Lingvistisk steganografi

Steganografi med hjälp av utvidgad synonymitet

A M A R U C U B A G Y L L E N S T E N

KTH Datavetenskap
och kommunikation

Examensarbete
Stockholm, Sverige 2011

Lingvistisk steganografi

Steganografi med hjälp av utvidgad synonymitet

A M A R U C U B A G Y L L E N S T E N

Examensarbete i medieteknik om 15 högskolepoäng
vid Programmet för medieteknik
Kungliga Tekniska Högskolan år 2011
Handledare på CSC var Johan Boye
Examinator var Mads Dam

URL: www.csc.kth.se/utbildning/kandidatexjobb/datateknik/2011/
cuba_gyllensten_amaru_K11042.pdf

Kungliga tekniska högskolan
Skolan för datavetenskap och kommunikation

KTH CSC
100 44 Stockholm

URL: www.kth.se/csc

Sammanfattning

Denna uppsats redogör för och analyserar flera synonymbaserade tillvägagångssätt för lingvistisk steganografi. Vidare kommer uppsatsen redogöra för området lingvistisk steganografi och steganografi i stort för att ge läsaren ett sammanhang. Nya metoder för lingvistisk steganografi baserade på utvidgad synonymitet läggs fram och analyseras. Resultatet är att de två nya metoderna verkar lovande men att mycket arbete krävs för att få dem fungerande och användbara.

Innehåll

Figurer

Kapitel 1

Introduktion

1.1 Bakgrund

1.1.1 Steganografi

> Definition of STEGANOGRAPHY
> 1 archaic : cryptography
> 2 : the art or practice of concealing a message, image, or file within another message, image, or file [1]

Steganografi och kryptografi är väldigt nära förknippade. Båda har som mål och syfte att upprätta ett exklusivt informationsflöde över osäkra kanaler.

Det kryptografiska tillvägagångssättet är att med någon bestämd metod förvränga meddelandet så att den ämnade mottagaren enkelt kan få tillbaka det till den ursprungliga formen, men en oönskad avlyssnare inte kan det. I varje fall inte lika enkelt. Det tillvägagångssättet leder däremot till att krypterade meddelanden är uppenbart krypterade, ett problem i de fall där avlyssnaren har makt och motivering att hindra oönskat (och kanske främst oövervakbart) informationsutbyte.

Steganografin, å andra sidan, utforskar möjligheten att gömma det faktumet att något oönskat informationsutbyte sker. Att gömma meddelanden i små mikroskopiska punkter i text eller bild, skriva med osynligt bläck eller att göra små ändringar i digitala bilder är exempel på mer eller mindre kända och använda metoder för att uppnå detta.

Eftersom somliga länder strävar efter att begränsa användningen och nyttan av kryptografi med diverse metoder[3, 2] blir steganografi och steganalys allt viktigare områden att studera.

1.1.2 Lingvistisk steganografi

Lingvistisk steganografi är, i detta sammanhang, konsten att gömma meddelanden i skrift. Det finns många fördelar med ett sådant tillvägagångssätt.

- Skriven text är i många avseenden väldigt redundant. Antalet olika sätt att representera en mening är nästintill oräknebar, och delmängden av alla vettiga och i kontexten rimliga representationer är stor.

- Skriven text finns överallt, och i stora mängder. Det anses inte vidare ovanligt att som privatperson skriva, skicka och läsa ansenliga textmängder per dag. Sociala medier som facebook, twitter och bloggar har inte lite med saken att göra.

- Skriven text har många egenskaper man, med dagens teknik, inte kan analysera automatiskt. Semantik är ett exempel på sådant.

