nu nødvendigt mere formelt at bevise sammenhængen mellem de tilknyttede tabeller i junction træet og egentlige sandsynligheder. Desuden er det nødvendigt at fastlægge rækkefølgen, hvormed absorberingerne skal foretages.

3.3.2 Sammenhængen mellem tabellerne og sandsynlighederne

En vigtigt egenskab ved Hugin-arkitekturen er sammenhængen mellem tabellerne på knuderne og separatorerne set i forhold til den simultane fordeling af alle variable i det bayesianske netværk.

Sætning 3.19

Lad BN være et bayesiansk netværk over universet U, og lad J være et tilsvarende junction træ for BN. Før absorbering gælder det, at $t_U = P(U)$, hvor

$$t_U = \frac{\prod_{x \in V(J)} t_x}{\prod_{S_{xy} \in S(J)} t_{S_{xy}}}.$$

Er J støttende, vil t_U være uændret under absorbering.

Bevis

Efter konverteringen af BN til J gælder der for alle S_{xy}, at $t_{S_{xy}} = \underline{1}$. Konstruktionen via Algoritme 3.13 på side 34 gør endvidere, at for hver variabel $V_i \in U$ eksisterer netop et $x \in V(J)$ med

$$t_x = \cdot \prod_{j=1}^{n_x} P(V_{(j)}|pa(V_{(j)})),$$

hvorom der gælder, at netop ét $V_{(j)} = V_i$. Dette kombineres med kædereglen, hvilket medfører, at

$$t_U = \frac{\prod_{x \in V(J)} t_x}{\prod_{S_{xy} \in S(J)} \underline{1}} = \prod_{x \in V(J)} \prod_{j=1}^{n_x} P(V_{(j)}|pa(V_{(j)})) = \prod_{V_i \in U} P(V_i|pa(V_i)) = P(U).$$

Iagtag nu absorbtion over en vilkårligt kant $xy \in E(J)$, således at y absorberer fra x. Det vil sige, at

$$t_y^* = t_y \frac{t_{S_{xy}}^*}{t_{S_{xy}}}, \quad \text{hvor} \quad t_{S_{xy}}^* = \sum_{x \setminus S_{xy}} t_x.$$

Hermed haves, at

$$\frac{t_y^*}{t_{S_{xy}}^*} = \frac{t_y \frac{t_{S_{xy}}^*}{t_{S_{xy}}}}{t_{S_{xy}}^*} = \frac{t_y}{t_{S_{xy}}}.$$

Dette medfører, at t_U er uændret under absorbtion, da t_y og $t_{S_{xy}}$ er de eneste tabeller, der ændres under absorberingen, men forholdet mellem dem er konstant.

□

For enhver knude, x, i et junction træ skaber det næste resultat en forbindelse mellem tabellen, t_x, knyttet til x og simultanfordelingen af de variable, som x repræsenterer, i det bayesianske netværk.

Sætning 3.20

Lad J være et konsistent junction træ over U, og lad t_U være givet ved

$$t_U = \frac{\prod_{x_i \in V(J)} t_{x_i}}{\prod_{S_{ab} \in S(J)} t_{S_{ab}}}. \tag{3.6}$$

Da gælder det for ethvert $x \in V(J)$, at

$$t_x = \sum_{U \setminus x} t_U.$$

Bevis

Induktion på antallet af knuder i J.

Basis: $|V(J)| = 1$. Lad $V(J) = \{x\}$, da er $U = x$, $t_U = t_x$ og $U \backslash x = \emptyset$. Det vil sige, at $t_x = \sum_{U\backslash x} t_x = \sum_{U\backslash x} t_U$.

Induktionstrin: Antag, at sætningen holder for alle junction træer af orden $|V(J)| = k$. Vi vil udnytte dette til at vise, at sætningen også holder for $|V(J)| = k + 1$.

Lad J være et junction træ af orden $k + 1$, og lad x være et blad i J med nabo y. Betragt junction træet $J' = J\backslash x$ af orden k, universet af J' vil være mængden

$$U' = (U\backslash x) \cup S_{xy}. \tag{3.7}$$

Udtryk (3.6) giver nu, at

$$t_U = \frac{\prod_{x_i \in V(J')} t_{x_i} \cdot t_x}{\prod_{S_{ab} \in S(J')} t_{S_{ab}} \cdot t_{S_{xy}}} = t_{U'} \frac{t_x}{t_{S_{xy}}}. \tag{3.8}$$

Beregn nu $\sum_{x\backslash S_{xy}} t_U$. Af udtryk (3.8) fås, at

$$\sum_{x\backslash S_{xy}} t_U = \sum_{x\backslash S_{xy}} t_{U'} \frac{t_x}{t_{S_{xy}}}.$$

Vi har, at $(x\backslash S_{xy}) \cap U' = \emptyset$, pga. konstruktionen af U'. Derfor fås, at

$$\sum_{x\backslash S_{xy}} t_U = \sum_{x\backslash S_{xy}} t_{U'} \frac{t_x}{t_{S_{xy}}} = \frac{t_{U'}}{t_{S_{xy}}} \sum_{x\backslash S_{xy}} t_x.$$

Eftersom J er konsistent haves, at

$$\sum_{x\backslash S_{xy}} t_x = t_{S_{xy}} = \sum_{y\backslash S_{xy}} t_y.$$

Det vil sige, at

$$\sum_{x\backslash S_{xy}} t_U = \frac{t_{U'}}{t_{S_{xy}}} \sum_{x\backslash S_{xy}} t_x = \frac{t_{U'}}{t_{S_{xy}}} t_{S_{xy}} = t_{U'}. \tag{3.9}$$

Induktionshypotesen kombineret med udtryk (3.9) giver os, at $\forall z \in V(J')$:

$$t_z = \sum_{U'\backslash z} t'_U = \sum_{U'\backslash z} \left(\sum_{x\backslash S_{xy}} t_U \right)$$

Vi har, at $z \cap (x \backslash S_{xy}) = \emptyset$, da J er et junction træ, og dermed

$(U' \backslash z) \cup (x \backslash S_{xy}) = (U' \cup (x \backslash S_{xy})) \backslash z = ((U \backslash x) \cup S_{xy} \cup (x \backslash S_{xy})) \backslash z = U \backslash z.$

Det vil sige, at

$$t_z = \sum_{U' \backslash z} \left(\sum_{x \backslash S_{xy}} t_U \right) = \sum_{(U' \backslash z) \cup (x \backslash S_{xy})} t_U = \sum_{U \backslash z} t_U.$$

Vi har nu vist, at $\forall z \in V(J) \backslash x : t_z = \sum_{U \backslash z} t_U$. For at fuldføre beviset af sætningen mangler vi at vise, at dette også gælder, når $z = x$, dvs. at $t_x = \sum_{U \backslash x} t_U$.

Vi har, at $\sum_{U \backslash x} t_U = \sum_{U' \backslash S_{xy}} t_U$, da $S_{xy} \cap (U \backslash x) = \emptyset$ og pga. udtryk (3.7).

Af udtryk (3.8) fås, at

$$\sum_{U \backslash x} t_U = \sum_{U' \backslash S_{xy}} t_{U'} \frac{t_x}{t_{S_{xy}}} = \frac{t_x}{t_{S_{xy}}} \sum_{U' \backslash S_{xy}} t_{U'}.$$

Da $y \in V(J')$, kan vi bruge induktionshypotesen, $t_y = \sum_{U' \backslash y} t_{U'}$, til at vise,

$$\sum_{y \backslash S_{xy}} t_y = \sum_{y \backslash S_{xy}} \left(\sum_{U' \backslash y} t_{U'} \right) = \sum_{U' \backslash S_{xy}} t_{U'}.$$

Det vil sige, at

$$\sum_{U \backslash x} t_U = \frac{t_x}{t_{S_{xy}}} \sum_{y \backslash S_{xy}} t_y.$$

Eftersom J er konsistent fås, at

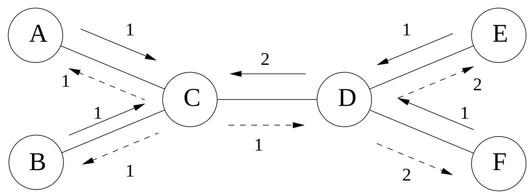
$$\sum_{U \backslash x} t_U = \frac{t_x}{t_{S_{xy}}} t_{S_{xy}} = t_x.$$

□

Det er lykkedes at vise sammenhænge mellem tabellerne på junction træet og sandsynlighederne af knuderne. Næste skridt er at systematisere rækkefølgen af absorberingerne og vise, at med den rette rækkefølge kan kriteriet om konsistens af junction træet opnås.

3.3.3 Rækkefølgen af absorberingerne

I Eksempel 3.18 på side 40 så vi, at den rette rækkefølge af absorberinger medførte, at en knude modtog den nødvendige information til at beregne sandsynligheder af variable efter. Dette kan både foregå i initialstillingen eller efter, at netværket har fået tilført evidens. Sandsynlighederne vil selvfølgeligt afhænge af, hvilken evidens netværket har fået tilført. Til fastlæggelse af rækkefølgen defineres derfor algoritmerne *SamlEvidens* og *SpredEvidens*. *SamlEvidens* svarer til, at en knude spørger alle de andre knuder i træet om deres information, hvorved informationen sendes fra bladene hen mod knuden, se Figur 3.7. *SpredEvidens* benyttes, når en knude ønsker at fortælle alle andre knuder i netværket om sin information. Herved sendes informationen fra knuderne ud mod bladene, se Figur 3.7.



Figur 3.7: Opdatering af sandsynligheder ved først *SamlEvidens* kaldt fra C (fuldt optrukne pile), efterfulgt af *SpredEvidens* fra C (stiplede pile). Tallene markerer, i hvilken rækkefølge informationen sendes.

Lad J være et junction træ. En indsamling af evidens igangsættes fra et $r \in V(J)$, knuden virker dermed som rod for indsamlingen af informationen. Algoritmen benytter to parametre, den første fortæller, hvilken knude der skal undersøges, og den anden parameter fortæller, hvilken knude *SamlEvidens* blev kaldt fra. Første kald sker vha. $SamlEvidens(r, r)$.

$SamlEvidens(r, y)$ =
- For alle r's naboer, x, hvor $x \neq y$
  - Kald $SamlEvidens(x, r)$.
- Hvis $r \neq y$, så lad y absorbere fra r.

Ved at kalde $SamlEvidens(r, r)$ for et $r \in V(J)$, får r fra de andre knuder tilført alt den information om sandsynligheder, som er nødvendig for at beregne $P(r)$ — også efter eventuel tilført evidens. Bemærk at sekvensen af absorberinger i Eksempel 3.18 på side 40 svarer til et kald af *SamlEvidens* med z som rod.

For at beregne sandsynligheden $P(r)$ for alle knuder i junction træet er det i princippet muligt at kalde *SamlEvidens* for hver enkelt knude. En mere effektiv løsning er dog kun at kalde *SamlEvidens* for én knude, hvorefter den indsamlede information spredes vha. en *SpredEvidens* algoritme. På denne måde udnyttes separatorerne til at spare et antal beregninger. Efter kald af *SamlEvidens* er separatorerne ikke længere bare lig med $\underline{1}$.

$SpredEvidens(r, y)$ = • For alle r's naboer, x, således at $x \neq y$
– lad x absorbere fra r.
– kald $SpredEvidens(x, r)$.

Hvis vi kun anvendte $SamlEvidens$, kunne separatorerne undværes, da de i Hugin-arkitekturen først er nødvendige i $SpredEvidens$. Men i bayesianske netværk er vi interesseret i at finde sandsynlighedsfordelingen for enhver variabel, dvs. at hvis vi kun havde $SamlEvidens$, så skulle vi kalde $SamlEvidens$ for hver variabel. Dette vil medfører, at vi vil få udført en del identiske beregninger på tabeller mere end én gang; mens dette kunne være gjort med et enkelt kald af $SamlEvidens$ og derefter et enkelt kald af $SpredEvidens$, hvorefter sandsynlighedsfordelingerne kan aflæses i de respektive knuder i junction træet, dog skal der fra knuden foretages en mindre marginalisering ned på den pågældende variabel.

Følgende egenskab gælder for de to algoritmer.

Sætning 3.21

Lad $x \in V(J)$. Efter på hinanden følgende kald af først $SamlEvidens(x, x)$ og dernæst $SpredEvidens(x, x)$ er J konsistent.

Bevis

Først vises, at efter sekvensen af kald gælder for enhver kant $xy \in E(J)$, at x har absorberet fra y og omvendt. Dette gøres ved induktion i antallet af knuder.

Basis er $|V(J)| = 1$. Da der ikke er nogen kanter, er den gensidige absorbtion trivielt opfyldt.

Antag nu, at den gensidige absorbtion holder for alle J, hvor $|V(J)| = k$. Vi skal vise, at et vilkårligt junction træ, J, hvor $|V(J)| = k + 1$, også opfylder kravet om gensidig absorbtion.

Vælg et blad y i J med nabo w. Sæt $J' = J\backslash y$. Betragt nu de to tilfælde, hvor roden x, enten opfylder $x \in J'$ eller $x = y$.

Hvis $x \in J'$, giver induktionhypotesen at sekvensen af kald kørt i J' medfører, at der for alle kanter i J' absorberes i begge retninger. Envidere vil kaldet af $SamlEvidens(x, x)$ i J' på et tidspunkt nå w. I J vil der i modsætning til J' også opstå et kald af $SamlEvidens(y, w)$. Herved vil w absorbere fra y. Da y er et blad, terminerer $SamlEvidens(y, w)$ og $SamlEvidens$ kan fortsætte på samme måde, som den ville have gjort i J'. Samme argumentation ud fra induktionshypotesen benyttes til at vise, at kaldet af $SpredEvidens(x, x)$ på et tidspunkt når til w, hvilket giver anledning til, at y absorberer fra w.

Hvis $x = y$, vil et kald af $SamlEvidens(y, y)$ i J svare til et kald af $SamlEvidens(w, w)$ i J' efterfulgt af, at y absorberer fra w. I J svarer kald af $SpredEvidens(y, y)$ til, at w først absorberer fra y, hvorefter $SpredEvidens(w, w)$ køres i J'

Næste skridt i beviset går ud på at vise, at hvis alle kanter i J absorberede i begge retninger, så er J konsistent. Dette gøres ved at betragte to situationer.

Hvis $|V(J)| = 1$ så er konsistens kravet trivielt opfyldt.

Hvis $|V(J)| > 1$ er det et krav, at for ethvert $x, y \in V(J) : xy \in E(J)$ gælder det, at x og y er konsistente.

Antag for nemheds skyld, at y først absorberede fra x, hvorefter x absorberede fra y. Heraf fås, at

$$t^*_{S_{xy}} = \sum_{x \setminus S_{xy}} t_x, \quad t^*_y = t_y \cdot \frac{t^*_{S_{xy}}}{t_{S_{xy}}}, \quad t^{**}_{S_{xy}} = \sum_{y \setminus S_{xy}} t^*_y \quad \text{og} \quad t^*_x = t_x \cdot \frac{t^{**}_{S_{xy}}}{t^*_{S_{xy}}}.$$

Resultatet heraf er, at

$$\sum_{x \setminus S_{xy}} t^*_x = \sum_{x \setminus S_{xy}} t_x \cdot \frac{t^{**}_{S_{xy}}}{t^*_{S_{xy}}} = \frac{t^{**}_{S_{xy}}}{t^*_{S_{xy}}} \sum_{x \setminus S_{xy}} t_x = \frac{\sum_{y \setminus S_{xy}} t^*_y}{\sum\limits_{x \setminus S_{xy}} t_x} \sum_{x \setminus S_{xy}} t_x = \sum_{y \setminus S_{xy}} t^*_y.$$

Hvilket viser, at J er konsistent.

$\square$

Sekvensen af først *SamlEvidens*(x, x) efterfulgt af et *SpredEvidens*(x, x) for et $x \in V(J)$ kaldes en **propageringsrunde**. Bemærk, at hvis man kun er interesseret i den marginale fordeling af en knude, så er det nok kun at kalde *SamlEvidens* med rod i denne knude.

3.3.4 Sammenkædning af bayesianske netværk og junction træer

Vi kan nu se på, hvorledes sandsynligheder om variable i et bayesiansk netværk kan uddrages fra et tilsvarende junction træ.

Sætning 3.22

Lad BN være et bayesiansk netværk over variablene U, og lad J være et junction træ, der repræsenterer BN. Givet evidensen $e = \{\underline{f}_1, \ldots, \underline{f}_n\}$, hvor $\{\underline{f}_1, \ldots, \underline{f}_n\}$ er findings på variablerne $\{V_1, \ldots, V_n\}$. Vælg da for hvert $\underline{f}_i$ et $x \in V(J)$, således at $V_i \in x$ og multiplicer f_i på t_x. Foretag nu en propageringsrunde i J.

Efter denne propageringsrunde gælder for ethvert $y \in V(J)$, at

$$t_y = \sum_{U \setminus y} P(U, e) = P(y, e), \quad \text{samt} \quad P(e) = \sum_y t_y$$

Ligeledes haves for enhver separator $S_{xy} \in S(J)$, at $t_{S_{xy}} = P(S_{xy}, e)$.

Bevis

Sætning 3.4 på side 31 giver, at $P(U,e) = P(U) \cdot \underline{f}_1 \cdots \underline{f}_n$. For ethvert $\underline{f}_i$, vælg et $x \in V(J) : V_i \in x$, og sæt $t_x = t_x \cdot \underline{f}_i$, så opnås ifølge Sætning 3.19 på side 41, at

$$t_U = \frac{\prod_{x \in V(J)} t_x}{\prod_{S_{xy} \in S(J)} t_{S_{xy}}} \cdot \underline{f}_1 \cdots \underline{f}_n = P(U,e) \tag{3.10}$$

Lad y være en knude i J. Efter en runde med propagering vil J ifølge Sætning 3.21 på side 46 være konsistent. Desuden giver Sætning 3.19 på side 41, at t_U fra udtryk (3.10) vil være uændret under denne propagering.

Det haves nu fra Sætning 3.20 på side 42, at $t_y = \sum_{U \setminus y} t_U$. Dermed haves, at

$$t_y = \sum_{U \setminus y} t_U = \sum_{U \setminus y} P(U,e) = P(y,e).$$

Og derfor også automatisk, at $P(e) = \sum_y P(y,e) = \sum_y t_y$.

Da junction træet er konsistent benyttes ovenstående til at vise, at for ethvert $S_{xy} \in S(J)$ gælder det, at

$$\begin{aligned} t_{S_{xy}} &= \sum_{x \setminus S_{xy}} t_x = \sum_{x \setminus S_{xy}} P(x,e) = P(S_{xy},e), \quad \text{og} \\ t_{S_{xy}} &= \sum_{y \setminus S_{xy}} t_y = \sum_{y \setminus S_{xy}} P(y,e) = P(S_{xy},e). \end{aligned}$$

□

Som en følge af Sætning 3.22 har vi, at hvis der ikke er indløbet evidens for nogen variabel i det bayesianske netværk, dvs. $e = \emptyset$, da gælder der for ethvert $y \in V(J)$, at $t_y = \sum_{U \setminus y} P(U) = P(y)$, og for enhver separator $S_{xy} \in S(J)$, at $t_{S_{xy}} = P(S)$.

Ønskes sandsynlighederne for en variabel, V_i, i det bayesianske netværk, gøres dette på følgende måde. Først vælges en knude x i det tilhørende junction træ J, således at V_i tilhører den klike som x repræsenterer, dvs. $V_i \in x$. Herefter følges anvisningerne fra Sætning 3.22. Med disse valg fås, at

$$P(V_i,e) = \sum_{x \setminus \{V_i\}} P(x,e).$$

Det er denne fremgangsmåde, som blev benyttet i Eksempel 3.18 på side 40. Valget af x er ikke entydigt bestemt, eftersom V_i kan være indeholdt i flere knuder af J's knuder, se Figur 3.6 på side 40. Da et konsistent junction træ ifølge Sætning 3.16 på side 37 også er globalt konsistent, vil valget af x dog være underordnet, fordi to knuder i træet indeholder samme information om deres fællesmængde og dermed specielt om V_i.