Det sistnämnda kan tyckas vara till mer problem än nytta. Däremot blir fördelen uppenbar då man betänker att man, för att hitta gömda meddelanden, måste genomsöka all texttrafik. Finns det en automatiserad metod för detta är det bara en fråga om datorkraft. Är man däremot tvungen att anställa människor för att hitta gömda meddelanden i all texttrafik blir uppgiften med ens långt dyrare och svårare. Problem likt dessa, som är svåra för datorer men relativt enkla för människor kallas HIPs (Human Interactive Proofs), CAPTCHAs bygger i princip på samma idé. Komplexitetsmässigt skulle man kunna resonera följaktligen:

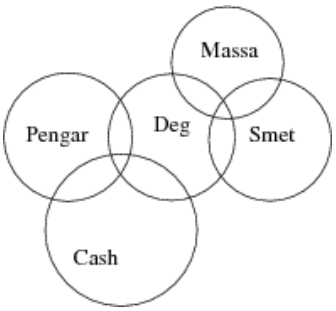
$$\begin{aligned} \mathcal{O}(n) &: \textit{arbete vid dechiffrering} \\ \mathcal{O}(m) &: \textit{den avlyssnade textmängdens storlek} \\ \mathcal{O}(mn) &: \textit{avlyssnarens arbete} \\ \mathcal{O}(n) &: \textit{konspiratörernas arbete} \end{aligned}$$

1.1.3 Problemformulering

Hur kan man gömma meddelanden i elektronisk text med en metod som, utifrån en originaltext, genererar ett steganogram som dels måste dechiffreras med mänsklig hjälp och vara så säkert att det inte enkelt kan klassificeras av en dator som ett steganogram.

1.2 Diverse begrepp

Semantik, syntax och pragmatik Ofta förekommande språkvetenskapliga begrepp. Semantik handlar om symbolernas, oftast ordens, innebörd och mening. Syntax handlar om hur man kan kombinera de enskilda symbolerna till större enheter, såsom hela meningar eller skrifter. Pragmatik handlar om relationen mellan semantiken och syftet.
Meningen “jag har gröna tofflor” betyder rent semantiskt att jaget äger minst ett par gröna tofflor. Vidare är meningen välformulerad rent syntaktiskt. Pragmatiskt däremot kan den, beroende på kontexten i vilken den yttrades, betyda



Figur 1.1. Synonymgrupper till ordet "deg" i ett väldigt begränsat språk

vitt skilda saker. Talaren kanske vill upplysa åhöraren om sitt innehav av gröna tofflor, alternativt vill talaren förtydliga om att dennes tofflor är gröna. Ett tredje, och lite mer långsökt, scenario är att talaren blivit erbjuden gröna tofflor men, eftersom denne redan har ett par och inte önskar fler, artigt tackar nej till erbjudandet.

Synonymgrupp En synonymgrupp är en samling ord som alla är synonyma med varandra. I figur 1.1 är {*deg,pengar,cash*} och {*deg,massa,smet*} båda synonymgrupper. Däremot är inte {*deg,pengar,massa*} en synonymgrupp då *pengar* och massa inte är synonyma. Ett ords synonymgrupper är helt enkelt alla de synonymgrupper det ordet tillhör.

Meddelande, täcktext och steganogram I en steganografisk kontext är meddelandet det du vill skicka utan att en avlyssnare ska veta att du skickar det. Steganogramet är det du till slut skickar, det som döljer meddelandet. En täcktext är, i de fall man använder en, en oskyldig text man vill koda in sitt meddelande i. Kort sagt: Ett steganogram är en täcktext i vilken man kodat in ett meddelande.

AI-komplett En informell term som används för att förklara svåra problem inom AI. Ett AI-komplett problem är ett problem som tycks kräva full mänsklig intelligens för att lösa, ett synnerligen relevant exempel på detta är språkförståelse.

Kapitel 2

Steganografi, teori och formaliseringar

2.1 Formaliseringar

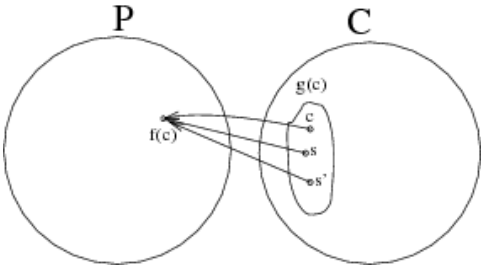
2.1.1 Steganografi med och utan nyckel

I sin grund ser systemet ut som följer: A och B vill kommunicera, men W har tillgång till all information de skickar och tillåter ingen misstänkt kommunikation. Av denna anledning måste kommunikationen mellan A och B vara till synes menlös. Önskar A och B skicka information som inte nödvändigtvis tillhör mängden av alla menlösa meddelande C, utan en annan mängd M, måste en inverterbar funktion $e : M \to C$ användas. Då funktionen är inverterbar är det en enkel sak för B att dekryptera ett steganogram $s = e(m)$ och få fram det ursprungliga meddelandet m då $m = e^{-1}(s)$.