3.4 Perspektivering

Propagering vha. Hugin-arkitekturen er ikke kun nyttig til beregning af sandsynligheder vha. produkter og summer. Propagering, som vi har præsenteret i dette kapitel, kaldes sum-propagering, mens arkitekturen f.eks. også kan benyttes til max-propagering. Max-propagering er ækvivalent til sum-propagering, men i stedet for summer benyttes max operationer. Ved at max-propagere findes den mest sandsynlige konfiguration af variablene i netværket. Da den distributive lov også gælder for max operationen, vil alle sætningerne for sum-propagering også gælde for max-propagering.

På nuværende stadium vil vi være i stand til at kunne skrive et program, der kan propagere i et bayesiansk netværk. Det vil være forholdsvist let at programmere rutiner til propagering i junction træet, det er blot at multiplicere og dividere nogle tabeller. Det svære ligger i at konvertere et bayesiansk netværk til et repræsentativt junction træ.

Kapitel 4

Spilteori

Til modellerne har vi behov for at finde en strategi for computerspilleren. I Kapitel 2 brugte vi vores intuition og sandsynlighedsberegning til at finde en strategi for Arne. Men det ville være rart, om vi havde et redskab, der kan finde den eller de bedste strategier for spillerne.

Spilteori er en matematisk teori, der netop forsøger at finde den bedste strategi for hver spiller i spillet ud fra viden om, at modspillerne reagerer på de andre spilleres handlinger. Det er vigtigt for teorien, at alle spillerne ved, at deres handlinger indvirker på, hvilke valg de resterende spillere vil foretage.

Vi vil benytte spilteori på den simple model af tænkeboks fra Kapitel 2, men først vil vi i dette kapitel beskrive de dele af spilteorien, som vi vil benytte. Kapitlet er baseret på [Rasmussen, 1989].

4.1 Indledende definitioner

Vi skal for spillet kunne angive spillere, strategier og dertil hørende udbytte. Ved en **spiller** forstås et individ, der gennem en række valg i spillet forsøger at maksimere sin gevinst.

Mange spil er baseret på indgreb udefra, dvs. der bliver foretaget valg, som ingen af spillerne er direkte ansvarlige for, f.eks. gennem terningkast. I spilteori introduceres derfor **naturen**, som står for at foretage disse valg.

Spillerne skal gennem spillets forløb foretage en række valg, i spilteori kaldes disse valgmuligheder for **handlinger**. De handlinger, spiller i har mulighed for at foretage på et givet tidspunkt i spilforløbet, betegnes spillerens **handlingsmængde**, A_i, til dette tidspunkt.

Under spillets forløb, skal spillerne gentagne gange foretage valg ud fra spillets situation. En fastlægning af, hvilken handling en spiller skal foretage i en vilkårlig situation, kaldes for spillerens strategi.

DEFINITION 4.1 (STRATEGI)
En spillers **strategi** er et regelsæt, der fortæller, hvilken handling han

skal vælge til ethvert tidspunkt i et spil ud fra hans informationsmængde[1].

Mængden bestående af alle mulige strategier, s_i, en spiller, i, har til rådighed, kaldes en **strategimængde**, S_i. En **strategikombination**, s, er en ordnet mængde indeholdende en strategi, s_i, for hver af de n spillere. Det vil sige, at $s = (s_1, \dots, s_n)$.

For at afgøre hvilken strategi, der er bedst for en spiller, ses på spillerens udbytte.

Definition 4.2 (Udbytte)

Ved spiller i's **udbytte** forstås en funktion $\pi_i(s_1, \dots, s_n)$, som i det tilfælde, hvor naturen har et valg, efter alle spillere har truffet deres valg af strategier, er angivet ved:

(1) Det udbytte han får, efter at samtlige spillere og naturen har valgt strategier, og spillet er spillet til ende.

Det vil sige, at efter spillerners valg er der et stokatisk valg, der skal udføres, før spillet er færdig. Og i det tilfælde, hvor spillet er færdigt, efter alle spillere har truffet deres valg af strategier, er udbyttet angivet ved:

(2) Det forventede udbytte som en funktion af de strategier, der vælges af de n spillere.

Et spil kaldes et **nulsumspil**, hvis summen af samtlige spilleres udbytte er nul, ligegyldig hvilke strategier de vælger. Det vil sige, i et spil med to spillere vil den ene spiller altid vinde, hvad en anden spiller taber.

4.2 Ligevægt

Spilteoretikere ønsker at forudsige resultatet af et spil, derfor er det interessant at se på, hvilke strategier spillerne vil vælge i forskellige situationer, for dermed at kunne vælge en optimal strategi.

Hvis samtlige spillere har valgt en strategi, hvor det ikke er mulig at forhøje sit udbytte givet de andre spilleres valgte strategier, siges spillet at have opnået en **ligevægt**. En ligevægt er en naturlig udvikling af et spil, hvis det bliver spillet mange gange. De strategier, der indgår i en ligevægt, kaldes **ligevægtsstrategier**.

De to mest kendte ligevægtskoncepter er Nash ligevægt og dominerende strategi ligevægt.

[1]Informationsmængden definerer vi senere i Definition 4.8 på side 55, kort beskrevet er det spillerens viden på et givet tidspunkt i spillet.

Når vi diskuterer ligevægtskoncepter, indfører vi for en vilkårlig vektor, y, notationen y_{-i} til at betegne vektoren $(y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n)$.

4.2.1 Dominerende strategi ligevægt

Det første ligevægtskoncept, vi vil komme ind på, er dominerende strategi ligevægt, men først skal en dominerende strategi defineres.

Definition 4.3 (Dominerende strategi)
Spiller i's strategi, s_i^*, siges at være en **dominerende strategi**, hvis den er hans strengt bedste svar, uanset hvilke strategier de resterende spillere vælger:

$$\pi_i(s_i^*, s_{-i}) > \pi_i(s_i', s_{-i}) \; \forall \; s_{-i} \; \forall \; s_i' \neq s_i^*$$

Spiller i's resterende strategier, $s_i' \neq s_i^*$, kaldes for hans **dominerede strategier**.

Ud fra begrebet dominerende strategi defineres nu ligevægtskonceptet dominerende strategi ligevægt.

Definition 4.4 (Dominerende strategi ligevægt)
En ligevægt siges at være en **dominerende strategi ligevægt**, hvis den indeholder en dominerende strategi for hver spiller.

En dominerende strategi ligevægt er sjælden, da det for mange spil er umuligt at finde en strengt dominerende strategi for samtlige spillere. Derfor defineres konceptet Nash ligevægt.

4.2.2 Nash ligevægt

Definition 4.5 (Nash ligevægt)
En ligevægt kaldes en **Nash ligevægt**, hvis ingen spiller kan forhøje sit udbytte ved at afvige fra sin strategi, givet at ingen andre spillere ændrer strategi:

$$\forall \; i, \; \pi_i(s_i^*, s_{-i}^*) \geq \pi_i(s_i', s_{-i}^*) \; \forall \; s_i'$$

En Nash ligevægt kan fortolkes som situationen, når alle spillere offentliggører deres valg af strategier, og selv med denne information er det ikke muligt for nogle af spillerne at øge deres udbytte.

Definitionen af Nash ligevægt gør, at en ligevægt ikke nødvendigvis er entydigt bestemt. Dette er et generelt problem inden for spilteori, og der findes ingen løsning til at forudsige, hvilken ligevægt et spil vil ende i. Det følger ligeledes fra definitionen, at alle dominerende ligevægte også er Nash ligevægte, men ikke alle Nash ligevægte er dominerende ligevægte.

4.3 Information

En normalform er en måde at beskrive et spil på.

Definition 4.6 (Normalform)
En **normalform** består af:

1. Alle strategikombinationer, $s_1, s_2, \dots, s_p$.
2. Udbyttefunktioner fra s_i ind i n-udbytte-vektoren π_i, hvor $i = 1, 2, \dots, p$.

Antallet af spillere er n, og p er antallet strategikombinationer.

For hver enkel strategikombination viser normalformen, hvilket udbytte strategikombinationen resulterer i. Normalformen kan for to spillere angives vha. en matrice. Denne matrice har vi valgt at kalde for en **udbyttematrice**. Et eksempel på en sådan udbyttematrice ses nedenfor.

$$\begin{array}{cc} & \text{Spiller 2} \\ \text{Spiller 1} & \begin{bmatrix} (-2,-2) & (-5,0) \\ (0,-5) & (\mathbf{3},\mathbf{3}) \end{bmatrix} \end{array}$$

Rækkerne indikerer, at spiller 1 benytter sin hhv. første og anden strategi, og søjlerne, at spiller 2 benytter sin hhv. første og anden strategi. Tallene i parenteserne er hhv. spiller 1's og spiller 2's udbytte ved at vælge en bestemt strategi. Den fede skrift makerer den dominerende strategi ligevægt for dette spil. Et andet eksempel på en udbyttematrice ses nedenfor.

$$\begin{array}{cc} & \text{Spiller 2} \\ \text{Spiller 1} & \begin{bmatrix} (2,-2) & (\mathbf{2},-\mathbf{2}) \\ (-3,3) & (-2,2) \end{bmatrix} \end{array}$$

Her indikerer den fede skrift den Nash ligevægt, spillet har. Grunden, til at dette er en Nash ligevægt og ikke en dominerende strategi ligevægt, er, at udbyttet for ligevægten ikke er større end udbyttet for hver af alle de andre. Hvis spiller 1 benytter sin strategi 1 og spiller 2 sin strategi 1, vil spillerne få det samme udbytte som i Nash ligevægten.

Et spil kan ud over en normalform også beskrives vha. et udstrakt træ.

Først defineres et par grundlæggende begreber, der skal anvendes i definitionen af et udstrakt træ. Et udstrakt træ består af knuder og kanter imellem disse. **Knuderne** repræsenterer et sted i spillet, hvor en spiller eller naturen skal foretage en handling, eller hvor spillet slutter. **Kanter** beskriver en spillers handling på et bestemt tidspunkt. En knude kaldes en **startknude**, hvis den ikke har nogle forfædre, hvor en forfader til en knude X er en knude, der bliver mødt før X. En **slutknude** er en knude uden efterfølgere, hvor en knude Y er en efterfølger til en knude X, hvis Y forekommer senere i spillet end X. Man taler om en **vej** i træet, når man via en følge af knuder og kanter kan komme fra startkunden til en slutknude. Disse begreber kan nu bruges i følgende definition.

Definition 4.7 (Udstrakt træ)
Et **udstrakt træ** er en spilbeskrivelse bestående af:

1. En konfiguration af knuder og kanter, hvor der ikke eksistere kredse på nogen vej fra en startknude til en slutknude.
2. En entydig angivelse af, hvilken spiller hver enkel knude hører til.
3. De sandsynligheder, naturen bruger til at vælge, hvilke kanter den kan vælge imellem.
4. Informationsmængderne, hvori hver spillers knuder er opdelt.
5. Hver spillers udbytte ved hver slutknude.

Figur 4.1 viser et eksempel på et udstrakt træ for to spillere.

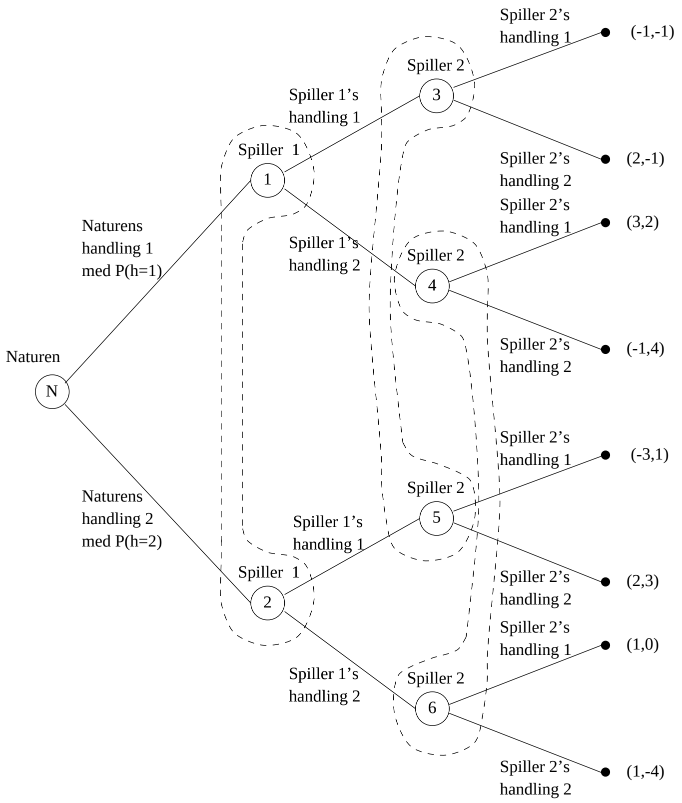
4.3.1 Informationsmængder

I definitionen af et udstrakt træ blev begrebet informationsmængde anvendt. Hermed defineres informationsmængde.

Definition 4.8 (Informationsmængde)
Spiller i's **informationsmængde**, ω_i, et vilkårligt sted i spillet er en mængde af forskellige knuder i et udstrakt træet. Spilleren ved, at han befinder sig i én af knuderne i informationsmængden, men han ved ikke præcis hvilken.

Det vil sige, at i enhver knude i en informationsmængde skal der være de samme valgmuligheder for en spiller, da han ellers vil kunne regne ud, hvilken knude han befinder sig i. På Figur 4.1 ses spillernes informationsmængder. Informationsmængderne $\{1, 2\}$, $\{3, 5\}$ og $\{4, 6\}$ eksistere kun i det tilfælde, hvor spillerene ikke kender naturens valg. Hvis f.eks. spiller 1 kender naturens valg, så vil han vide, om han er i knude 1 eller 2 og dermed have informationsmængderne $\{1\}$ og $\{2\}$.

Indtil nu har vi antaget, at alle spillerne kendte det udstrakte træs udseende, men faktisk ved alle spillere også, at alle spillere ved, hvordan det udstrakte træ ser ud, og alle spillere ved, at alle spillere ved, at alle ved, hvordan det udstrakte træ ser



Figur 4.1: Et eksempel på et udstrakt træ.

ud o.s.v. Dette kaldes **fælles viden**, og dvs. at information er fælles viden, hvis informationen er kendt for alle spillere.

4.4 Rene og randomiserede strategier

En strategi kan enten være ren eller randomiseret. En ren strategi er en regel, der fortæller en spiller præcist, hvilken handling spilleren skal vælge, mens en randomiseret strategi fortæller spilleren, med hvilke sandsynligheder spilleren skal vælge, hvilken handling der skal foretages.

De to typer strategier defineres i de to følgende definitioner.

Definition 4.9 (Ren strategi)

En **ren strategi** for spiller i er en funktion defineret på spiller i's mulige informationsmængder med værdier i mængden af handlinger:

$$s_i : \omega_i \to a_i$$

Det vil sige, at en ren strategi, s_i, for hver spiller, i, angiver præcist én handling for hver mulig informationsmængde.

Definition 4.10 (Randomiseret strategi)
En **randomiseret strategi** afbilleder enhver af en spillers informationsmængder til en sandsynlighedsfordeling over handlingerne:

$$s_i : \omega_i \to m(a_i), \quad \text{hvor } m(a_i) \geq 0 \text{ og } \sum_{a_i \in A_i} m(a_i) = 1$$

Ved en **komplet randomiseret strategi** er der kun positive sandsynligheder til enhver handling, således at

$$s_i : \omega_i \to m(a_i), \quad \text{hvor } m(a_i) > 0 \text{ og } \sum_{a_i \in A_i} m(a_i) = 1.$$

En udgave af et spil kaldes en **randomiseret udvidelse**, hvis spillet er udvidet med randomiserede strategier fra en udgave med rene strategier.

4.5 Randomiseret Nash ligevægt og begynderfordel

Vi vil i dette afsnit beskrive begreberne randomiseret Nash ligevægt og begynderfordel. Herudover vil en sætning om eksistens af Nash ligevægte blive anført.

4.5.1 Randomiseret Nash ligevægt

Vi vil beskrive en randomiseret Nash ligevægt, samt hvordan den findes vha. følgende spil:

$$\begin{array}{cc} & \text{Spiller 2} \\ \text{Spiller 1} & \begin{bmatrix} (2,-2) & (-2,2) \\ (-3,3) & (0,0) \end{bmatrix} \end{array}$$

I dette spil er der ingen rene Nash ligevægte, men der findes en randomiseret Nash ligevægt. Det vil sige, at spiller 1 med sandsynlighed $p \in [0,1]$ benytter strategi 1 og med sandsynlighed $1-p$ strategi 2, og spiller 2 benytter med sandsynlighed $q \in [0,1]$ strategi 1 og med sandsynlighed $1-q$ strategi 2. For at et spil er i en randomiseret Nash ligevægt, skal spiller 1 givet sandsynligheden q for spiller 2 have samme udbytte ved at vælge strategi 1 og 2. Det vil i dette tilfælde sige, at følgende ligning skal gælde:

$$2 \cdot q + (-2) \cdot (1-q) = -3 \cdot q + 0 \cdot (1-q)$$

Herudfra kan q findes:

$$\begin{aligned} 2 \cdot q + 3 \cdot q - 2 - 2 \cdot q &= 0 \Leftrightarrow \\ 7 \cdot q - 2 &= 0 \Leftrightarrow \\ q &= \frac{2}{7} \end{aligned}$$

Det samme skal gælde for spiller 2, hvilket vil sige, at følgende ligning også gælder:

$$-2 \cdot p + 3 \cdot (1 - p) = 2 \cdot p + 0 \cdot (1 - p)$$

Herudfra kan p findes:

$$\begin{aligned} -2 \cdot p - 2 \cdot p + 3 - 3 \cdot p &= 0 \Leftrightarrow \\ -7 \cdot p + 3 &= 0 \Leftrightarrow \\ p &= \frac{3}{7} \end{aligned}$$

I den randomiserede Nash ligevægt vil spiller 1, givet sandsynligheder for, hvordan spiller 2 randomiserer mellem sine strategier, have samme udbytte for strategi 1 og strategi 2. Det omvendte skal også gælde for spiller 2.

Det vil sige, at spillet har følgende randomiserede Nash ligevægt:

- Spiller 1 randomiserer mellem strategi 1 og 2 med hhv. $\frac{3}{7}$ og $\frac{4}{7}$ sandsyndlighed.
- Spiller 2 randomiserer mellem strategi 1 og 2 med hhv. $\frac{2}{7}$ og $\frac{5}{7}$ sandsyndlighed.

4.5.2 Begynderfordel og eksistens af Nash ligevægt

I et nulsumspil har den spiller, der starter spillet, en **begynderfordel**, hvis der eksisterer en Nash ligevægt, således at han får positivt udbytte. Det kan vises, at for et nulsumspil vil alle Nash ligevægte have samme forventet udbytte til spiller 1, og dermed også til spiller 2. Det vil sige, hvis en spiller har positivt udbytte i en Nash ligevægt, så vil han have det samme positive udbytte i de eventuelle. andre Nash ligevægte i spillet. Der kan altså f.eks. ikke være to Nash ligevægte med forskelligt udbytte x og y til spiller 1. Vi kan vise, at dette gælder generelt ved at se på et spils udbyttematrice.

$$\begin{array}{cc} & \text{Spiller 2} \\ \text{Spiller 1} & \begin{bmatrix} & \vdots & & \vdots & \\ \dots & (\mathbf{x_{ij}}, -\mathbf{x_{ij}}) & \dots & (x_{ik}, -x_{ik}) & \dots \\ & \vdots & & \vdots & \\ \dots & (x_{lj}, -x_{lj}) & \dots & (\mathbf{x_{lk}}, -\mathbf{x_{lk}}) & \dots \\ & \vdots & & \vdots & \end{bmatrix} \end{array}$$

Vi ser først på rene Nash ligevægte. Vi antager, at $(x_{ij}, -x_{ij})$ og $(x_{lk}, -x_{lk})$ er udbyttet for to Nash ligevægte, nemlig hvor spiller 1 benytter hhv. strategi i og l, og spiller 2 benytter hhv. strategi j og k. Da $(x_{ij}, -x_{ij})$ er en Nash ligevægt, vil $x_{lj} \leq x_{ij}$ og $-x_{ik} \leq -x_{ij}$. Det vil sige, at $x_{lj} \leq x_{ij} \leq x_{ik}$. På samme måde kan vi argumentere for Nash ligevægten $(x_{lk}, -x_{lk})$ og komme frem til, at $x_{ik} \leq x_{ij} \leq x_{lj}$. Det vil sige, at $x_{ij} = x_{lk}$, ergo har alle rene Nash ligevægte ens udbytte i et nulsumspil.