$$\begin{aligned} e &: M \to C \\ e^{-1} &: C \to M \\ e\left(e^{-1}(m)\right) &= m \end{aligned} \tag{2.1}$$

Problem uppstår däremot om W känner till funktionerna e eller e^{-1}. Misstänker W att en text t innehåller ett hemligt meddelande kan denne helt enkelt undersöka om $e^{-1}(t)$ är menlöst eller inte. Lösningen till detta är att man låter en del av metoden vara hemlig, nämligen nyckeln. Idén är att endast A och B känner till nyckeln, som valts ur en mängd möjliga nycklar K. Avlyssnaren W måste i det fallet gissa på alla möjliga nycklar för att få fram meddelandet m i motsats till A och B. Då ser kryptering- och dekrypteringsfunktionerna annorlunda ut.

$$\begin{aligned} e &: M \times K \to C \\ d &: C \times K \to M \\ d(e(m,k),k) &= m \end{aligned} \tag{2.2}$$



Figur 2.1. Mappning till klassmängd P från textmängd C med funktionerna f och g

2.1.2 Alternativ formalisering: Steganografi från meddelandemängd

En väldigt enkel och elegant, men inte lingvistisk, steganografisk metod är att skicka en kortlek. Fransk-engelska kortlekar (de "vanliga") har i regel 52 kort. Det finns således 52! olika sätt att ordna en sådan kortlek på. Kortens ordning kan alltså medvetet väljas av A för att skicka ett specifikt meddelande till B av 52! möjliga. Man kan resonera som så att om A skickar ett specifikt meddelande till B av en mängd möjliga meddelanden, *en mängd som både A och B kan generera utifrån det skickade meddelandet,* så är valet av skickat meddelande informationsbärande. Mer formellt kan man utrycka det på följande sätt:

$$\begin{gathered} f : C \to P \\ g : C \to 2^C \\ g\left(c\right) = \{x \in C | f\left(x\right) = f\left(c\right)\} \end{gathered} \tag{2.3}$$

En visuell representation ses i figur 2.1. Funktionen f representerar någon typ av klassificering. I fallet med kortleken kan man tolka $f\left(c\right) = p$ som att en blandad kortlek c tillhör klassen av kortlekar p ur mängden av alla möjliga klasser P. Det finns trots allt många olika typer av kortlekar (tarot, spanska etc.), och än fler meddelanden kan skickas om A även kan "tappa bort" kort[1]. Funktionen $g\left(c\right)$ tolkas då som mängden av alla möjliga skickade kortlekar som tillhör klass p, g är alltså en sorts invers till f. Kan man sortera denna mängd finns det funktioner $i : C \times 2^C \to \mathbb{N}$ och $j : \mathbb{N} \times 2^C \to C$ där $i\left(c, S\right) = n$ och $j\left(n, S\right) = c$ om och endast om $S_n = c$; eller i data-termer: då kan man skapa en sorterad och indexerad mängd i vilken man, givet ett index kan få fram elementet som hör till det indexet och vice versa. Att B kan dekryptera ett steganogram ges av sambandet $s \in g\left(c\right) \to g\left(c\right) = g\left(s\right)$ enligt ekvation 2.3. Ekvationerna 2.4 och 2.5 förklarar förfarandet utan respektive med nyckel.