Den samme argumentation kan bruges for randomiserede Nash ligevægte samt rene kontra randomiserede Nash ligevægte, hvor en celle i, j i matricen blot angiver, at spiller 1 benytter den rene eller randomiserede strategi i, og spiller 2 benytter den rene eller randomiserede strategi j. Af denne grund giver definitionen af begynderfordel mening, og det er altså ikke muligt, at begynderfordelen afhænger af, hvilken Nash ligevægt der betragtes.

Ifølge [Myerson, 1991] gælder følgende vigtige sætning, som vi vil anføre uden bevis.

Sætning 4.11

For ethvert endeligt spil S eksisterer der mindst én Nash ligevægt.

4.6 Motivation af spilteorien

Grunden, til at spilteorien er særdeles interessant, er, at hvis man kender en af Nash ligevægtene for et spil, så ved man, at hvis strategien i Nash ligevægten benyttes, er det den, der giver det bedste udbytte uafhængig af modspillernes valg af strategier.

Dette vil vi vise ved følgende overvejelse, hvor vi har valgt at betragte et nulsumspil. Tænkeboks er netop et nulsumspil, hvilket vi vil argumentere for i det følgende kapitel.

Vi antager, at vi har to hold, hvor hvert hold består af to spillere, en spiller for hver af situationerne, når holdet hhv. sidder i forhånd og baghånd. Det ene hold kaldes Nash holdet, og det andet kaldes Verdensholdet. Spilleren, der sidder i forhånd på Nash holdet, betegnes N_F, og spilleren der sidder i baghånd N_B. Ligeledes betegnes spillerne på verdensholdet V_F og V_B. Nash holdet benytter altid en strategi fra en Nash ligevægt, mens Verdensholdet, som består af verdensmestrene i spillet, benytter den strategi, de mener, er den bedste. Hvis Nash holdet spiller mod Verdensholdet, dvs. N_F spiller mod V_B og V_F mod N_B, så vil det samlede udbytte for Nash holdet blive større end eller lig med nul.

Dette kan vi vise ved at betragte udbyttematricen for spillet i Figur 4.2.

Hvis vi ser på det tilfælde, hvor N_F spiller mod V_B, ses det ud fra matricen, at udbyttet er hhv. x_{lk} og $-x_{lk}$. Fra definitionen af Nash ligevægt kan det ses, at $-x_{lj} \geq -x_{lk}$. I tilfældet, hvor V_F spiller mod N_B, ses det yderligere, at $x_{lj} \geq x_{ij}$. Ud fra disse betragtninger kan vi opstille følgende ulighed:

$$
\begin{array}{ccc}
 & & \text{Baghånd} \\
 & & \begin{array}{cc} N_B & V_B \end{array} \\
\text{Forhånd} & \begin{array}{c} \\ V_F \\ \\ N_F \\ \\ \end{array} & \begin{bmatrix}
 & \vdots & & \vdots & \\
\dots & (x_{ij}, -x_{ij}) & \dots & (x_{ik}, -x_{ik}) & \dots \\
 & \vdots & & \vdots & \\
\dots & (\mathbf{x_{lj}}, -\mathbf{x_{lj}}) & \dots & (x_{lk}, -x_{lk}) & \dots \\
 & \vdots & & \vdots &
\end{bmatrix}
\end{array}
$$

Figur 4.2: Udbyttet i matricen er angivet på formen (forhånd, baghånd), hvor udbyttet for Nash ligevægten er angivet med fed skrift.

$$
\begin{aligned}
x_{lk} \geq x_{lj} &\geq x_{ij} \Leftrightarrow \\
x_{lk} + (-x_{ij}) &\geq 0
\end{aligned}
$$

Dermed har vi vist, at det samlede udbytte for Nash holdet altid vil være større end eller lig med nul i et nulsumspil. Det vil sige, at Nash holdet altid vil vinde eller få uafgjort mod et hvilket som helst andet hold.

Vi har i dette kapitel opstillet de definitioner og begreber for spilteorien, vi skal benytte fremover. Vi vil i næste kapitel benytte spilteorien på den samme udgave af tænkeboks, som vi konstruerede et influensdiagram for i Kapitel 2.

Kapitel 5

Spilteori for simplificeret tænkeboks

Vi vil i dette kapitel foretage en spilteoretisk undersøgelse af det tænkeboks, som blev beskrevet i Kapitel 2. Specielt ønsker vi at finde de bedste strategier for hver af spillerne, og om der er en begynderfordel i spillet. For at gøre dette vil vi finde strategimængderne for Arne og Bente samt opstille en udbyttematrice for strategierne. Udbyttet for den enkelte spiller angives ved 1, hvis spilleren vinder og -1 ellers, dvs. at tænkeboks er et nulsumspil. Ud fra denne udbyttematrice bestemmes eventuelle Nash ligevægte, der kan benyttes til undersøgelser af, om der er en begynderfordel.

5.1 Strategier

Vi ser først på Arnes mulige rene strategier, dvs. vi ønsker at bestemme hans strategimængde. Da der kun er fem mulige meldinger, 1, 2, 2·1, 2·2 og løft, vil Arne i værste tilfælde få meldingen tre gange i et spil. Den tredje gang er altid løft, efter at der er meldt 2·2. Vi betragter spillet som slut, når en spiller melder løft. Arne har dermed én startmelding og evt. én reel melding mere. Hans startmelding kan være 1, 2, 2·1 og 2·2, altså fire mulige meldinger. Melder Arne 1, kan han få 2, 2·1 eller 2·2 tilbage, dette er vist i Tabel 5.1.

Arnes første melding som kan give en andenmelding	1	2	2·1
Meldinger Arne kan få tilbage	2, 2·1, 2·2	2·1, 2·2	2·2

Tabel 5.1: Mulige første meldinger for Arne, samt hvilke meldinger det kan resultere i fra Bente.

Hvis Bente melder 2, har han tre mulige tilbagemeldinger og ved 2·1 to mulige tilbagemeldinger. Dette samt alle andre tilfælde er vist i Tabel 5.2.

Bentes første melding	2	2·1	2·2
Arnes mulige tilbagemeldinger	2·1, 2·2, løft	2·2, løft	løft

Tabel 5.2: Mulige tilbagemeldinger for Arne.

Arne har mange forskellige strategier, hvor en strategi f.eks. er:

		Arnes terning viser 1, han melder	Arnes terning viser 2, han melder
1. runde		1	2
2. runde	Bente meldte 2	2·1	2·2
	Bente meldte 2·1	2·2	2·2
	Bente meldte 2·2	løft	løft

Ud fra Tabel 5.1 og Tabel 5.2 kan vi finde alle Arnes strategier. Disse kan også findes ud fra det udstrakte træ for spillet på Figur 5.1. Vi har kun medtaget undertræet af det udstrækte træ, hvor både Arne og Bente har slået 1. De andre undertræer ser ud på samme måde. Disse undertræer bliver sat sammen ved, at naturen først vælger, hvad hhv. Arne og Bente har slået, se Figur 5.2 på side 64.

Hvis f.eks. Arne har meldt 1, og Bente melder 2, så har han tre mulige tilbagemeldinger at vælge imellem, nemlig 2·1, 2·2 og løft. Bente har også muligheden for at melde 2·1, som kan give anledning til to mulige tilbagemeldinger for Arne. Da en strategi skal angive en tilbagemelding på samtlige af Bentes meldinger, bliver der 2×3 mulige kombinationer, når Arne lægger ud med at melde 1. Arnes startmelding kan også være 2, 2·1 eller 2·2, der alle bidrager med forskellige nye strategier. Af denne grund skal det antal mulige kombinationer, hver startmelding bidrager med, summeres. Det mulige antal strategier, hvor Arne slår 1, er angivet i Tabel 5.3.

		Hvis Arne melder 1	2	2·1	2·2	I alt
Hvis Bente melder	2	3				
	2·1	2	2			
Mulige kombinationer		6	2	1	1	10

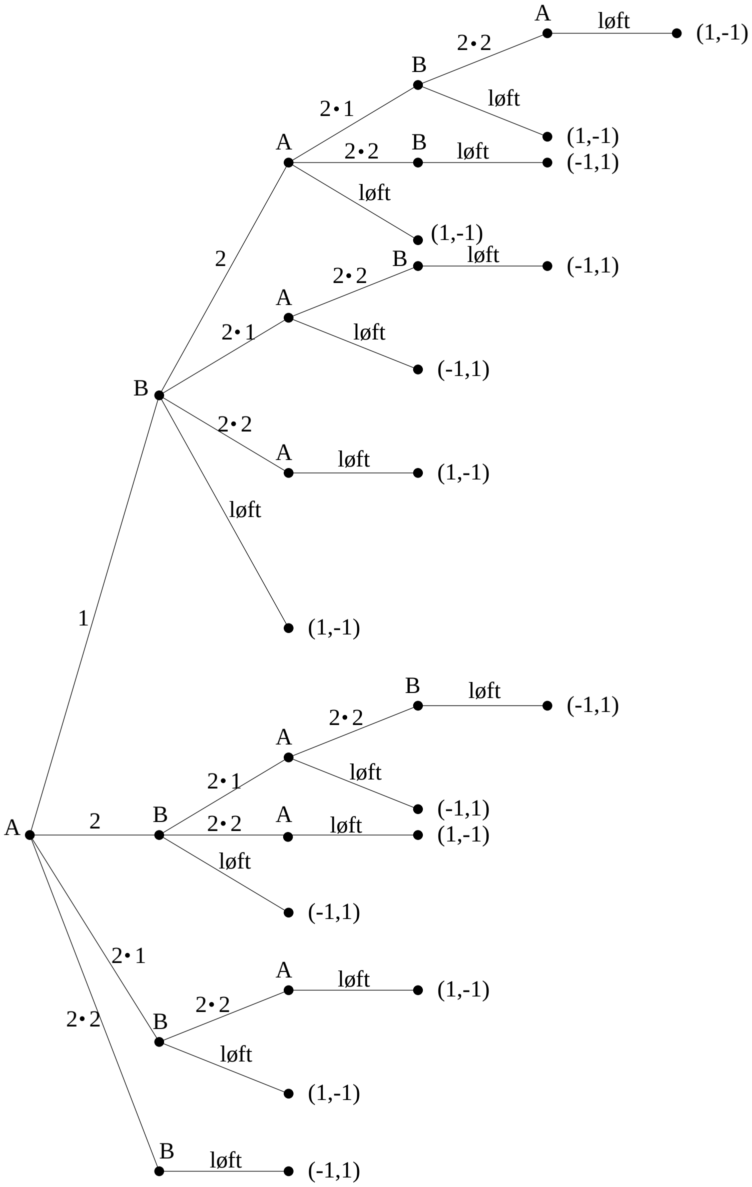
Tabel 5.3: Antal mulige strategier for Arne.

Hvis Arnes terning viser 1, er der dermed ti mulige strategier. Hvis terningen viser 2, er der de samme ti mulige meldinger. Disse meldinger kan blandes vilkårligt, hvilket giver 100 strategier i alt.

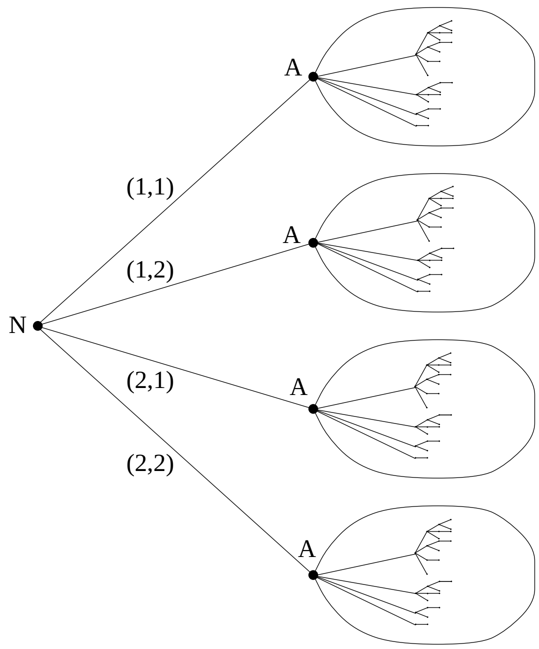
Bente har 900 mulige strategier, hvor en strategi f.eks. er givet ved:

Arne meldte	1		2		2·1		2·2	
Bentes terning viser	1	2	1	2	1	2	1	2
Bente melder	2·1	2	2·2	2·2	2·2	2·2	løft	løft
Arne melder		2·1						
Bente melder		løft						

At der er 900 forskellige strategier ses ud fra Tabel 5.4 på side 64, eftersom der er 30 kombinationer, når hun har slået 1, og dermed også 30 kombinationer, når hun har slået 2. Hermed er der i alt $30 \times 30 = 900$ kombinationer. Hvis f.eks. Arne melder 1 og Bentes terning viser 1, så har Bente fire mulige meldinger nemlig 2, 2·1, 2·2 og løft. I det tilfælde, hvor hun melder 2, og Arne melder 2·1, har hun en melding, hvor



Figur 5.1: Et undertræ af det udstrækte træ for spillet, hvor udbyttet er angivet på formen (Arnes udbytte, Bentes udbytte).



Figur 5.2: Det udstrækte træ for spillet.

hun kan vælge mellem 2·2 og løft. Det vil sige, at Bente har fem mulige meldinger, hvis Arne melder 1.

		Antal mulige meldinger
Hvis Arne melder	1	5
	2	3
	2·1	2
	2·2	1
Mulige kombinationer		30

Tabel 5.4: Antal mulige strategier for Bente, når hun har slået 1.

Dimensionen af udbyttematricen bliver dermed 100×900, hvilket er en uhåndterlig matrice. Af denne grund vil vi gerne reducere matricen, dette gøres ved at fjerne de strategier, der åbenlyst er domineret af andre. Vi ser på de meldinger, spillerne aldrig vil melde, samt de meldinger, de altid vil vælge at melde i en given situation. Disse meldinger er:

Aldrig melde 2·2, hvis man har slået 1.
Aldrig melde 2·1, hvis man har slået 2.

Aldrig løfte på 1, hvis man har slået 1.
Aldrig løfte på 2, hvis man har slået 2.

Altid løfte på 2·1, hvis man har slået 2.
Altid løfte på 2·1, hvis man har slået 1.

Den sidste regel er måske ikke helt åbenlys, men har man slået 1, er det en dårlig idé at melde 2·2, da man så vil tabe, da der med sikkerhed ikke er 2·2 på bordet, hvorimod at der med 50% sandsynlighed er 2·1. Bemærk, at den anden regel er den eneste, hvor vi antager, at modstanderen opfører sig rationelt. Alle andre regler er uafhængige af mostanderens opførsel.

Ved at indføre disse regler fjernes størstedelen af strategierne. Hvis vi indfører reglerne for Arne, har han tolv strategier tilbage. Hans mulige antal strategier vil ikke længere være ens for både etteren og toeren. Vi kan for de to mulige slag lave en tabel, der viser, hvordan vi kom frem til de tolv strategier. Tabel 5.5 ser på det tilfælde, hvor Arne slog 1, og Tabel 5.6 på det tilfælde hvor Arne slog 2.

		Hvis Arne melder				I alt
		1	2	2·1	2·2	
Hvis Bente melder	2	2				
	2·1	1	1			
Mulige kombinationer		2	1	1	0	4

Tabel 5.5: Antal mulige meldinger for Arne, hvor hans terning viser 1.

		Hvis Arne melder				I alt
		1	2	2·1	2·2	
Hvis Bente melder	2	1				
	2·1	1	1			
Mulige kombinationer		1	1	0	1	3

Tabel 5.6: Antal mulige meldinger for Arne, hvor hans terning viser 2.

Det vil sige, at Arne har $4 \times 3 = 12$ strategier. Bente vil også kun have tolv strategier tilbage, hvilket ses ud fra Tabel 5.7.

		Antal mulige meldinger, når Bente har slået		I alt
		1	2	
Hvis Arne melder	1	2	3	
	2	2	1	
	2·1	1	1	
	2·2	1	1	
Mulige kombinationer		4	3	12

Tabel 5.7: Antal mulige strategier for Bente efter reduceringen.

Efter denne reduktion er det blevet mere medgørligt at opstille de tilbageværende strategier for hver af spillerne. Resultatet ses i Tabel 5.8. Søjlen "start" i a) indeholder Arnes startmelding, hvorimod søjlen "2" giver modsvar til Bentes melding, når hun melder 2. Respons på meldingerne 2·1 og 2·2 er ikke medtaget, eftersom der kun løftes. Symbolet "-" angiver, at en mulighed ikke kan optræde.

#	start	2
1	2·1,2·2	-,-
2	2,2·2	-,-
3	1,2·2	2·1,-
4	1,2·2	L,-
5	2·1,2	-,-
6	2·1,1	-,L
7	2,2	-,-
8	1,2	2·1,-
9	1,2	L,-
10	2,1	-,2·2
11	1,1	2·1,2·2
12	1,1	L,2·2

a)

#	1	2
1	2,2	2·1,2·2
2	2,2·2	2·1,2·2
3	2,L	2·1,2·2
4	2·1,2	2·1,2·2
5	2·1,2·2	2·1,2·2
6	2·1,L	2·1,2·2
7	2,2	L,2·2
8	2,2·2	L,2·2
9	2,L	L,2·2
10	2·1,2	L,2·2
11	2·1,2·2	L,2·2
12	2·1,L	L,2·2

b)

Tabel 5.8: Arnes og Bentes tolv tilbageværende strategier i hhv. a og b. En enkelt celle er på formen x, y, hvor x er meldingen, når en spiller slår 1, mens y er meldingen, når spilleren slår 2. I a angiver den første søjle, at der er tale om Arnes startmelding, og den anden hvad Arne melder, hvis Bente meldte 2. I b angiver øverste række hvad Arnes startmelding var.

I Tabel 5.8 angiver søjlerne i b) modsvar til Arnes mulige startmeldinger. Respons på Arnes meldinger 2·1 og 2·2 er ikke medtaget, da Arne aldrig vil begynde med at melde disse. Det vil sige, at vi nu har en udbyttematrice med dimensionen 12×12.

Næste skridt er at beregne værdierne i denne matrice, dvs. at udregne spillernes udbytte, når de benytter de forskellige strategier. En værdi af en celle (i, j) i udbyttematricen er givet ved formlen

$$udbytte[i,j] = \sum_{a=1}^{2} \sum_{b=1}^{2} findVinder(A_i, B_j, a, b),$$

hvor $findVinder$ er en funktion, der simulerer udfaldet af et enkelt spil, når Arne og Bente hver har valgt hver deres strategi, hhv. A_i og B_j, og deres terningeslag er hhv. a og b. Vinder Arne, er $findVinder(A_i, B_j, a, b) = 1$, i tilfælde af en sejr til Bente er $findVinder(A_i, B_j, a, b) = -1$.

Beregningen af udbyttematricen udføres vha. et C++ program, som vi har implementeret til formålet, jf. Appendiks B. Ved at vælge en passende datastruktur til repræsentation af den enkelte spillers strategi, bliver det muligt at skrive en funktion med samme opførsel som $findVinder$.

Resultatet for den samlede udbyttematrice beregnes vha. programmet og kan ses i Tabel 5.9. Ud fra matricen er det nu let at bestemme, om en strategikombination opfylder kravet i Definition 4.5 på side 53 til en Nash ligevægt. Undersøgelsen er ligeledes blevet automatiseret af programmet. Nash ligevægtene er markeret i tabellen med fed skrift.