[1]Antalet meddelanden blir $\sum_{i=0}^{52} \binom{52}{i}(52-i)! = \sum_{i=0}^{52} \frac{52!}{i!(52-i)!}(52-i)! = 52! \sum_{i=0}^{52} \frac{1}{i!} \approx 52!e$ då $\sum_{i=0}^{\infty} \frac{1}{i!}$ konvergerar mot e väldigt snabbt.

$$
\begin{gathered}
e : \mathbb{N} \times C \to C \\
d : C \to \mathbb{N} \\
e\left(m, c\right) = j\left(m, g\left(c\right)\right) \\
d\left(s\right) = i\left(s, g\left(s\right)\right) \\
d\left(e\left(m, c\right)\right) = m
\end{gathered}
\tag{2.4}
$$

$$
\begin{gathered}
f : C \times K \to P \\
g : C \times K \to 2^C \\
g\left(c, k\right) = \{x \in C | f\left(x, k\right) = f\left(c, k\right)\} \\
e : \mathbb{N} \times C \times K \to C \\
d : C \times K \to \mathbb{N} \\
e\left(m, c, k\right) = j\left(m, g\left(c, k\right)\right) \\
d\left(s, k\right) = i\left(s, g\left(s, k\right)\right) \\
d\left(e\left(m, c, k\right), k\right) = m
\end{gathered}
\tag{2.5}
$$

Den optimala funktionen f skulle mappa alla i sammanhanget k^2 menlösa täcktexter c till en och samma klass $p \in P$, alltså klassen av alla vettiga täcktexter. Tyvärr kan det nog med säkerhet sägas att en funktion g som genererar alla i sammanhanget vettiga texter är, för de naturliga språken, omöjlig att finna i dagsläget och lång tid framöver.

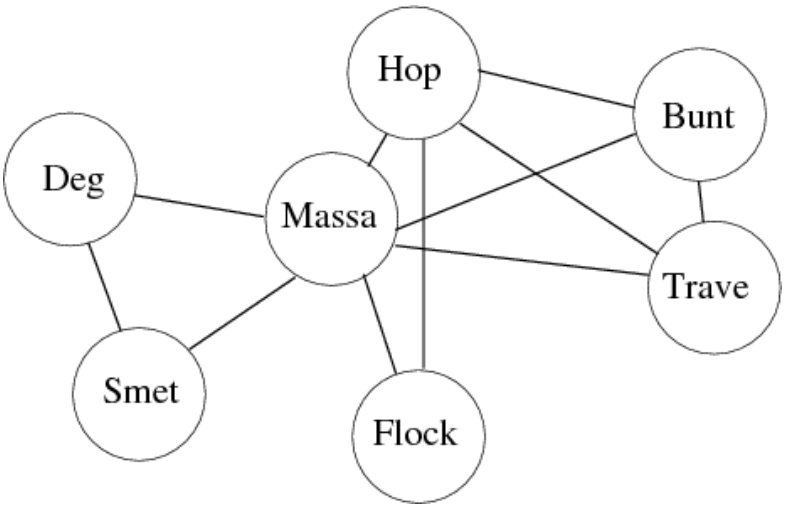
2.2 Synonymsubstitution

Synonymsubstitution är en beprövad och använd metod inom lingvistisk steganografi. Den beskrivs mer ingående i Towards Linguistic Steganography: A systematic investigation of approaches, systems and issues[10].

2.2.1 Metoden i stort

Synonymsubstitution är en metod som enklast förklaras som steganografi från meddelandemängd med nyckel enligt 2.5. I stort kan metoden sägas vara att skapa en meddelandemängd ur alla möjliga val av synonyma ord i steganogramet/täcktexten. Nyckeln i fråga är inte en nyckel i ordets rätta bemärkelse utan snarare en kontext, ett pragmatiskt sammanhang. Människor tycks ha en intuitiv förståelse för sådant, det kan sägas vara vad mänsklig kommunikation bygger på, men det är ett "AI-komplett" problem.

[2]Sammanhanget får alltså ta nyckelns plats. Notera däremot att en uppfattning av ett sammanhang väldigt väl kan liknas vid en nyckel om man antar att W inte är en människa.