A\ B	1	2	3	4	5	6	7	8	9	10	11	12
1	**0**	0	0	0	**0**	**0**	**0**	0	0	0	**0**	**0**
2	**0**	0	0	0	**0**	**0**	**0**	0	0	0	**0**	**0**
3	0	2	2	-2	0	0	0	2	2	-2	0	0
4	0	2	2	-2	0	0	0	2	2	-2	0	0
5	**0**	0	0	0	**0**	**0**	**0**	0	0	0	**0**	**0**
6	-2	-2	-2	0	0	0	-2	-2	-2	0	0	0
7	**0**	0	0	0	**0**	**0**	**0**	0	0	0	**0**	**0**
8	0	2	2	-2	0	0	0	2	2	-2	0	0
9	0	2	2	-2	0	0	0	2	2	-2	0	0
10	0	-2	-2	2	0	0	0	-2	-2	2	0	0
11	**0**	0	0	0	**0**	**0**	**0**	0	0	0	**0**	**0**
12	**0**	0	0	0	**0**	**0**	**0**	0	0	0	**0**	**0**

Tabel 5.9: Udbyttematricen for Arne og Bentes 12×12 kombinationer af strategier. Nash ligevægtene er fremhævet med fed skrift.

For eksempel er spillet i en Nash ligevægt, når Arne benytter strategi 11, og Bente benytter strategi 1. Disse strategier siger, at ligegyldigt hvad Arne slår, så vil han starte med at melde 1, Bente vil melde 2, ligegyldigt hvad hun har, hvorefter Arne melder 2·1, hvis han har 1, og 2·2 hvis han har 2. Det vil sige, at hvis Arnes og Bentes terning viser det samme, så vil Arne vinde, da han melder to ens, idet hans terning viser, hvorimod hvis deres terninger viser forskelligt, så vil han tabe.

Det ses ud fra udbyttematricen, at der eksisterer et antal rene Nash ligevægte. Det forventede udbytte i alle disse Nash ligevægte viser sig at være 0. Det er derfor ikke muligt for nogen af spillerne at vælge en favorabel strategi, som i det lange løb vil kunne give dem et positivt udbytte. Heraf kan det endvidere konkluderes, at tænkeboks med de regler benyttet i dette afsnit ikke har en indbygget begynderfordel.

Vi har ikke beskæftiget os med randomiserede Nash ligevægte, da vi har fundet flere rene Nash ligevægte, og en randomiseret Nash ligevægt aldrig kan have et bedre udbytte end en ren, se afsnit 4.5.2.

5.1.1 En alternativ metode

At der kun er Nash ligevægte med udbytte på 0, kunne vi også forsøge at ræsonnere os frem til, sålænge problemet ikke er mere kompliceret end her. Vi begynder med at fjerne de strategier, der umiddelbart er domineret af andre, som vi også gjorde for at reducere i udbyttematricen. Herefter finder vi en strategi, først for Arne, som synes god, for herefter at tjekke det mere formelt. Vi vælger at se nærmere på følgende strategi:

Hvis Arnes terning viser 1, melder han 2·1.
Hvis Arnes terning viser 2, melder han 2·2.

Vi kan vise, at følgende lemma altid gælder.

Lemma 5.1

Med strategien ovenfor får Arne altid et forventet udbytte på 0, uanset hvilken strategi Bente følger.

Bevis

I de regler, vi opstillede tidligere, sagde vi bl.a., at der altid skal løftes på 2·1, så Bente vælger at løfte på begge meldinger fra Arne. Vi ser på, i hvilke tilfælde Arne vinder, angivet med 1, og hvilke han taber, angivet med -1, med de to strategier:

	Bentes terning viser 1	Bentes terning viser 2
Arnes terning viser 1	1	-1
Arnes terning viser 2	-1	1

Da der er 25% sandsynlighed for hvert tilfælde, er Arnes forventede udbytte altid 0 med denne strategi.

□

Der eksisterer et tilsvarende lemma for Bente som for Arne. For Bente har vi valgt at se nærmere på følgende strategi:

Arne meldte	1		2		2·1	
Bentes terning viser	1	2	1	2	1	2
Bente melder	2·1	løft	løft	2·2	løft	løft

Lemma 5.2

Med strategien ovenfor får Bente altid et forventet udbytte på 0, ligegyldigt hvad Arne gør.

Bevis

Vi ser på tilfældene et af gangen, hvor vi med 1 angiver, at Bente vinder, og med -1, at hun taber. Først det tilfælde, hvor Arne melder 1:

	Arnes terning viser 1	Arnes terning viser 2
Bentes terning viser 1	1	-1
Bentes terning viser 2	-1	1

Lad $p \in [0, 1]$ være sandsynligheden for at Arne melder 1, givet at han har 1, og lad $q \in [0, 1]$ være sandsynligheden for at Arne melder 1 givet, at han har 2. Det ses, at for alle værdier af p og q vil Bente vinde i 50% af tilfældene, hvis Arnes terning viser 1, og i 50% af tilfældene, hvis den viser 2. Det giver det samme resultat, når Arne melder 2. Hvis Arne derimod melder 2·1, betyder det ifølge vores reduceringsregler, at han har slået 1. Her har Bente også 50% chance for at vinde, nemlig i det tilfælde, hvor hun sidder med en toer. Det vil sige, at Bentes forventede udbytte er 0 med denne strategi.

□

Disse to lemmaer giver anledning til, at der kun findes Nash ligevægte med udbytte på 0. Dette ses ved at antage til modstrid, at der for Arne var en Nash ligevægt med udbytte på x, hvor $x > 0$. Dette vil medføre, at Bentes udbytte var $-x$, fordi tænkeboks er et nulsumspil. Da Bente har en strategi, hvor hun er garanteret et udbytte på 0, vil hun i stedet vælge denne. Det vil sige, at Arnes udbytte også er 0. Modsat vil Arne også vælge den strategi, der sikrer ham et udbytte på 0, hvis alternativet er mindre end 0.

Vi kunne dermed ræsonnere os frem til, at der kun er Nash ligevægte med udbytte på 0. Men vi kan ikke med sikkerhed ræsonnere os frem til, hvilke Nash ligevægte der er. Selvom vi kunne, så vil det også tage lang tid frem for at benytte spilteorien, som vi gjorde i afsnittet før. At ræsonnere sig frem til resultater bliver også mere og mere uoverkommeligt, jo mere kompliceret modellen er.

5.2 Spilteori og influensdiagrammet for det simple tænkeboks

Vi har i dette kapitel fundet udbyttematricen for Arnes og Bentes strategier på nær de umiddelbart dominerende strategier. Denne matrice vil vi anvende til at sammenholde spilteorien fra afsnit 5.1 og influensdiagrammet fra Figur 2.7 på side 18. Dette kan vi gøre ved at indtaste en strategi fra udbyttematricen i influensdiagrammet og evaluere dette, hvorved vi får forskellige optimale modstrategier. De fundne strategier sammenholdes med udbyttematricen fra Tabel 5.9 på side 67. De skulle gerne stemme overens med de strategier fra spilteorien, der giver det største udbytte for den spiller, vi forsøger at finde strategien for.

Vi kan f.eks. indtaste Arnes ottende strategi i influensdiagrammet og sammenligne resultatet med matricen. Konkret vil det sige, at vi skal se på de tilfælde, hvor Bente slår 1 og 2, kombineret med Arnes forskellige første meldinger. Ved hjælp af Hugin giver dette nogle meldinger, som siden sammensættes til en eller flere strategier for Bente. Den eller de strategier kan så sammenholdes med de strategier, udbyttematricen giver os som de bedste svar mod Arnes ottende strategi.

Vi ser nu på eksemplet med at fastsætte Arnes strategi til den ottende strategi. De mulige kombinationer er:

Arne melder 1, og Bente har slået 1: I denne situation får vi, at Bente vil melde $2 \cdot 1$.

Arne melder 1, og Bente har slået 2: I dette tilfælde vil Bente melde 2. Da Arne herefter vil melde $2 \cdot 1$, får hun meldingen tilbage igen, hvorefter hun vil løfte.

Arne melder 2, og Bente har slået 1: I denne situation vil Bente enten melde $2 \cdot 1$ eller løfte.

Arne melder 2, og Bente har slået 2: Her vil Bente melde $2 \cdot 2$.

Der er ikke flere tilfælde, da Arne ifølge sin strategi ikke vil melde 2·1 eller 2·2 i sin førstemelding, hvilket betyder, at vi ikke kan sætte evidens på disse, og se hvad Bente vil melde. Hvis vi sætter de ovenstående informationer sammen, fås følgende to strategier, der dog ikke helt er strategier, da de ikke kan tage højde for, at Arne melder 2·1 eller 2·2. Det betyder dog kun, at alle valg af strategi for $2 \cdot 1$ og $2 \cdot 2$ er lige gode, da situationen aldrig opstår.

Arne meldte	1		2	
Bentes terning viser	1	2	1	2
Bente melder	2·1	2	2·1	2·2
Arne melder		2·1		
Bente melder		løft		

Arne meldte	1		2	
Bentes terning viser	1	2	1	2
Bente melder	2·1	2	løft	2·2
Arne melder		2·1		
Bente melder		løft		

Disse to svarer til strategi 4 og 10 i udbyttematricen. Hvis vi sammenligner dette resultat med spilteoriresultatet, dvs. udbyttematricen, kan vi se, at hvis Arne benytter strategi 8, så er det netop Bentes strategi 4 og 10, der giver det bedste udbytte for Bente, nemlig udbyttet 2. På samme måde kan vi angive andre strategier for Arne og finde Bentes bedste modstrategier. Vi kan også angive strategier for Bente og finde Arnes bedste strategier. Dette kan vi gøre i det samme influensdiagram, hvor vi blot bytter om på Arne og Bente. Her skal vi blot huske at sætte evidensen på AM_1 til 0, og så sætte evidens på Arnes slag i BS.

Vi har bl.a. set på tilfældet, når Bente benytter strategi 7, her vil Arne benytte en vilkårlig strategi på nær strategi 6. Hvis Bente benytter strategi 2, vil Arne benytte strategi 3, 4, 8 eller 9. Også disse eksempler stemmer overens med udbyttematricen fra spilteorien.

På denne måde er der en forbindelse mellem spilteori og influensdiagrammer. I spilteorien finder vi Nash ligevægte, dvs. optimale strategier for flere spillere på samme tid, mens vi i influensdiagrammerne finder optimale strategier for én spiller, givet strategierne for de resterende spillere.

5.3 Opsummering

Gennem en reduktion af de åbenlyst dominerede strategier lykkedes det at reducere dimensionen af udbyttematricen fra en 100×900 til en 12×12 matrice. Selv om dette er en mere medgørlig størrelse, valgte vi alligevel at automatisere beregningen af matricens værdier gennem et program, hvorved vi fandt 36 Nash ligevægte.

På trods af at den version af tænkeboks vi undersøgte, var utrolig simpel, viste det sig, at spilteori er meget tungt at arbejde med. Derfor er det helt essentielt med sunde overvejelser for hurtigt at slette de åbenlyst dominerende strategier. Selv med en sådan indsats bliver vi ved mere avancerede versioner af tænkeboks nødt til at støtte os til en algoritme med tilhørende computerprogram til generering af alle de reducerede strategier. I værste fald findes der ikke computere med nok hukommelse til udregning, hvormed man må forsøge at benytte andre metoder, så som at ræsonnere sig frem til fornuftige resultater.

I Kapitel 2 blev der konstrueret et influensdiagram til at finde de bedste modstrategier, givet en strategi. I dette kapitel analyserede vi det simple tænkeboks og fandt udbyttematricen for spillet. Vi indtastede en strategi fra udbyttematricen i influensdiagrammet, evaluerede dette og fandt, at de bedste modstrategier stemte overens med de strategier i udbyttematricen, der gav det bedste udbytte. På denne måde fandt vi en sammenhæng mellem spilteori og influensdiagrammer.

Vi vil i næste kapitel analysere en udvidelse af spillet tænkeboks vha. spilteori i håb om at kunne uddrage generelle tendenser om tænkeboks, når der er flere meldinger.

Kapitel 6

Spilteori på en udvidelse af simpel tænkeboks

I dette kapitel vil vi se på en udvidelse af tænkeboks i forhold til den udgave, vi så på i Kapitel 2. Denne udvidelse vil vi benytte til at få en viden om, hvordan spillet udvikler sig, jo større spillet bliver, og hvordan vi vha. spilteorien kan håndtere dette.

Den udvidede udgave af tænkeboks vi vil se på i dette kapitel, er udvidet med antallet af sider på terningen fra to til tre. Vi ønsker at undersøge den problemstilling, hvor spillet har flere meldinger, således at spillet kan forløbe over flere meldingsrunder. Vi har valgt, at begge spillere fortsat kun skal have én terning hver.

6.1 Strategier

Ligesom ved det simple spil starter vi med at se på strategierne for Arne og Bente. Vi fandt ud af, at strategierne skulle reduceres for at kunne håndterer spillet vha. spilteorien, og med denne udvidelse af spillet er det ikke blevet mindre aktuelt. De reduceringsregler, vi har opstillet, er fremkommet ved at analysere spillet for derved at komme frem til meldinger, der er domineret af andre. Reglerne er følgende:

Start aldrig med at melde $2{\cdot}1$, $2{\cdot}2$ eller $2{\cdot}3$.

Aldrig melde $2{\cdot}1$ eller $2{\cdot}2$, hvis man har en 3'er.
Aldrig melde $2{\cdot}1$ eller $2{\cdot}3$, hvis man har en 2'er.
Aldrig melde $2{\cdot}2$ eller $2{\cdot}3$, hvis man her en 1'er.

Aldrig løfte på 1, hvis man har en 1'er.
Aldrig løfte på 2, hvis man har en 2'er.
Aldrig løfte på 3, hvis man har en 3'er.

Altid løfte på $2{\cdot}1$.
Altid løfte på $2{\cdot}2$.

Det er ikke alle reduceringsreglerne, som er åbenlyse. Den første regel kan virke som en stærk begrænsning, men at melde par vil være uklogt at lægge ud med. Dette skyldes, at sandsynligheden, for at modspilleren har det samme antal øjne, der er blevet meldt par i, kun er en tredjedel. Man taber altså to tredjedele af gangene, når modstanderen også følger de ovenstående reduceringsregler og dermed løfter.

Efter disse reduceringer af strategier vil vi nu se på antallet af tilbageværende strategier, først ser vi på Arnes. Opstillingen af strategier er blevet meget mere kompliceret, da spillet kan vare flere meldingsrunder end før, og derved skal Arne flere gange end før tage stilling til en melding fra Bente. Arne kan i værste fald komme til at skulle tage stilling til en melding tre gange, før der skal løftes. I Tabel 6.1 ses antallet af Arnes mulige meldninger, når han har slået 1.

		Hvis Arne melder			I alt
		1	2	3	
Hvis Bente melder	2	3			
	3	2	2		
	2·1	1	1	1	
	2·2	1	1	1	
Mulige kombinationer		6	2	1	9

Tabel 6.1: Antal mulige strategier for Arne, når han har slået 1. Arne begynder som altid med at melde.

Tretallet i situationen, hvor Arne melder 1, og Bente melder 2, kommer af, at Arne herefter skal tage stilling til, om han skal melde 3, 2·1 eller løfte. Vælger han at melde par, løfter Bente ifølge reglerne. Men melder han 3, kan Bente vælge mellem at melde par eller løfte. Melder Bente par, kan Arne i hans tredje melding kun vælge at løfte. Derfor bliver antallet af mulige kombinationer for Arne de tre mulige tilbagemeldinger, han har i sin anden melding, når Bente melder 2.

Resultatet i Tabel 6.1 bliver det samme, hvis Arne har slået 2 og 3. Da der er ni strategier for hver af Arnes mulige slag, bliver hans samlede antal strategier $9^3 = 729$. Vi ser herefter på Bentes mulige strategier, hvor Tabel 6.2 viser antallet af strategier for Bente.

		Bentes slag			I alt
		1	2	3	
Hvis Arne melder	1	4	5	5	
	2	3	2	3	
	3	2	2	1	
Mulige kombinationer		9	9	9	729

Tabel 6.2: Antal mulige strategier for Bente, hvis hun har slået hhv. 1, 2 eller 3.

Bentes strategier er fundet ud fra hendes slag, samt hvad Arne har meldt. De fire kombinationer i situationen, hvor Bente har slået 1, og Arne har meldt 1, er fundet ved at se på antallet af Bentes mulige tilbagemeldinger. I Bentes første melding kan hun vælge mellem 2, 3 og 2·1. Hvis hun vælger at melde 2, og Arne melder 3 i sin anden melding, har hun i sin anden melding to mulige meldinger at vælge mellem, nemlig 2·1 og løft. Så tilfældet, hvor hun melder 2 i første melding, bidrager med to strategier. Disse to skal så lægges sammen med de to strategier, hvor hun vælger at melde hhv. 3 og 2·1 i sin første melding. Grunden, til at Bente har en strategi mere i situationerne, hvor hun har slået 2 eller 3, og Arne melder 1, er, at hun i sin første melding også har meldingen løft, hvilket hun ifølge reglerne ikke kunne have i den førnævnte situation.

Det vil sige, at vi får en udbyttematrice på 729×729. Med den størrelse af matricen bliver det besværligt og tidskrævende at finde en Nash ligevægt. Men det, der er interessant at finde ud af, er, om der findes en begynderfordel i spillet. Dette kan vi i stedet forsøge at finde ud af vha. en lignende metode, som vi brugte i afsnit 5.1.1.

6.2 Ræsonnering

Vi vil i dette afsnit forsøge at ræsonnere os frem til en god strategi for Bente, som vil resultere i, at hendes forventede udbytte bliver positivt, uanset hvilken strategi Arne vælger. Grunden, til at vi vil se på en strategi for Bente fremfor Arne, er, at Arne oftere end Bente vil have en andenmelding, hvilket gør ræsonneringen lidt mere indviklet. At Arne oftere har en andenmelding skyldes vores reduceringsregler, hvor vi siger, at Arne altid vil begynde med at melde 1, 2 eller 3. Det er kun i det tilfælde, hvor Arne melder 1, Bente 2 og Arne 3, at Bente har nogle interessante andenmeldinger, da Arne ellers melder par som andenmelding, hvorefter Bente blot løfter.

Vi vil gerne finde en strategi for Bente med positivt forventet udbytte. Hvis dette er muligt, kan vi benytte Sætning 4.11 på side 59, som siger, at et endeligt spil mindst har én Nash ligevægt, til at udtale os om spillet har en begynderfordel.

Vi benytter reduceringsreglerne samt vores egne erfaringer med tænkeboks til at finde en "god" strategi for Bente. Den strategi, vi har valgt at se nærmere på, er:

Arne meldte	1			2			3		
Bentes terning viser	1	2	3	1	2	3	1	2	3
Bente melder	2·1	løft	3	løft	2·2	3	løft	løft	2·3

Vi har valgt, at Bentes strategi ikke skal tage højde for Arnes meldinger 2·1, 2·2 eller 2·3, da Arne med de valgte forsimplinger aldrig vil begynde med at melde disse, og derved behøver Bente ikke at tage højde for disse meldinger.

Vi ser nu på, hvad Bente maksimalt kan få i udbytte i alle situationerne. Dette ses i Tabel 6.3, hvor 1 angiver, at Bente vinder, og -1, at hun taber.

Hvis f.eks. Arne melder 1 og Bentes terning viser 1, så vil hun melde 2·1, hvorfor Arne ifølge reduceringsreglerne vil løfte. I dette tilfælde vil Bente vinde, når Arnes terning viser 1, og tabe hvis den viser 2 eller 3. Hvis derimod Arne melder 2, Bentes terning viser 3, så vil hun melde 3. Hvis Arnes terning viser 1 eller 2, så vil han ifølge

Arne meldte		1			2			3		
Bentes terning viser		1	2	3	1	2	3	1	2	3
Bente melder		2·1	løft	3	løft	2·2	3	løft	løft	2·3
Arnes terning viser	1	1	-1	1	1	-1	1	1	1	-1
	2	-1	1	1	-1	1	1	1	1	-1
	3	-1	1	-1	1	-1	-1	-1	-1	1

Tabel 6.3: Bentes maksimale udbytte, hvis hun benytter ovenstående strategi, hvor 1 viser, at Bente vinder, og -1 at hun taber.

reduceringsreglerne melde to ens i det, hans terning viser eller løfte. I begge tilfælde vil Bente vinde. Hvis hans terning derimod viser 3, vil han melde 2·3, hvorved Bente taber. Da Arne har samme udbytte for hver af sine strategier, er Bentes udbytte uafhængigt af Arnes strategi. Hermed ses det af tabellen, at Bente vinder 5 ud af de 9 tilfælde, der skal betragtes givet Arnes strategi. Det vil sige, at Bente altid vil vinde 56% af spillene, hvis hun anvender den viste strategi.