Figur 2.2. Synonymgraf

För enkelhetens skulle betraktar vi ett enskilt ord w i en kontext k. $f(w,k)$ kan då sägas vara innebörden av w i kontexten k. $g(w,k)$ blir då mängden av alla möjliga ord som passar in i kontexten k, alltså den synonymgrupp till w som passar in i kontexten k. Valet av ord $s \in g(w,k)$ kan således förmedla $|g(w,k)|$ olika meddelanden. Betraktar man så en text, bestående av orden $w_1, w_2 \cdots w_n$ i kontext k, och tillämpar samma metod på varje enskilt ord, blir mängden möjliga meddelanden $\prod_{i=1}^{n} |g(w_i,k)|$, alltså produkten av varje ords synonymgrupps storlek. Förhoppningen är att A och B ska ha samma begrepp om funktionen f och att k inte ska ändras vid kryptering.

2.2.2 Tillämpning

Mer praktiskt kan synonymsubstitution tillämpas genom att ur en synonymordbok extrahera alla synonymgrupper. Betraktar man synonymordboken som en graf, där hörnen är ord och en kant mellan två hörn innebär att de två orden är synonyma, blir problemet en typ av Klick-problem[3]. En synonymgrupp är helt enkelt en klick, en komplett delgraf. Synonymgrupperna till w är då de klickar som innehåller w. I figur 2.2 är de maximala klickarna som innehåller ordet *massa* {*deg,massa,smet*},{*flock,hop,massa*} och {*bunt,hop,massa,trave*}. Synonymgrupperna till *massa* är således alla delmängder till dessa tre synonymgrupper som innehåller ordet *massa*.

Problem uppstår däremot då funktionen f, alltså klassificeringen av ord, inte delas av A och B; även människor kan ha olika uppfattning om vad som är synonymt i ett visst sammanhang. Ordet trave i meningen *Det låg en trave böcker på bordet* kanske somliga anser är synonymt med hop, massa och bunt medans andra kanske

[3]Ett klickproblem är ett problem som rör de kompletta delgraferna till en graf. En komplett graf är en graf där alla hörn har en kant med alla andra hörn.

anser att hop och massa inte är passande synonymer till trave. Genom att begränsa antalet synonymgrupper och hur de ser ut kan man begränsa denna typ av problem. Ett exempel på en sådan lösning är att endast använda de maximala klickarna som synonymgrupper.

Andra problem är att kontexten k kan komma att ändras vid kryptering. Att de enskilda synonymsubstitutionerna alla är rimliga är ingen garanti för att de, tillsammans, är det. Vidare är det orimligt att ta i beaktning alla möjliga meningar som kan skapas givet en samling synonymgrupper. Detta problem kan lösas genom att endast låta ett fåtal, förhoppningsvis oberoende, ord användas vid kryptering eller genom att låta ett meddelande kodas till flera möjliga steganogram. På så vis kan den som krypterar välja ett steganogram som passar.

2.2.3 Svagheter och problem

Misstänker W att steganografi förekommer finns många olika angreppssätt att tillgå. Det enklaste vore att analysera ordfrekvens och jämföra denna med den vanliga. Det finns däremot metoder för A och B att återskapa en rimlig ordfrekvens. Ett för A och B svårare scenario är om W analyserar orden tillsammans. Den kanske lite underliga meningen *Ensam och rik* kan, givet en viss uppsättning synonymgrupper, kodas om till *Allena och tät.* Sannolikheten för att de två orden *allena* och *tät* ska förekomma i samma mening är troligtvis långt lägre än produkten av deras enskilda sannolikhet. Metoder finns för att utnyttja denna typ av språkliga underligheter[**?**]. Däremot finns det inte, mig veterligen, en enkel metod för A och B att tillgå för att undgå en sådan typ av analys eftersom synonymsubstitutionen endast sker på ordnivå.

Ett sätt att lösa problemet för konspiratörerna kan tyckas vara att utvidga synonymbegreppet till att innefatta även stil, men där uppstår än fler problem. För att skicka mycket information måste synonymgrupperna vara stora. Däremot måste de vara tillräckligt små för att ta hänsyn till stil. För att minska risken för missförstånd mellan A och B bör de också vara få till antalet och så olika varandra som möjligt. Det blir kort sagt en smått omöjlig situation.