Vi har vist, at Bente har en strategi, hvormed hun vinder mere, end hun taber, ligegyldigt hvad Arne gør. Desuden vides, at spillet ifølge Sætning 4.11 på side 59 mindst har én Nash ligevægt. Kombineres disse to resultater, fås at den eller de Nash ligevægte, der eksisterer, er til fordel for Bente. Grunden til dette er, at vi lige har vist, at Bente, ligegyldigt hvad Arne gør, altid vil vinde i mindst 56% af tilfældene. Mere formelt kan vi vise dette ved en modstrid. Vi antager, at der eksistere en Nash ligevægt, hvor Bente har et udbytte, der er mindre end 56%. Dette kan ikke være en Nash ligevægt, da vi har vist, at Bente har en anden strategi, hvor hun, ligegyldigt hvad Arne melder, har et udbytte på 56%, som hun af denne grund hellere vil benytte. Det vil altså sige, at spillet ikke har en begynderfordel, men derimod er det er fordel at sidde i baghånd.

6.3 Vurdering

I dette kapitel har vi fundet frem til, at simpel tænkeboks, hvor der er to spillere hver med én tresidet terning, ikke har en begynderfordel. Det er derimod en fordel at sidde i baghånd i spillet. Vi har ikke fundet den eller de konkrete Nash ligevægte, da dette viste sig at blive meget tidskrævende. I stedet for koncentrerede vi os om, om spillet har en begynderfordel eller ej, hvilket vi ikke behøvede at finde en Nash ligevægt for at kunne udtale os om.

Ud fra overvejelserne i dette kapitel kan vi desværre ikke konkludere noget om fuld tænkeboks, da det ikke vides, hvordan resultatet om begynderulempen bliver påvirket af ekstra terninger, flere spillere, jokere m.m.

I næste kapitel vil vi netop se på alle delelementerne af tænkeboks, men vi vil gøre det i forbindelse med influensdiagrammer. Dette vil vi gøre for at finde et influensdiagram til at modellere det fulde tænkeboks med.

Kapitel 7

Model for fuld tænkeboks

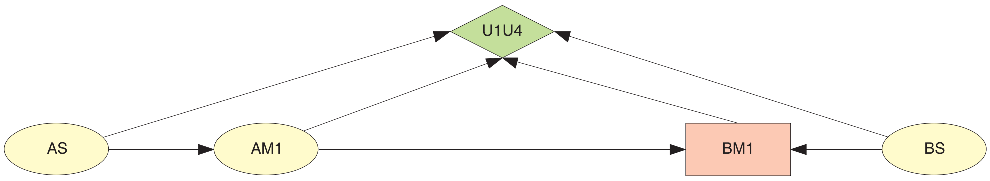
Indtil videre har vi kun beskæftiget os med en simplificeret version af tænkeboks, i dette kapitel ser vi nærmere på den fulde version af spillet.

For at kunne overføre vores simple model til den store version af tænkeboks, ser vi først på delelementerne af udvidelsen. Hernæst forsøger vi at kombinere disse elementer og opnå en model for den fulde version af tænkeboks.

Sidst i kapitlet ser vi på en alternativ tilgangsvinkel til et bayesiansk netværk for tænkeboks.

7.1 Overvejelser af udvidelser af modellen

I det simplificerede tænkeboks kunne Arne og Bente højst melde tre gange hver. Det kunne ses af influensdiagrammet, Figur 2.7 på side 18, at disse meldinger blot var modellen for den første meldingsrunde gentaget tre gange efter hinanden, hvor hver meldingsrunde er forbundet til den næste meldingsrunde vha. af en nyttefunktion mellem Bentes melding, BM_i, og Arnes næste melding, AM_{i+1}. I dette afsnit vælger vi derfor kun at analysere den første meldingsrunde og sammenhængen til næste meldingsrunde, således at hver spiller får lov til at melde én gang hver i første meldingsrunde. Influensdiagrammet for det simple tænkeboks kommer derved til at se ud som på Figur 7.1. Som tidligere ønsker vi også her, at Arne skal have mulighed for at melde 0, når han ikke skal melde først.



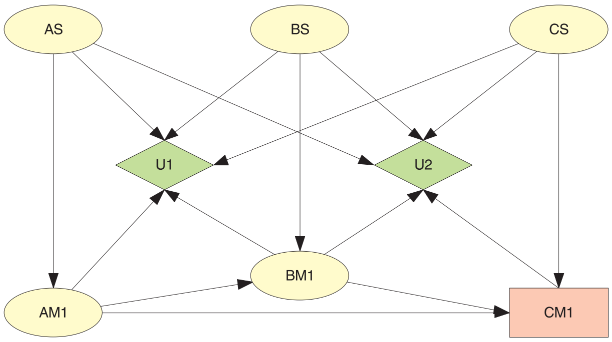
Figur 7.1: Influensdiagrammet for den første meldingsrunde i den simple model fra Kapitel 2.

Dette influensdiagram vil vi nu bruge som baggrund for at undersøge de enkelte elementer af udvidelsen.

7.1.1 Flere spillere

I det simple tænkeboks i Kapitel 2 modelleres spillet, således at der kun er to spillere, Arne og Bente. Vi vil i dette afsnit overveje, hvordan modellen, for det simple tænkeboks, kan udbygges til at håndtere flere spillere. Til illustration af denne problemstilling, vælger vi at udvide den simple model med en spiller og dernæst udlede en generel tendens vha. dette.

På Figur 7.1 kan det ses, at der er en nyttefunktion mellem hver af meldingerne. Denne nyttefunktion sikrer at ugyldige meldinger ikke bliver meldt og angiver, hvem der vinder og taber, når en af spillerne løfter. Skal dette overføres til et influensdiagram for tre spillere fås et diagram som vist i Figur 7.2. Beslutningstageren er spiller C, og hans optimale strategi kan findes ved at løse influensdiagrammet. Den optimale strategi afhænger af spiller A's og B's strategier. Spiller A's strategi kendes ikke af spiller C, men C's bedste "gæt" for A's strategi kan indsættes som sandsynligheden $P(AM_1|AS)$. På tilsvarende vis kan spiller B's strategi indsættes.

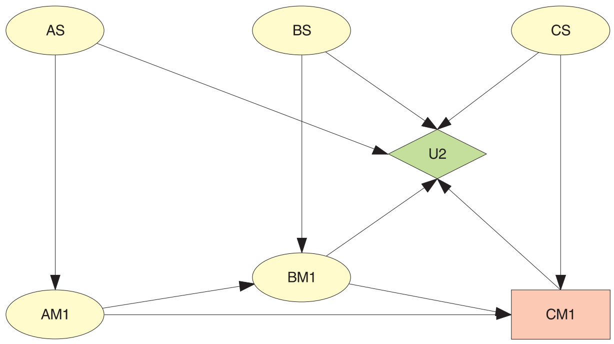


Figur 7.2: Influensdiagrammet for den første meldingsrunde i en model med tre spillere.

Bemærk, at nyttefunktionen mellem AM_1 og BM_1 er overflødig, idet nyttefunktionen kun skal angive, om spillet slutter efter B's melding, dvs. om B løfter. Hvis dette er tilfældet vinder C altid, og nyttefunktionen sættes til 1. Dette kan inkorporeres i U_2 ved at give denne funktion værdien 1 i alle de tilfælde, hvor B løfter. Derfor kan influensdiagrammet reduceres til Figur 7.3.

Udvidelsen med en spiller af det simple tænkeboks har betydning på følgende områder:

Meldingsknuder: Udvidelsen vil have betydning for, hvor mange meldingsknuder der findes, da hver spiller skal foretage en melding i hver meldingsrunde. Yderligere kan det udledes, at strategierne for hver spiller bliver større. Dette kan ses af, at i den simple model uden udvidelse afhænger Arnes anden melding af AS og BM_1. Det vil sige, at antallet af indgange i tabellen $P(AM_2|AS, BM_1)$ er afhængigt af antallet af tilstande i AS, BM_1 og AM_2. Dermed har den $2 \cdot 5 \cdot 4 = 40$ indgange. Antallet af indgange i tabellen for Arnes anden mel-



Figur 7.3: Influensdiagrammet for den første meldingsrunde i en reduceret model med tre spillere.

ding, i den udvidede model, afhænger af tilstandene i AS, BM_1, CM_1 og AM_2. Da der med 0 og løft er 8 meldinger i alt i spillet har den $2 \cdot 8 \cdot 7 \cdot 6 = 672$ indgange.

Nyttefunktion: Ved udvidelse af det simple tænkeboks med en spiller er det ikke nødvendigt med flere nyttefunktioner end den allerede eksisterende, men kompleksiteten af denne vokser, hvilket kan ses af følgende. Den simple models nyttefunktion afhænger af AS, AM_1, BM_1 og BS, dvs. der er $2 \cdot 5 \cdot 5 \cdot 2 = 100$ indgange. Med udvidelsen afhænger nyttefunktionen af AS, BS, CS, CM_1 og BM_1. Nyttefunktionen i den første meldingsrunde har dermed $2 \cdot 2 \cdot 2 \cdot 7 \cdot 8 = 448$ indgange.

Flere meldingsrunder: Når vi udvider modellen med en spiller, medfører det, at vi stadig kun behøver tre meldingsrunder til at simulere spillet. Dette kan ses af følgende meldingsrække:

$$\begin{array}{ccccccccccccccccc} A & & B & & C & & A & & B & & C & & A & & B & & C \\ 0 & \rightarrow & 0 & \rightarrow & 1 & \rightarrow & 2 & \rightarrow & 2 \cdot 1 & \rightarrow & 2 \cdot 2 & \rightarrow & 3 \cdot 1 & \rightarrow & 3 \cdot 2 & \rightarrow & \text{løft} \end{array}$$

Generelt kan vi ud fra følgende overvejelse finde en formel for det maksimale antal af meldingsrunder.

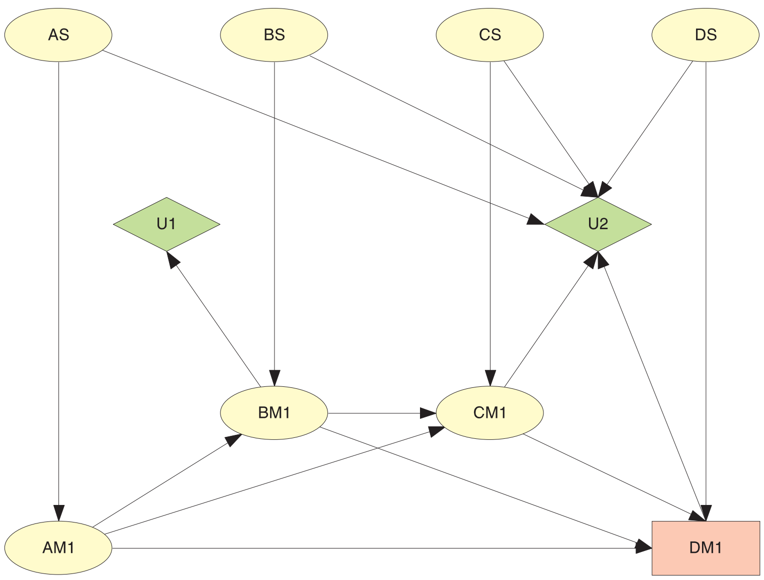
Antag, at vi har en udgave af tænkeboks med n spillere, der i første meldingsrunde hver har t terninger med hver s sider. Med de simple regler, vi har benyttet indtil nu, vil der netop være $nts + 2$ mulige forskellige meldinger. Ved at opstille de mulige meldinger på nær løft og 0 ses, hvorledes produktet i udtrykket er fremkommet, når det samlede antal terninger er nt.

$$\begin{array}{cccc} 1 \cdot 1 & 1 \cdot 2 & \cdots & 1 \cdot s \\ 2 \cdot 1 & 2 \cdot 2 & \cdots & 2 \cdot s \\ \vdots & & & \vdots \\ (\text{nt}) \cdot 1 & (\text{nt}) \cdot 2 & \cdots & (\text{nt}) \cdot s \end{array}$$

Når vi ser på antallet af meldingsrunder, ser vi på den største meldingsfølge, der kan optræde. Dette sker i tilfældet, hvor de første $n-1$ spillere melder 0, og samtlige nts meldinger meldes, før der løftes. Antallet af meldinger bliver således $(n-1)+nts+1 = nts+n$, hvorved det maksimale antal meldingsrunder bliver:

$$\frac{nts+n}{n} = ts+1$$

Hvis der er fire eller flere spillere, er det ligesom i den simple model nødvendigt at have en nyttefunktion umiddelbart før og efter C's melding i diagrammet. Herved kan det i forbindelse med løft undersøges, om C's melding var korrekt eller ej. Endvidere er det nødvendigt at kunne undersøge, om andre spillere løfter, hvorved C vinder. Derfor placeres en nyttefunktion for hver af de andre spilleres meldingsknuder, på nær AM_1, dette er vist på Figur 7.4. Bemærk, at disse nyttefunktioner kun skal have input fra de respektive spilleres meldinger, idet nyttefunktionen kun skal undersøge, om der løftes, og at nyttefunktionen for CM_1, på samme måde som på Figur 7.2 på side 78, er inkorporeret i U_2. C vinder i alle tilfælde, hvor der løftes.



Figur 7.4: Influensdiagrammet for den første meldingsrunde i en reduceret model med fire spillere.

Af overvejelserne kan følgende udledes om tænkeboks med n spillere.

Meldingsknuder: Kompleksiteten af hver meldingsknude øges, idet hver af meldingsknuderne afhænger af, hvad de andre spillere har meldt i den forudgående meldingsrunde. Det vil sige, at hver gang en spiller tilføjes til modellen, vokser antallet af forældre til en meldingsknude med 1. Desuden vil antallet af tilstande i hver meldingsknude vokse, eftersom tilføjelse af en spiller giver flere mulige meldinger i spillet.

Nyttefunktioner: Som det ses af Figur 7.4, er der i hver meldingsrunde i modellen to typer af nyttefunktioner: En, der kun afhænger af en enkelt meldingsknude, og en, der afhænger af to meldingsknuder samt alle spillernes slag. Kompleksiteten af begge typer forøges kraftigt, idet antallet af mulige meldinger vokser. For de nyttefunktioner, der afhænger af spillernes slag gælder, at antallet af indgange maksimalt bliver $2^n \cdot m^2$, hvor n er antallet af spillere, og m er antallet af mulige meldinger. Dette skyldes de n slagvariable, der hver har to tilstande, og de to meldingsvariable, der hver især maksimalt har m tilstande. Antallet af meldinger, m, kan beregnes udfra antallet af spillere. Eftersom hver spiller bidrager med to meldinger, og der i forvejen findes meldingerne 0 og løft, fås $m = 2n + 2$.

Flere meldingsrunder: Det kan konkluderes, at spillet stadigvæk maksimalt vil vare tre meldingsrunder, når simpel tænkeboks udvides med flere spillere. Dette skyldes, at hver ny spiller kun tilføjer to nye meldingsmuligheder samt en ekstra 0 melding.

7.1.2 Flere terninger

I dette afsnit ses på konsekvenserne af at udvide spillet således, at hver spiller har flere terninger. Som udgangspunkt har hver spiller én terning, og vi vil se på, hvad der ændres, hvis hver spiller i stedet havde to terninger. Sidst i afsnittet ser vi på tilfældet, hvor hver spiller har 4 terninger.

Antallet af tilstande i de knuder, der repræsenterer en spillers slag, vokser fra 2 til 3, idet slagkombinationerne for en spiller bliver $11, 12$ og 22. Her dækker 12 f.eks. over tilfældet, hvor en af terningerne viser 1 og den anden viser 2. De samlede slagkombinationer for to spillere bliver, som vist i Tabel 7.1.

Slagkombination	Sandsynlighed
1111	6.25%
1112	25.00%
1122	37.50%
1222	25.00%
2222	6.25%

Tabel 7.1: De mulige slagkombinationer, og deres sandsynligheder, hvor hver af de to spillere har to terninger.

De knuder, der repræsenterer meldinger, vokser fra maksimalt seks tilstande til maksimalt ti tilstande, som vist i Tabel 7.2.

Meldinger med 1 terning	Meldinger med 2 terninger
0, 1, 2, $2 \cdot 1$, $2 \cdot 2$, løft	0, 1, 2, $2 \cdot 1$, $2 \cdot 2$, $3 \cdot 1$, $3 \cdot 2$, $4 \cdot 1$, $4 \cdot 2$, løft

Tabel 7.2: Meldinger for to spillere ved henholdsvis en og to terninger hver.

Nyttefunktionen på Figur 7.1 på side 77, der afhænger af to meldinger og to slag, skal nu tage højde for $9 \cdot 9 \cdot 3 \cdot 3 = 729$ kombinationer af forældretilstande, mod $5 \cdot 5 \cdot 2 \cdot 2 = 100$ kombinationer før udvidelsen.

Hvis vi går til fire terninger til hver spiller, bliver der fem slagkombinationer for hver spiller, disse kombinationer er de samme som i Tabel 7.1, da to spillere med to terninger hver har samme slagkombinationer som én spiller med fire terninger. Antallet af meldinger stiger til 18. Dette medfører, at nyttefunktionen må tage højde for $17 \cdot 17 \cdot 5 \cdot 5 = 7225$ indgange.

Da tilføjelse af ekstra terninger til hver spiller giver flere mulige meldinger, kan spillet nu varer flere meldingsrunder. Derfor skal der tilføjes flere tilstande til både slag- og meldingsvariable.

Tilstandene i en slagvariabel er er de mulige slagkombinationer for en enkelt spillers slag. Antallet af mulige slagkombinationer med t terninger, der hver har s sider, er givet ved:

$$\sum_{i_1=1}^{s} \sum_{i_2=i_1}^{s} \cdots \sum_{i_t=i_{t-1}}^{s} 1 \tag{7.1}$$

Det vil sige, at slagkombinationerne ordnes i stigende rækkefølge, således at ingen kombination er repræsenteret på to forskellige måder. For eksempel repræsenteres 1232 og 2312 begge som 1223.

7.1.3 Flere sider

I dette afsnit ses på konsekvenserne af at udvide spillet, således at terningerne får flere sider.

Tabel 7.3 angiver antallet af mulige meldinger ved spil med 2 spillere, der hver hhv. har en to- eller tresidet terning.

Meldinger ved 2 sider	Meldinger ved 3 sider
0, 1, 2, $2 \cdot 1$, $2 \cdot 2$, løft	0, 1, 2, 3, $2 \cdot 1$, $2 \cdot 2$, $2 \cdot 3$, løft

Tabel 7.3: Meldinger, når terningerne har henholdsvis to og tre sider.

Nyttefunktionen, der afhænger af to meldinger og to slag, vokser derfor fra 100 indgange, se afsnit 7.1.2, til $7 \cdot 7 \cdot 3 \cdot 3 = 441$ indgange.

Udvider vi terningerne til at have seks sider, får vi 14 mulige meldinger og 6 forskellige slagkombinationer for hver spiller. Nyttefunktionen skal derfor i dette tilfælde tage højde for $13 \cdot 13 \cdot 6 \cdot 6 = 6084$ tilstande.

Ud over tilføjelse af ekstra tilstande til de knuder, der repræsenterer slag og meldinger, vil tilføjelse af ekstra sider til terningerne også resultere i tilføjelse af flere knuder til de ekstra meldingsrunder.

7.1.4 Jokere

Vi vil nu undersøge, hvorledes tilføjelse af jokere til regelsættet vil påvirke influensdiagrammet. Vi ser to måder hvorpå jokere kan tilføjes til regelsættet.

Simple jokere

Ved simple jokere forstås en udgave, hvor en etter svarer til et slag på 2, 3, 4, 5 eller 6. På denne måde giver en joker anledning til fem situationer, der skal evalueres, når der løftes.

Da jokeren på denne måde bliver et alias for en af de andre sider på terningen, er det ikke længere muligt med meldinger af formen $x \cdot 1$. Med ovenstående forklaring af variablene n, t og s giver dette, at der i et spil med simple jokere netop er $nt(s-1)+2$ mulige forskellige meldinger. Bemærk, at et spil med simple jokere, hvor terningerne har to sider, ikke giver mening, da meldingen $(tn) \cdot 2$ ellers altid vil optræde.

Haves en allerede eksisterende model, vil indførelse af simple jokere medføre, at en del tilstande i meldingsvariablene skal slettes, hvorved det maksimale antal meldingsrunder, spillet kan vare, reduceres fra $ts+1$ til $t(s-1)+1$. På grund af færre mulige meldinger bliver der færre meldingsrunder i spillet, og derfor skal nogle variable i modellen slettes.

Sletningen af tilstande vil også medføre, at budet for de andre spilleres strategier skal ændres. Indførelse af simple jokere vil ændre sandsynlighederne, for at en melding faktisk kan eksistere, dvs. alle meldinger vil nu blive mere sandsynlige. Spillerne vil derfor være mere tilbøjelig til at byde op end før. Desuden skal nyttefunktionerne, der undersøger, om en melding findes på bordet, opdateres.

Jokerserie

Tilsvarende kunne man også indføre det endnu mere avancerede begreb om jokerserier. Som før giver en etter anledning til fem situationer. Nyt er dog, at hvis den i'te spiller sidder med t_i terninger tilbage og har slået $1, 2, \ldots, t_i$ så tæller slaget for $t_i + 1$ jokere, og giver dermed anledning til yderligere $(t_i + 1)(s - 1)$ situationer, der skal evalueres, når den samlede score på bordet skal afgøres.

I et spil med jokerserier, vil antallet af forskellige meldinger med de tidligere definitioner af n, t og s, være beskrevet ved følgende formel:

$$\text{Antal mulige meldinger} = n(t+1)(s-1)+2 \tag{7.2}$$

Bemærk, at jokerserier også giver mening, hvis $s = 2$. Det resulterende spil går dog kun ud på at gætte, hvor mange toere der er i spillet.

Da der nu er flere mulige meldinger, vil spillet kunne vare et ekstra antal meldingsrunder, hvilket influensdiagrammet skal kunne håndtere. Som før er det derudover kun de andre spilleres strategier og de nyttefunktionerne, der undersøger, om en melding findes på bordet, som skal ændres i modellen.

7.1.5 Melding af ens af et ubestemt antal øjne

En yderligere udvidelsesmulighed er indførelse af meldinger på formen $x \cdot ens$, således at man i meldingen ikke skal lægge sig fast på, hvilken slags ens der forefindes. På den måde er det bedre muligt at skjule information om sit eget slag for modstanderen. Spilles uden jokere, vil tilføjelse af denne meldingstype resultere i spil, der har følgende antal meldinger:

$$\text{Antal mulige meldinger} = nt(s+1)+2. \tag{7.3}$$

7.1.6 Terninger tages fra spillet

I den simple tænkeboksmodel fra Kapitel 2 spilles kun én runde, hvor hver spiller har det samme faste antal terninger. Men i rigtig tænkeboks, jf. afsnit 1.1 gælder det, at når en runde er afsluttet, og taberen er fundet, lægger de resterende spillere én terning fra, indtil kun én spiller har en eller flere terninger tilbage. Dermed er antallet af terninger ikke konstant igennem hele spillet.

Dette giver umiddelbart anledning til forskellige modeller, således at der er en model for hver kombination af antal terninger for hver spiller. Hvis der f.eks. er fire spillere hver med 0, 1, 2, 3, eller 4 terninger, giver dette fem mulige antal terninger for hver spiller, dvs. i alt $5^4 = 625$ modelvarianter. Strukturen for disse modeller vil være den samme, men antallet af variable falder jo færre terninger, der bliver. En grund til at fjerne variable kan være, at en spiller ikke har flere terninger tilbage, hvormed alle knuder, der knytter sig til spilleren, skal fjernes fra influensdiagrammet. En anden grund er, at et færre antal terninger på bordet giver anledning til færre mulige meldinger. Dette betyder, at når en terning fjernes, skal et antal meldingsknuder i sidste meldingsrunde af modellen fjernes. Desuden vil variablene indeholde et færre antal tilstande end før.

I praksis er det dog ikke muligt at have en model for hver mulig situation. En mulig løsning kunne i stedet være en model, der indeholder alle disse mulige kombinationer af terninger. I de modeller vi har betragtet indtil nu har Arnes slag, AS, udelukkende kunnet repræsentere et fast antal terninger, men i stedet kan AS konstrueres, således at der er mulighed for at slå med $0 \leq k \leq t$ antal terninger. Dette giver AS, i tilfældet hvor $t = 4$, følgende tilstande:

$$\text{færdig}, 1, \ldots, 6, 11, \ldots, 66, 111, \ldots, 666, 1111, \ldots, 6666 \tag{7.4}$$

Da antallet af tilstande her er antallet af slagkombinationer med højst t terninger, har vi af udtryk (7.1) på side 82, at antallet af tilstande i en slagvariabel, som f.eks. AS, er givet ved summen:

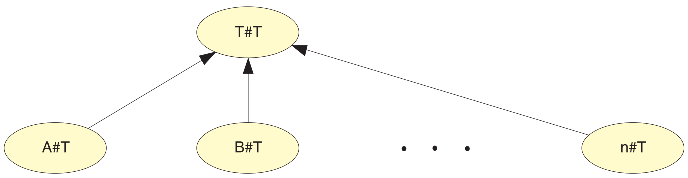
$$1 + \sum_{k=1}^{t} \left(\sum_{i_1=1}^{s} \sum_{i_2=i_1}^{s} \cdots \sum_{i_k=i_{k-1}}^{s} 1 \right) \tag{7.5}$$

Dermed bliver antallet af tilstande i slagvariablen, AS, i tilfælde (7.4) $1 + 6 + 21 + 56 + 126 = 210$, da der er fire terninger hver med 6 sider.

Det er hermed nødvendigt med en informationsvariabel, der angiver, hvor mange terninger, spilleren har tilbage, f.eks. Arnes antal terninger, $A\#T$. Figur 7.5 er et bayesiansk netværk, der viser, hvordan Arnes slagkombinationer afhænger af Arnes antal terninger. Det bayesianske netværk fungerer ved at filtrere slagene i AS, således at når $A\#T = 1$, så er alle tilstande i AS svarende til slag med én terning uniformt fordelt, mens de resterende har sandsynlighed 0. Tilsvarende for $A\#T = 2, 3$ og 4 er alle tilstandene i AS svarende til slag med henholdsvis to, tre og fire terninger tildelt de respektive sandsynligheder for at slagkombinationen optræder i et slag med $A\#T$ terninger. De resterende tilfælde har sandsynlighed 0.

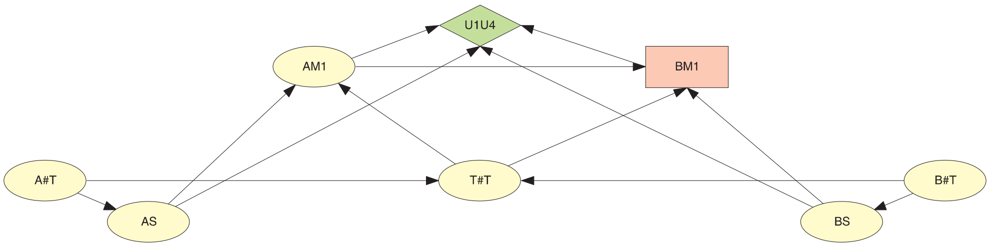
Figur 7.5: Bayesiansk netværk over, hvordan Arnes slag, AS, afhænger af antallet af hans terninger, $A\#T$.

Når en spiller skal til at melde, skal han vide, hvor mange terninger, der i alt er på bordet. Derfor indføres en variabel, $T\#T$, der angiver det totale antal terninger, jf. Figur 7.6. Antallet af tilstande i $T\#T$ vil være $nt+1$, da der teoretisk set kan opstå den situation, hvor alle spillere har 0 terninger.



Figur 7.6: Summering af antallet af terninger vha. variablen $T\#T$.

Figur 7.7 viser et influensdiagram, hvor beslutningstageren er Bente, og hvor Bentes "gæt" for Arnes strategi er indtastet i modellen. Problemet med dette diagram er, at kompleksiteten blot flyttes fra et antal modeller ind i én model. Det vil sige, at vi får én model med endnu større kompleksitet end før. For eksempel har AM_1 nu $210 \cdot (4 \cdot 4 + 1) = 3570$ tabelindgange i et spil med fire spillere, der fra begyndelsen af spillet hver har fire terninger med seks sider. Hvorimod AM_1 før kun havde de 126 tabelindgange, der svarer til antallet af tilstande i AS. Noget tilsvarende gør sig gældende for de andre meldingsvariable.



Figur 7.7: Samlet model for et spil med to spillere, hvor spillerne kan lægge terninger væk efter hver runde.

7.2 Kombination af udvidelserne

Vi vil i dette afsnit forsøge at sammensætte en kombination af de udvidelser, vi overvejede i afsnit 7.1, for at opnå en model for det fulde tænkeboks med fire spillere. Siden vil vi undersøge kompleksiteten af dette.

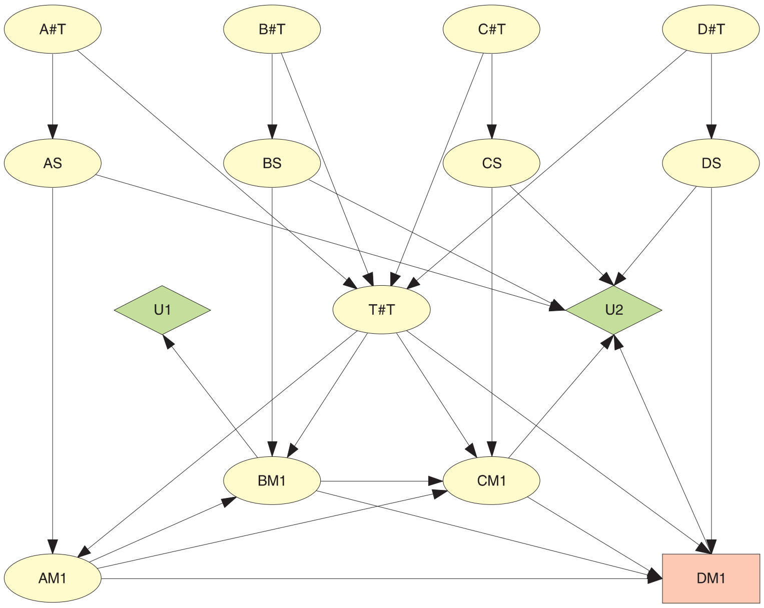
Vi betragter kun første meldingsrunde. Ud fra Figur 7.2 på side 78 og Figur 7.7 kan vi opstille strukturen af influensdiagrammet, som ses på Figur 7.8.

Den stokastiske variabel $T\#T$ i Figur 7.8 er speciel i den forstand, at den i stedet for en sandsynlighed beskriver funktionen:

$$T\#T = A\#T + B\#T + C\#T + D\#T$$

Da DM_1 besluttes på grundlag af $T\#T$ og ikke de enkelte spilleres antal terninger, mister beslutningstageren i influensdiagrammet på figuren information om antallet af terninger hos den enkelte modspiller, således at han ikke kan vurdere sandsynligheden for jokerserier hos den enkelte.

Vi har tilføjet en ny melding til samtlige spilleres meldingsvariable på nær AM_1 og BM_1. Denne melding er "har_løftet", der angiver, at runden er afsluttet, idet en spiller har løftet. Grunden til denne melding er, at vi ellers ville få en forkert forventet nytte, hvilket ses af følgende eksempel. Antag at vi ikke har meldingen "har_løftet", og der løftes i BM_1. I dette tilfælde vinder D, hvorfor den forventede nytte skal være 1. Spørgsmålet er, hvordan modspillernes strategi reagerer, når en tidligere spiller har løftet. Der ses nu på de to mulige tilfælde:



Figur 7.8: Struktur for influensdiagram over første meldingsrunde i fuld tænkeboks.

Der løftes ikke, når en tidligere spiller har løftet: I dette tilfælde vil spilleren foretage en melding, hvorved næste spiller igen vil melde videre, og vi vil i diagrammet få simuleret flere spilforløb. Hvis der f.eks. er to spilforløb, og D vinder første spil, men taber næste, ses den forventede nytte som 0 i stedet for 1.

Der meldes løft, når en tidligere spiller har løftet: Dette vil medføre, at samtlige spillere løfter, hvorved den forventede nytte bliver forhøjet med én for hver af de resterende meldinger fra modspillerne, se afsnit 7.1.1. Dette er ikke noget egentligt problem, hvis D vinder spillet. Men i det tilfælde, hvor det er i AM_2, der meldes løft, og D's melding ikke er opfyldt, vil den forventede nytte blive positiv, selvom D taber runden.

Ved hjælp af meldingen "har_løftet" kan vi få den rigtige forventede nytte fra influensdiagrammet. Dette gøres ved at sætte den forventet nytte for meldingen "har_løftet" til 0 og vælge, at alle spillere vælger "har_løftet", hvis en tidligere spiller har meldt enten "løft" eller "har_løftet". Vi vælger altså f.eks.:

$$P(AM_2|DM_1) = \begin{array}{|c|c|c|} \hline DM_1 \backslash AM_2 & \text{har\_løftet} & \cdots \\ \hline \vdots & 0\% & \cdots \\ \text{løft} & 100\% & \cdots \\ \text{har\_løftet} & 100\% & \cdots \\ \hline \end{array}$$

Vi vil nu se nærmere på kompleksiteten af den opstillede model. Dette vil vi gøre ved først at se på de knuder, der repræsenterer slag, dernæst de, der repræsenterer meldinger, og til sidst nyttefunktionerne.

Slagvariable: Slagvariablene har fået en forældreknude, der angiver, hvor mange terninger spilleren slår med. Denne forældreknude har fem tilstande.

I slagvariablen skal vi nu angive alle mulige slagkombinationer ud fra de forskellige antal af terninger, spilleren slår med. Samlet bliver der, som nævnt i afsnit 7.1.6, 210 forskellige tilstande i knuden.

Meldingsvariable: Kompleksiteten af meldingsknuderne stiger kraftigt i den fulde version af tænkeboks. Vi vælger at analysere CM_1 for at illustrere dette. Det ses af Figur 7.8, at CM_1's forældre er AM_1, BM_1, $T\#T$ og CS. Vi har altså, at antallet af tabelindgange i CM_1 er antallet af tilstandene i forældrene multipliceret med meldingsmulighederne i CM_1. For at finde antallet af tilstande i meldingsvariablene ses på udtrykkene (7.2) på side 84 og (7.3) på side 84. Kombineres disse, fås, at antallet af mulige meldinger ud over "har_løftet" bliver

$$n(t+1)s + 2 = 4 \cdot 5 \cdot 6 + 2 = 122.$$

Det vil sige, at antallet af tilstande i AM_1, BM_1, CM_1 og DM_1 bliver henholdsvis 121, 122, 123 og 122, da der ikke kan meldes "løft" eller "har_løftet" i AM_1, ikke kan meldes "har_løftet" i BM_1 og ikke kan meldes 0 i DM_1.

Antallet af tilstande i T#T blev fastslået til 17 i afsnit 7.1.6, og antallet af tilstande i CS er som nævnt 210. Derved bliver antallet af tabelindgange i CM_1:

$$121 \cdot 122 \cdot 123 \cdot 17 \cdot 210 = 6.5 \cdot 10^9$$

Nyttefunktioner: Den største stigning i kompleksitet ved udvidelse af simpel tænkeboks, forekommer i nyttefunktionerne. Som nævnt i afsnit 7.1.1 på side 78, er der to typer af nyttefunktioner.

Betragt Figur 7.8, det ses, at U_1 har forælderen BM_1, hvorfor antallet af indgange til funktionen bliver 122. Den anden nyttefunktion, U_2, har forældrene AS, BS, CS, DS, CM_1 og DM_1. Antallet af indgange bliver derfor $210^4 \cdot 123 \cdot 122 = 2.9 \cdot 10^{13}$.

Hvis en tabel for U_2 kunne repræsenteres vha. en enkelt byte for hver indgang, ville det kræve 29 terra-bytes!

Foreliggende analyse af den fulde version af tænkeboks tager kun højde for første meldingsrunde. Tager vi antallet af mulige meldinger i betragtning, kan spillet forløbe over 31 meldingsrunder. Dette er dog ikke realistiskt, da dette ville kræve at samtlige meldinger bliver meldt. Vores erfaring fra at spille tænkeboks er, at spillet ikke vil forløbe over mere end to meldingsrunder. Dog må modellen tage højde for tre meldingsrunder, da vi også ønsker at modellere det tilfælde, hvor beslutningstageren skal starte spillet.

Med dette taget i betragtning, indeholder modellen derfor bl.a. 12 meldingsknuder og fem af de store nyttefunktioner. Samlet skal der indtastes ca. $1.5 \cdot 10^{14}$ indgange.

7.2.1 Simplificering af den store model

Da vi må erkende, at den store model har for store krav, både mht. til hardware og indtastningsarbejde, forsøger vi i dette afsnit at forenkle modellen.

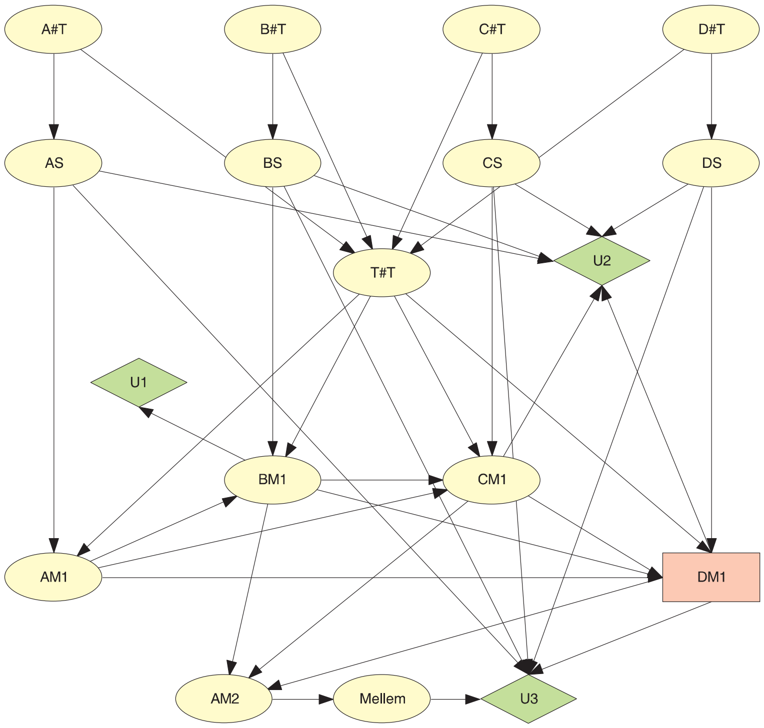
Det mest iøjnefaldne sted at simplificere modellen er i nyttefunktionerne, placeret umiddelbar før og efter beslutningtagerens melding. Begrundelsen for dette er, at den største kompleksitet i hele modellen er placeret der. Hvis det ikke er muligt at få reduceret denne kompleksitet tilstækkeligt, vil det være omsonst at simplificere resten af modellen.

Nyttefunktionen mellem beslutningstagerens melding og næste melding skal afgøre om beslutningstagerens melding er opfyldt, hvis næste spiller løfter. Dermed er vi altså kun interesseret i, om spilleren efter beslutningstageren løfter eller ej, og ikke i, hvad hans melding ellers er. Derfor kan vi indføre en mellemvariabel mellem spillerens melding og nyttefunktionen. Denne mellemvariabel skal kun have to tilstande, der angiver hvorvidt spilleren løftede eller ej, se Figur 7.9. Vi får på denne måde antallet af indgange sænket til $210^4 \cdot 122 \cdot 2 = 4.7 \cdot 10^{11}$. Dette er stadigvæk en meget stor kompleksitet, hvorfor vi må konkludere, at det ville være en fordel, hvis vi kan opstille en alternativ model, der har mindre kompleksitet.

7.3 En alternativ model

Vi har hidtil kun set på modeller, hvor vi har benyttet strategier og modelleret spillets forløb. I dette afsnit vil vil benytte en helt anderledes tilgangsvinkel, hvor det eneste, der interesserer os, er sandsynligheden for, at en melding faktisk findes på bordet. Det vil sige, at vi både er interesseret i situationen, hvor alle spillerne skal melde uden at nogen af dem har set deres slag, og den situation, hvor vi kender vores eget slag, men ikke tillægger de andre spillers melding nogen vægt. Dette modelleres ved at anvende et bayesiansk netværk til udregning af sandsynligheden for hver mulig melding. Ved at aflæse initialsandsynlighederne eller propagere evidens for eget slag får men sandsynligheden for at en melding er opfyldt. Et sådant netværk vil stadigvæk kunne fungerere som et ekspertsystem, men i forhold til influensdiagrammet får brugeren mindre støtte til at træffe sin beslutning: Brugeren skal nu selv vurdere tallene og på baggrund af dem træffe sin beslutning.

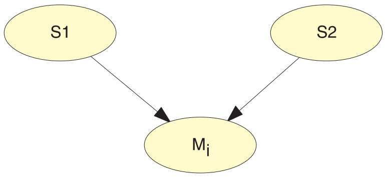
Først betragter vi en simplificering af tænkeboks. Herefter undersøger vi kompleksiteten af denne udgave, og til sidst anvender vi tilsvarende kompleksitetsovervejelser på en fuld tænkeboksmodel med 4 spillere. Alle overvejelserne gennemføres kun for første meldingsrunde, hvor alle spillere endnu har t terninger.



Figur 7.9: Simplificering af kompleksitet i nyttefunktion.

7.3.1 Opbygning af den alternative model i Hugin

I den alternative model tager vi udgangspunkt i et spil, med 2 spillere, der hver har 2 terninger med 3 sider. I spillet er der mulighed for at foretage samtlige meldinger, dvs. der også kan meldes $x \cdot ens$. Ligeledes er der mulighed for at bruge jokerserier. Modellen opbygges, således at hver spillers slag repræsenteres af en slagvariabel, og hver melding bliver til en variabel, der har de to slagvariable som forældre. Figur 7.10 viser det bayesianske netværk for en enkelt melding M_i i denne model. Som udgangspunkt skal netværket kunne give os sandsynligheden for enhver mulig melding i spillet, på nær 0 og løft.

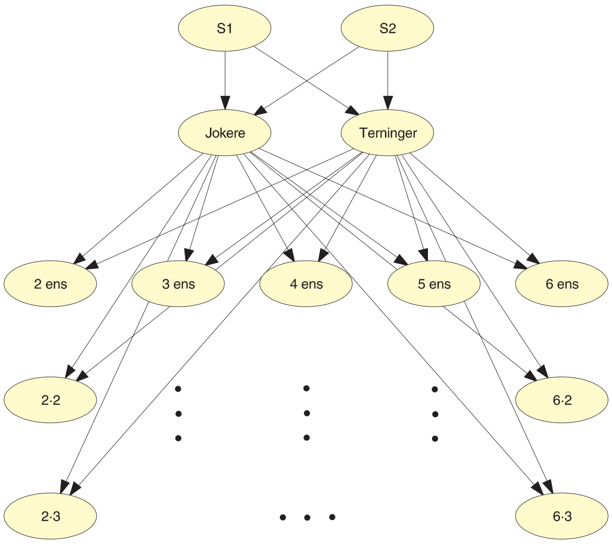


Figur 7.10: Det bayesianske netværk for en enkelt Melding, M_i i den alternative model.

Hver af slagvariablene S_1 og S_2 indeholder initialsandsynlighederne for alle slagkombinationerne. Af udtryk (7.1) på side 82 fås, at hver slagvariabel har seks tilstande. Da visse kombinationer af slag betragtes som ens, vil initialsandsynligheden for eksempelvis 12 være dobbelt så stor som for 11. Dette skyldes, at både 12 og 21 opfattes som slaget 12. En slagvariabel i Figur 7.10 får dermed følgende initialsandsynligheder:

Slag	11	12	13	22	23	33
Initialsandsynlighed	$\frac{1}{9}$	$\frac{2}{9}$	$\frac{2}{9}$	$\frac{1}{9}$	$\frac{2}{9}$	$\frac{1}{9}$

Efter at have illustreret grundidéen i vores alternative model, er det også klart, at kompleksiteten af en enkelt meldingsknude vil blive enorm, når antallet af slagkombinationerne i slagvariablene samt antallet af spillere øges. Vi ønsker derfor en smartere udgave, der vil være i stand til at reducere kompleksiteten, når spillet bliver større. Udgangspunktet vil være, at vi ønsker at opdele terningerne på bordet i jokere og ikke-jokere. Det bayesianske netværk herfor ses på Figur 7.11.



Figur 7.11: Det bayesianske netværk for den alternative model med opdeling af terningerne på bordet i jokere og ikke-jokere.

Da vi ønsker at modellere jokerserie, har vi valgt at have en variabel, der angiver antallet af jokere, der i alt er på bordet. Derudover har vi en variabel, $Terninger$, der fortæller, hvilke kombinationer, der findes på bordet, når alle terninger, der tæller som jokere, er taget fra. Da der her højst kan være 6 jokere, har variablen $Jokere$ følgende syv tilstande:

$$0, 1, 2, 3, 4, 5 \text{ og } 6$$

Disse tilstande angiver antallet af jokere i de to spilleres slag. Da både 1 og 12 tæller som jokere, findes disse ikke blandt de mulige terningkombinationer, der er tilbage, dvs. $Terninger$ har tilstandene:

$$3, 22, 23, 33, 223, 233, 333, 2222, 2223, 2233, 2333, 3333 \text{ og } None$$

Ses på $P(2ens|Jokere, Terninger)$, vil denne have værdien 1 i de tilfælde, hvor $Jokere$ ikke er i tilstand 0, eller hvor $Terninger$ ikke er i tilstand 3 eller 23 og derfor indeholder to ens terningeslag.

I denne udgave af tænkeboks er tilstandene 2 og 222 for variablen $Terninger$ ikke mulige, da én af spillerne dermed skal have slået 12 for at opnå denne tilstand, hvilket tæller for tre jokere. Tilstanden $None$ er medtaget, da der kan opstå situationer, hvor alle terninger er jokere, og dermed er der ikke flere terninger tilbage på bordet, efter at jokerne er fjernet.

Hver af de 18 meldingsvariable indeholder to tilstande, der hhv. fortæller, hvor sandsynlig meldingen er. Hver af disse meldingsvariable afhænger af forældrene, $Jokere$ og $Terninger$. Der er her ikke behov for en egentlig strategi for meldingsvariablene, da det ud fra $Jokere$ og $Terninger$ direkte kan udledes, om en given melding eksisterer på bordet eller ej.

7.3.2 Kompleksitetsberegning

Vi ønsker at gøre os nogle overvejelser om, hvorledes kompleksiteten af denne model til udtræk af rene sandsynligheder afhænger af spillets parametre n,t og s. Antag i det følgende, at spillet har n spillere, der hver har t s-sidede terninger. Først ønskes antallet af tilstande i de enkelte variable bestemt ud fra disse parametre.

Det bayesianske netværk vil indeholde variablerne $S_1, \ldots, S_n$, der hver repræsenter en spillers slag. For et vilkårlig S_i ønsker vi at kunne repræsentere alle mulige slagkombinationer, der kan udføres med t s-sidet terninger, således at rækkefølgen af slagene er underordnet. Fra udtryk (7.1) på side 82 haves, at dette antal er givet ved følgende formel:

$$|S_i| = \sum_{i_1=1}^{s} \sum_{i_2=i_1}^{s} \cdots \sum_{i_t=i_{n-1}}^{s} 1$$

Variablen $Terninger$ skal have tilstande således at den kan repræsentere samtlige mulige kombinationer af terninger, der er tilbage, når alle terningerne, der tæller for jokere er blevet fjernet fra bordet. Antallet af jokere på bordet kan variere fra ingen til samtlige terninger på bordet. Det vil sige, at antallet af jokere, der fjernes, kan variere fra 0 til tn.

I første omgang vil vi nøjes med at fjerne alle ettere fra bordet. Det vil sige, at vi skal have undersøgt alle kombinationer fra 1 til tn terninger, hvor terningens øjne viser fra 2 til s. Dertil adderes det ene tilfælde, hvor alle terninger på bordet var jokere.

Vi har nu følgende formel

$$|Terninger| \leq \sum_{k=1}^{n \cdot t} \left(\sum_{i_1=2}^{s} \sum_{i_2=i_1}^{s} \cdots \sum_{i_k=i_{k-1}}^{s} 1 \right) + 1. \tag{7.6}$$

Egentligt ønsker vi at se bort fra alle terninger på bordet, som resulterer i jokere. Har en af spillerne slået en jokerserie skal et ekstra antal terninger udover etterne derfor fjernes. Da en jokerserie ikke kun består af 1'ere, når $t > 1$, bliver det kompliceret at beregne, hvilke af de ovenstående tilstande der skal fjernes. Vi benytter derfor kun resultatet af (7.6) som en øvre grænse for antallet af tilstande.

Variablen *Jokere* indeholder det præcise antal jokere, der forefindes på bordet. Spillet er med jokerserie, derfor kan hver spiller bidrage med op til $t+1$ jokere, hvorved der maksimalt kan blive $n(t+1)$ jokere i spillet. Da der desuden skal tages højde for situationen, hvor ingen af terningerne på bordet er jokere, fås at

$$|Jokere| = n(t+1) + 1.$$

De ovenstående udtryk for antallet af tilstande i hver variabel i det bayesianske netværk, bruges nu til at give et overslag for størrelsen af tabellen, $P(V_i|pa(V_i))$, der skal angives for hver variabel. Ved at sammenholde med Figur 7.11 på side 91 bestemmes $pa(V_i)$ for hver variabel, hvorefter tabel dimensionen beregnes vha.

$$\dim(V_i) = |V_i| \cdot \left(\prod_{V_j \in pa(V_i)} |V_j| \right).$$

Herved opnås følgende tabeldimensioner for modellen fra Figur 7.11 på side 91.

V_i	$\dim(V_i)$
S_i	$\|S_i\|$
$Terninger$	$\|Terninger\|\|S_i\|^n$
$Jokere$	$\|Jokere\|\|S_i\|^n$
M_i	$2 \cdot \|Terninger\|\|Jokere\|$

Vi har nu udledt de nødvendige formler til beregning af kompleksiteten udfra spillets parametre. Disse formler anvendes i Tabel 7.4 til at give de konkrete tal for spillene $T_1 : (n = 2, t = 2, s = 3)$ og $T_2 : (n = 4, t = 4, s = 6)$.

Antallet af meldingsknuder i netværket for T_1 og T_2 vil være hhv. 18 og 120. Ved at addere tabeldimensionerne for alle variable i et bayesiansk netværk fås det totale antal celler, der skal specificeres for at implementere netværket. Dette totale antal celler for en model vælger vi at betegne $\dim_T$, som er givet ved følgende:

$$\dim_T = \sum_{V_i \in U} dim(V_i),$$

hvor BN er et bayesiansk netværk over universet U.

Spillets Parametre	$(n = 2, t = 2, s = 3)$	$(n = 4, t = 4, s = 6)$
$\lvert S_i \rvert$	6	126
$\lvert Terninger \rvert$	15	20349
$\lvert Jokere \rvert$	7	21
$\lvert M_i \rvert$	2	2
$\dim(S_i)$	6	126
$\dim(Terninger)$	540	$5.129 \cdot 10^{12}$
$\dim(Jokere)$	252	$5.293 \cdot 10^{9}$
$\dim(M_i)$	210	$8.547 \cdot 10^{5}$

Tabel 7.4: Antallet af tilstande og dimensioner af de tilknyttede betingede sandsynligheder i vores alternative netværk. Tallene er givet for spillet i vores eksempel og fuld tænkeboks med fire spillere.

Derved bliver $\dim_T$, for modellerne til spillene T_1 og T_2 hhv. 4584 celler og $5.134 \cdot 10^{12}$ celler.

Havde vi i stedet for benyttet et netværk, som det angivet på Figur 7.10 på side 90, til alle meldinger, ville vi f.eks. for T_2 havde fået $\dim_T = 120 \cdot 126^4 = 6.049 \cdot 10^{10}$. På den baggrund virker vores tilgangsvinkel med $Jokere$ og $Terninger$ ikke særligt overbevisende. $Jokere$ og $Terninger$ formår at reducere kompleksiteten for en enkelt melding, men de store tabeldimensioner flyttes i stedet for hen til de to nye variable.

Styrken ved vores tilgangsvinkel er, at det nu er muligt, ved at indføre mellemvariable, at reducere antallet af forældre til de to variable uden at miste information om de enkelte spillers slag. En tilsvarende simplificering ville ikke kunne lade sig gøre i modellen fra Figur 7.10 på side 90, da informationen om jokerserier ville gå tabt.

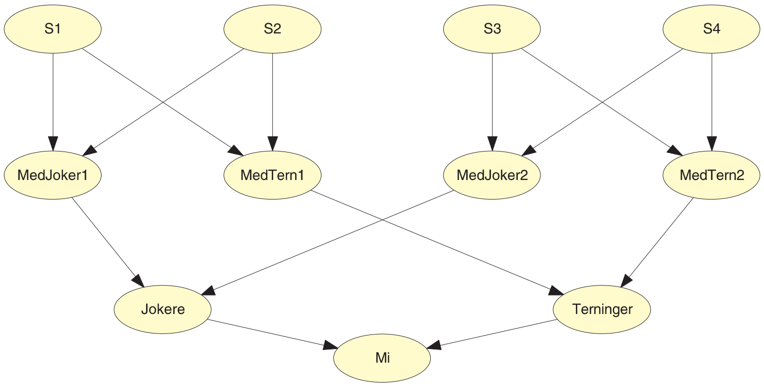
7.3.3 Simplificering af modellen ved hjælp af mellemvariable

Den store kompleksitet i modellen skyldes, at både $Jokere$ og $Terninger$ afhænger af alle slagvariablene. Ved at foretage en reducering i modellen for spillet T_2 vises, hvorledes kompleksiteten generelt kunne reduceres for den alternative model.

De fire slagvariable i modellen for T_2 opdeles i to grupper af hver to, og til hver gruppe indskydes en mellemvariabel for både $Jokere$ og $Terninger$. Ud fra de to mellemvariable for $Jokere$ skal vi udlede det samlede antal jokere, og tilsvarende skal $Terninger$ udlede kombinationen af terninger ud fra de to mellemvariable for $Terninger$, jf. Figur 7.12.

Tabelstørrelserne for variablerne i netværket bliver således som angivet i Tabel 7.5.

Af værdierne i Tabel 7.5 beregnes $\dim_T$ for netværket med mellemvariable for T_2 til $3.385 \cdot 10^{10}$. Det er desværre kun en lille forbedring i forhold til værdien på $6.049 \cdot 10^{10}$ for den primitive model fra Figur 7.10 på side 90. Problemet er, at selv



Figur 7.12: Simplicering af modellen for T_2, således at kompleksiteten på $Jokere$ og $Terninger$ bliver reduceret. Enhver melding, M_i, vil have $Jokere$ og $Terninger$ som forældre.

Spillets Parametre	$(n = 4, t = 4, s = 6)$
$\|S_i\|$	126
$\|MedJoker1\|$	11
$\|MedJoker2\|$	11
$\|Jokere\|$	21
$\|MedTern1\|$	1287
$\|MedTern2\|$	1287
$\|Terninger\|$	20349
$\|M_i\|$	2
$\dim(S_i)$	126
$\dim(MedTern_1)$	$2.043 \cdot 10^7$
$\dim(MedTern_2)$	$2.043 \cdot 10^7$
$\dim(Terninger)$	$3.371 \cdot 10^{10}$
$\dim(MedJoker_1)$	$1.746 \cdot 10^5$
$\dim(MedJoker_2)$	$1.746 \cdot 10^5$
$\dim(Jokere)$	2541
$\dim(M_i)$	$8.584 \cdot 10^5$

Tabel 7.5: Antallet af tilstande og dimensioner for vores sandsynlighedsberegningsmodel for spillet T_2 efter en reducering vha. mellemvariable.

efter reduceringen, er $\dim(Terninger) \gg |S_i|^n$. Det viser sig, at med tilsvarende reduceringer vha. mellemvariable som på Figur 7.12 er det først ved $n \geq 6$, at $\dim(Terninger) < |S_i|^n$.

Med andre ord er er vores opdeling mellem jokere og ikke-jokere først rigtigt værdifuld, når der er 6 eller flere spillere i spillet. På dette tidspunkt er størrelsen af tabellerne alligevel så enorme, at de aldrig ville kunne indtastes alligevel. Den primitive model til beregning af sandsynligheder som i Figur 7.10 på side 90 vil faktisk være bedre egnet til små n.

7.4 Opsummering

Vi har i dette kapitel beskrevet, hvordan vi kan opstille et influensdiagram for fuld tænkeboks med fire spillere. Kompleksiteten af den opstillede model gør det dog umuligt at implementere den. Størstedelen af kompleksiteten kommer fra nyttefunktionerne, der skal indtastes ved at udfylde tabeller. Det ville derfor være at foretrække, hvis det i et program som Hugin var muligt at angive en algoritme til beregning af nytter i nyttefunktioner i stedet for at skulle udfylde alle tabeller eksplicit. Dette ville muligvis bringe opgaven ned på et niveau, hvor det bliver muligt at modellere spillet vha. influensdiagrammer.

Den alternative metode, beskrevet i afsnit 7.3, kan benyttes til at beregne sandsynligheden for de enkelte meldinger. På denne måde bruges Hugin mest som en stor lommeregner. De fundne sandsynligheder skal siden danne grundlag for brugerens beslutning. Selvom modellen ikke giver egentlige strategier, er den af næsten samme kompleksitet som influensdiagrammet. Derfor vil den alternative løsning, ligesom influensdiagrammet, ikke være praktisk gennemførlig for den fulde version af tænkeboks.

7.5 Perspektivering

Som vi så i afsnit 7.2 ligger kompleksiteten i vores influensdiagram i nyttefunktionerne, der afhænger af spillernes slag samt en melding, og kompleksiteten i den alternative model lå i variablen $Terninger$. Vi kunne i influensdiagrammet overveje at benytte opsplitningen af slagvariablene i $Jokere$ og $Terninger$, som vi gjorde i den alternative metode, hvorved vi får kompleksiteten ned på samme niveau som i den alternative model.

Da det for begge modeller er praktisk umulig at modellere fuld tænkeboks, hvor der tages højde for alle mulige meldinger, er det nødvendigt at fravælge nogle af disse. Dette kunne gøres ved kun at se på antallet af ens, der er på bordet, således behøver vi ikke al information om de andre spilleres slag. Vi er mere interesseret i at vide sandsynligheden for det maksimale antal ens, de andre spillere kan bidrage med. Dette kombineret med eget slag, kan så anvendes til at finde sandsynlighederne for de forskellige meldinger af ens. Men denne reducering af kompleksitet sker på bekostning af tabt information, da vi ikke længere får information om hvilke ens der er tale om.

Kapitel 8

Konklusion

I rapporten begyndte vi med at opbygge et bayesiansk netværk for en simpel udgave af tænkeboks, som vi benyttede til at beskrive spillets forløb. Dette bayesianske netværk konverterede vi til et influensdiagram, hvorved vi fandt en bedste modstrategi til modspillerens formodede strategi.

Vi fandt i kapitlet om propagering ud af, at selve propageringen kunne beskrives vha. en algoritme, men at de indledende manøvre med at omdanne det bayesianske netværk til et junction træ, således at kompleksiteten af udregningerne minimeres, kun kunne klares vha. en heuristik. Vi beviste endvidere, at hvis implementationen af Hugin følger den beskrevne Hugin-arkitektur, så svarer tallene, som Hugin leverer, overens med sandsynlighederne for variablene efter tilført evidens.

Efter at have set på den simple udgave af tænkeboks vha. bayesianske netværk og influensdiagrammer, analyserede vi spillet på en teoretisk måde, for at se om spillet havde en begynderfordel. Hertil benyttede vi den matematiske disciplin spilteori på den simple udgave af spillet. Vi fandt ud af, at der var 36 Nash ligevægte, men at de alle havde udbytte på 0. Det var derfor ikke muligt, at nogle af spillerne havde en strategi, hvor de ville vinde mere, end de tabte. Vi fandt også ud af, at selv i det simple spil er det nødvendigt at reducere antallet af strategier med de strategier, som er dominerede af andre. Dette var nødvendigt, da spilteorien ellers ville blive for tung at arbejde med.

For at kunne vurdere, hvorledes spillet udvikler sig ved uvidelsen, benyttede vi spilteori på en udvidelse af den simple udgave, hvor antallet af sider på terningerne var tre i stedet for to. Vi kom her til den konklusion, at det ikke ville kunne svare sig af tidsmæssige årsager at finde en Nash ligevægt. Vi vidste, at den eksisterede, og kunne derfor nøjes med at finde en strategi for Bente, hvor hun ville have et positivt udbytte uanset Arnes valg af strategi. Ud fra dette kunne vi konkludere, at der ikke var en begynderfordel, men at det var den, der sad i baghånd, der havde fordelen.

Da det blev for uoverskueligt at analysere tænkeboks med spilteori, valgte vi, i Kapitel 7, at analysere, hvordan en model for fuld tænkeboks skulle opstilles som et influensdiagram i Hugin. Vi tog udgangspunkt i den allerede opstillede model for det simple tænkeboks og fandt ud af, hvorledes den skulle udvides, for at den kunne modellere et fuldt tænkeboks. Det fremkomne influensdiagram viste sig at være af en sådan kompleksitet, at det ikke er praktisk anvendeligt. Derfor så vi på en alternativ model, der ikke beskæftigede sig med strategier, men kun med sandsynligheder. Denne viste sig imidlertid også at være for kompleks til at være praktisk anvendelig.

8.1 Perspektivering

Vi har fået kendskab til brugen af Hugin og bayesianske netværk. Hugin benytter bl.a. sumpropagering, som vi gik i dybden med, da vi mente, det var et spændende område at få bedre forståelse for. Herudover har vi opnået færdigheder i at modellere problemer vha. influensdiagrammer. Et andet område, vi har arbejdet med, er at analysere spil vha. spilteori. Vi har fundet ud af, at der var store forskelle på tankegangen i bayesianske netværk og spilteori. Ligeledes har de hver især deres fordele og ulemper.

Den væsentligste ulempe for både bayesianske netværk og spilteori er, at de vokser meget i kompleksitet for større modeller. Dette gælder specielt for spilteori, hvor man nogle gange kan blive nødt til at bruge en alternativ tilgang i stedet for systematisk analyse af alle strategier. Til gengæld har spilteori de fordele, at den er meget grundig, og når man først har fundet en Nash ligevægt vha. spilteori, så ved man, at man har en af de bedste strategier. Fordelen ved bayesianske netværk er, at de er meget fleksible i den forstand, at de kan anvendes både som sandsynlighedsberegner og som simulator af spilforløb, samt at man kan konstruere modeller, der modellerer ens egen opfattelse af spillet. Fordelen ved influensdiagrammer er, at den giver en bedste modstrategi, givet en strategi for modstanderen, dog kan det være problematisk at finde en god strategi for modstanderen, da denne strategi altid vil være en estimering. En anden ulempe ved influensdiagrammer er, at kompleksiteten nemt kan risikere at blive meget stor.

Hvis vi havde haft længere tid, ville det næste skridt være at finde en alternativ model, som det i Hugin er muligt at opbygge et netværk for. Dette kunne vi gøre ved at reducere mængden af informationer, vi ønsker, modellen skal kunne give os.

En anden udvidelsesmulighed kunne være at opbygge en model, hvori vi kunne benytte learning, således at strategien kunne forbedres løbende. Vi kunne også lave systemet, således at den genkender sin modspiller, vha. en database over brugere.

Appendiks A

Program til kørsel af modeller i Hugin

For at finde ud af, hvilken af strategierne i de bayesianske netværk fra afsnit 2.5.3, der klarer sig bedst, har vi ladet dem spille mod gættet på Bentes strategi. Dette kunne gøres i Hugin ved at indsætte evidens for computerspillerens slag, for derefter gentagne gange at indsætte evidens for en melding og propagere. Problemet herved er dels, at vi ved en randomiseret strategi ikke kan få Hugin til at vælge en handling og dels, at arbejdet er meget tidskrævende. Løsningen på problemet blev, at vi programmerede et C-program, der vha. Hugin API og C's tilfældighedsgenerator kunne foretage valg af en handling og propagere evidens i modellen.

A.1 Hugin API

Med Hugin API til C har vi de samme muligheder som i Hugin Runtime-programmet, dog med undtagelse af den grafiske brugergrænseflade. Hugin API er altså en samling af funktioner svarende til de funktionaliteter, der er Hugin Runtime-programmet. Fordelen ved at anvende Hugin API er, at vi sammen med C har mulighed for automatisk at afprøve en model flere gange. For eksempel at lade to modeller spille mod hinanden 10.000 gange. Ved hjælp af en tilfældighedsgenerator i C er det også muligt at automatisere processen, hvor et deterministisk valg skal foretages på baggrund af sandsynlighederne i en nondeterministisk strategi. Dette er nødvendigt i den model for tænkeboks, hvor der anvendes nondeterministiske strategier, hvor der ud fra sandsynligheder vælges en deterministisk melding.

Hugin API giver mulighed for at indlæse en model, propagere, initialisere, aflæse sandsynligheder og indsætte evidens i en model. API'en giver også mulighed for at kunne indsætte og fjerne variable og tilstande samt opstille relationer mellem variable i en model. Med andre ord er der de samme muligheder, som der er i Hugin Runtime.

Følgende er en forklaring af de funktioner fra Hugin API, vi bruger i programmet.

`h_domain_t h_load_domain (h_string_t` *file_name*`)` henter en HKB (Hugin Knowledge Base) modelfil fra disken. Modelfilen kan være konstrueret i Hugin Runtime.

`h_status_t h_domain_initialize (h_domain_t` *domain*`)` initialiserer huginmodellen, således at tidligere indsat evidens slettes, og modellen gives initialsandsynlighederne.

`h_string_t h_node_get_name (h_node_t` *node*`)` returnerer navnet på knuden *node*.

`h_node_t h_domain_get_node_by_name (h_domain_t` *domain*`, h_string_t` *name*`)` returnerer knuden med navnet *name* fra huginmodellen *domain*.

`h_count_t h_node_get_number_of_states (h_node_t` *node*`)` returnerer antallet af tilstande i knuden *node*.

`h_string_t h_node_get_state_label(h_node_t` *node*`, h_size_t` *x*`)` returnerer navnet på knuden *node*'s *x*'te tilstand.

`h_number_t h_node_get_belief (h_node_t` *node*`, h_size_t` *state*`)` returnerer "belief" i tilstand *state* i knuden *node*, hvor *state* er indekset på den ønskede tilstand.

`h_status_t h_node_select_state (h_node_t` *node*`, h_size_t` *state*`)` indsætter evidens i tilstand *state* i knuden *node*, hvor *state* er indekset på den ønskede tilstand.

`h_status_t h_domain_propagate (h_domain_t` *d*`, h_equilibrium_t` *eq*`, h_evidence_mode_t` *mode*`)` propagerer huginmodellen *d* med propageringsmåden *eq*, og evidensmode *ev*, f.eks. `h_mode_normal`, hvor evidensen ganges på knuderne som i Sætning 3.22 på side 47.

Se evt. [Hugin Expert A/S, 1997] for yderligere information om Hugin API.

A.2 Programmet

Programmet opbygges, således at der arbejdes med to netværk af gangen. Et spilforløb kan simuleres vha. et enkelt netværk, men ved at arbejde med to opnår vi større fleksibilitet. Dette skyldes, at hvis vi har tre forskellige netværk og/eller strategier med tilhørende HKB filer, kan vi simulere spil imellem et vilkårligt par af disse, uden selv at skulle flette netværkene sammen til et enkelt netværk. De eneste krav vi stiller til HKB filerne er, at netværkene skal indeholde knuderne AS, AM_1, AM_2, AM_3, BS, BM_1, BM_2 og BM_3.

Algoritmen A.1 beskriver funktionaliteten af programmet. Programmet er programmeret til kun at gå et fast antal meldingsrunder svarende til modellerne. Da alle meldingsrunderne er stort set ens, generaliseres dette i algoritmen, uden at tage hensyn til modellens størrelse. Dog er der en undtagelse i første meldingsrunde, hvor model A ikke kan løfte, men det ændrer ikke noget på algoritmen.

Algoritmen benytter funktionsbiblioteket Hugin API, således at Hugin API tager sig af de egentlige interaktioner på de pågældende modeller. Der er i det egentlige program kodet små funktioner til at udføre de trin, der gentages ofte. F.eks. er der funktioner til at slå med terninger og en funktion til at vælge evidens på én tilstand i en variabel ud fra den sandsynlighedsfordeling, der er propageret frem til.

Algoritme A.1

1. Hent de to angivne modeller ind i hukommelsen.
2. Gentag følgende for at udføre det angivne antal spil
   (a) Initialiser begge modeller.
   (b) Slå med terningerner vha. tilfældighedsgeneratoren og opdatér de to modeller i henhold hertil, således at BS i model A (spiller 1) får det første slag, mens BS i model B (spiller 2) får det andet slag.
   (c) Model A gøres opmærksom på, at denne model skal starte, dvs. at variablen $AM1$ gives evidens på 0.
   (d) Herefter propageres begge modeller, så slagene får indflydelse på første melding, og model A er parat til at starte.
   (e) Gentag (Initierende er $x = 1$):
      i. Model A's x'te melding:
         A. Ud fra sandsynlighederne i variabel BM_x i model A vælges der vha. tilfældighedsgeneratoren, hvilken melding model A skal vælge.
         B. Dette opdaters i begge modeller, således at den valgte melding gives som evidens på BM_x i model A og AM_x i model B.
         C. Herefter propageres begge modeller, så den nye evidens udbredes i begge modeller.
         D. Hvis model A valgte løft, hop da til “Find vinderen af spillet”.
      ii. Model B's x'te melding:
         A. Ud fra sandsynlighederne i variabel BM_x i model B vælges der vha. tilfældighedsgeneratoren, hvilken melding model B skal vælge.
         B. Dette opdaters i begge modeller, således at den valgte melding gives som evidens på BM_x i model B og AM_{x+1} i model A.
         C. Herefter propageres begge modeller, så den nye evidens udbredes i begge modeller.
         D. Hvis model B valgte løft, hop da til “Find vinderen af spillet”.
      iii. Tæl x én op.
   (f) Find vinderen af spillet:
      i. Undersøg vha. reglerne for tænkeboks, hvem der vandt spillet.

    ii. Spilstatistikken opdateres, således at den vindende spiller får noteret endnu en sejr.

3. Udskriv spilstatistik på skærmen.

Programmet tager som input parametrenavnene for modellerne, der skal konkurrere, deres filnavne og antallet af spil, de skal spille. Den førstangivne model starter samtlige spil. Herefter rapporterer programmet, hvor mange gange de to modeller vandt ud af det samlede antal spil.

Appendiks B

Program til bestemmelse af udbyttematricen

Dette appendiks indeholder en beskrivelse af design og implementation for C++ programmet, som blev benyttet til at beregne udbyttematricen for de tilbageværende tolv strategier for hhv. Arne og Bente beskrevet i Kapitel 5.

Som beskrevet vil det være muligt at beregne en celle (i, j) i udbyttematricen vha. funktionen $findVinder$.

$$udbytte[i, j] = \sum_{a=1}^{2} \sum_{b=1}^{2} findVinder(A, B, a, b), \tag{B.1}$$

hvor A og B er en strategi for hhv. Arne og Bente.

Programmet skal derfor være i stand til at simulere $findVinder$'s opførsel. Med dette mål i sigte skal en passende datastruktur til repræsentation af en spillers strategi vælges. Herefter kan $findVinder$ simuleres ved følgende pseudokode.

- Givet de to spillers terningeslag, lad dem spille mod hinanden indtil en af spillerne melder løft.
- Når der er løftet undersøg da, om meldingen, der forårsagede løftet, fandtes. Benyt denne information til at tildele de enkelte spillere deres gevinst.

B.1 Datastrukturer

En strategi blev i afsnittet om spilteori defineret til at være en regel, der fortæller spilleren, hvilken handling han skal udføre på ethvert tidspunkt i spillet. Datastrukturen for en strategi tager udgangspunkt i denne definition.

Den enkelte handling for en spiller beskrives i en klasse `Handling`. Klassen indeholder to variable, `mpaaen` og `mpaato`, der fortæller, hvad en spiller skal melde i en given situation, når han har slået 1 eller 2. Problemet er nu at udtrykke begrebet "i en given situation" vha. datastrukturen.

Det udstrakte træ for spillet i Figur 5.1 på side 63 giver en grafisk illustrering af samtlige spilforløb i vores simple tænkeboks. En enkelt instans af `Handling` repræsenterer nu en af spillerens beslutningsknuder i det udstrakte træ. Værdien af variablene `mpaaen` og `mpaato` bestemmer dermed, hvilken gren fra beslutningsknuden spilleren vil vælge i sin strategi.

Herefter vil modspilleren foretage en handling, og hvis han ikke løfter, bliver det den oprindelige spillers tur igen. For at kunne holde styr på, hvilken retning modstanderens handling førte os i det udstrakte træ, indeholder klassen `Handling` referencer, som givet modstanderens valg af melding fører os frem til den næste handling, spilleren skal foretage. Da vi altid skal løfte på meldingen $2 \cdot 2$ indeholder klassen `Handling` dermed tre referencer: `paa1`, `paa2` og `paa2x1`, som hver peger på en ny instans af `Handling`.

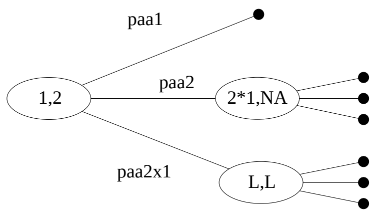
Til ovenstående variable definerer klassen accessor metoder, der bruges til udtræk af næste handling og melding fra klassens variable. Klassen vil i C++ kode blive som følger.

```
class Handling {
public:
  Melding  mpaaen,mpaato;
  Handling *paa1,*paa2,*paa2x1;

  Handling(Melding men, Melding mto,
           Handling *paa1, Handling *paa2,Handling *paa2x1);

  Handling *naesteHandling(Melding sidsteMelding);
  Melding  naesteMelding(int slag);
};
```

Et træ af kædede `Handlinger` kan således beskrive en strategi for spiller S_i. Hver gang S_i skal foretage en beslutning i det udstrakte træ, aflæses næste melding vha. `naesteMelding`. Hver gang modspilleren foretager en melding, flyttes til næste instans af `Handling` vha. metoden `naesteHandling`. For eksemplet vil Arnes 7. strategi fra Tabel 5.8 på side 66 se ud som i Figur B.1



Figur B.1: Repræsentation af en strategi som et træ af instanser af klassen `Handling`. Hver knude indeholder meldinger på formen (`mpaaen`, `mpaato`). Bemærk, hvorledes dette træ ligner det udstrakte træ.

Fordelen ved ovenstående repræsentation er, at den tager udgangspunkt i det udstrakte træ, og at den kan benyttes til repræsentation af både Arne og Bentes strategier.

I et array, `Astrat`, over Arnes strategier vil Arnes 7. strategi kodemæssigt kunne beskrives ved

```
Astrat[ 7] = new Handling(en,to,NULL,
                                new Handling(p1,NA,NULL,NULL,NULL),
                                new Handling(loeft,loeft,NULL,NULL,NULL));
```

Ovenstående kode kan evt. sammenlignes med den grafiske fremstilling i Figur B.1.

B.2 Forløbet af et spil

Efter at datastrukturen for en strategi er blevet fastlagt, skal resultatet af det enkelte spil beregnes. Funktionen `findVinder`, håndterer dette problem. Givet en strategi, `Astrat`, for Arne, en strategi, `Bstrat`, for Bente og spillernes slag, udfører `findVinder` beregningstrinene givet i udtryk (B.1).

```
int findVinder(Handling *Astrat, Handling *Bstrat,int slag[ANTAL]) {
      ⟨Initialisering af variable til kontrol af spilforløbet.⟩
      ⟨Igangsætning af spillet ved at Arne melder.⟩
      ⟨Foretag meldinger indtil en af spillerne løfter.⟩
      ⟨Bestem og returner det korrekte udbytte af det udførte spil.⟩
}
```

Den første komponent af `findVinder` med initialisering af variable sørger for at definere variable, der bruges til at holde styr på spillets forløb.

⟨Initialisering af variable til kontrol af spilforløbet.⟩ ≡

```
      Handling *strategi[ANTAL];
      strategi[A] = Astrat;
      strategi[B] = Bstrat;

      Melding sidsteMelding[ANTAL];
      Spiller sidsteSpiller, naesteSpiller;
      ◊
```

Variablen `sidsteMelding` holder styr på, hvad hver spiller sidst har meldt. Derimod husker `sidsteSpiller` spilleren, der sidst har meldt, og `naesteSpiller` betegner spilleren, hvis tur det er til at melde. De to strategier for Arne og Bente samles i ét array `strategi`.

Spillet sættes nu i gang ved at lade Arne foretage sin første melding. Herved bliver Arne spilleren, der sidst har foretaget en melding.

⟨Igangsætning af spillet ved at Arne melder.⟩ ≡

```
      sidsteMelding[A] = strategi[A]->naesteMelding(slag[A]);
      sidsteSpiller = A;
      ◊
```

Spillerne skiftes nu til at foretage deres melding, indtil en af spillerne melder løft. For hver melding, der foretages, er det nødvendigt at opdatere variablene, som holder styr på spillets gang.

⟨Foretag meldinger indtil en af spillerne løfter.⟩ ≡

```
while (sidsteMelding[sidsteSpiller] != loeft) {
   naesteSpiller = DENEFTER(sidsteSpiller);

   strategi[naesteSpiller] =
     strategi[naesteSpiller]->naesteHandling(sidsteMelding[sidsteSpiller]);

   sidsteMelding[naesteSpiller] =
     strategi[naesteSpiller]->naesteMelding(slag[naesteSpiller]);

   sidsteSpiller = naesteSpiller;
}
```

◊

Bemærk, at `naesteHandling` hele tiden returnerer den gren af træet for strategien, der svarer til grenen, modstanderen vha. sin melding fulgte, i det udstrakte træ.

Når en af spillerne har meldt løft, skal det undersøges, om meldingen inden løftet eksisterede på bordet. Dette sker vha. funktionen `findUdbytte`, der på baggrund af terningerne på bordet, sidste melding før løftet og spilleren, som løftede, afgør, om udbyttet af spillet bliver -1 eller 1. Vi vil i denne oversigt ikke beskrive implementationen af `findUdbytte` nærmere.

⟨Bestem og returner det korrekte udbytte af det udførte spil.⟩ ≡

```
return
  findUdbytte(slag,sidsteMelding[DENFOER(naesteSpiller)],naesteSpiller);
```

◊

Med `findVinder` som redskab er det en let opgave at bestemme den samlede udbyttematrice samt at lokalisere eventuelle rene Nash ligevægte. Programmet er ligeledes i stand til at udskrive udbyttematricen som LaTeX kode. Disse implementationsdetaljer er dog ikke særligt interessante og er derfor ikke medtaget.

Litteratur

[Chartrand og Oellermann, 1993] Gary Chartrand og Ortrud R. Oellermann. *Applied and Algorithmic Graph Theory*. McGraw-Hill, Inc., 1993. ISBN 0-07-112575-2.

[Hugin Expert A/S, 1997] Hugin Expert A/S. *Hugin API 3.1 Reference Manual*, 1997. http://www.hugin.dk/ftp/doc/api-manual.ps.

[Jensen, 1996] Finn Verner Jensen. *An introduction to Bayesain Networks*. UCL Press, 1996. ISBN 1-85728-332-5.

[Jensen, 1997] Finn Verner Jensen. Inference in directed acyclic graphs. *ISSN 0908-1216*, 1997.

[Myerson, 1991] Roger B. Myerson. *Game Theory — Analysis of Conflict*. Harward University Press, 1991.

[Rasmussen, 1989] Eric Rasmussen. *Games and information — an introduction to game theory*. Basil Blackwell, inc., 1989. ISBN 0-631-15709-3.

[Schafer, 1996] Glenn Schafer. *Probabilistic Expert System*. CBMS-NSF Regional conference series in applied mathematics, 1996. ISBN 0-89871-373-0.

Indeks