Kapitel 3

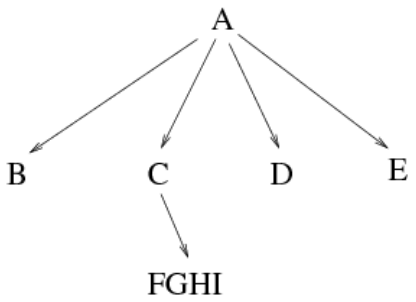
Egen undersökning: Utvidgad synonymsubstitution

Tanken med den utvidgade synonymsubstitutionen är att öka mängden informationseffektiviteten och stärka kravet på mänsklig interaktion genom att låta fler godtyckligheter i språket vara informationsbärande. Som exempel tas här ett stycke ur bibeln:

> *I landet Us fanns en man som hette Job. Det var en oförvitlig och rättrådig man som fruktade Gud och skydde allt ont. Sju söner och tre döttrar hade fötts åt honom. Han ägde 7 000 får, 3 000 kameler, 500 par oxar och 500 åsneston och hade många tjänare. Han var den mäktigaste mannen i hela Östlandet. Hans söner brukade turas om att bjuda hem varandra till fest, och de inbjöd då också sina tre systrar att komma och äta och dricka tillsammans med dem. När alla hade hållit sin fest såg Job till att hans barn blev renade. Tidigt på morgonen offrade han brännoffer för var och en av dem. Han tänkte: "De kan ju ha syndat och i sina tankar förbannat Gud." - Så brukade Job alltid göra.*

Ordningen av dessa meningar är i många avseenden godtycklig. *Sju söner och tre döttrar hade fötts åt honom* kan mycket väl byta plats med *Han ägde 7000 får, 3000 kameler, 500 par oxar och 500 åsneston och hade många tjänare.* Satserna i meningarna har också en i vissa fall godtycklig ordning, på samma sätt som valet av synonyma ord i satserna. Det skulle exempelvis lika gärna kunnat stå *Trenne döttrar och sju söner hade fötts honom* som *Sju söner och tre döttrar hade fötts åt honom.*

Om en för A och B avgörbar delmängd av alla sådana möjliga texter med samma pragmatiska och semantiska innebörd finns har man hittat en funktion g som sedan kan användas vid kryptering enligt metod 2.5. Här redogörs för några sådana metoder som kan användas.



Figur 3.1. Meningsberoendegraf

3.1 Meningspermutation[1]

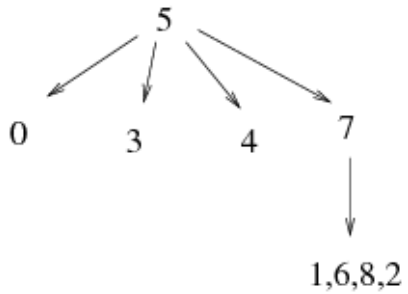
3.1.1 Metod

För att finna alla möjliga synonyma permutationer av ett stycke kan man analysera det som en beroenderelation mellan meningar. Somliga meningar måste stå före andra meningar, vissa meningar kanske måste följa direkt efter varandra o.s.v. Betrakta det tidigare exemplet:

> *A: I landet Us fanns en man som hette Job.*
> *B: Det var en oförvitlig och rättrådig man som fruktade Gud och skydde allt ont.*
> *C: Sju söner och tre döttrar hade fötts åt honom.*
> *D: Han ägde 7 000 får, 3 000 kameler, 500 par oxar och 500 åsneston och hade många tjänare.*
> *E: Han var den mäktigaste mannen i hela Östlandet.*
> *F: Hans söner brukade turas om att bjuda hem varandra till fest, och de inbjöd då också sina tre systrar att komma och äta och dricka tillsammans med dem.*
> *G: När alla hade hållit sin fest såg Job till att hans barn blev renade.*
> *H: Tidigt på morgonen offrade han brännoffer för var och en av dem.*
> *I: Han tänkte: "De kan ju ha syndat och i sina tankar förbannat Gud."*
> *- Så brukade Job alltid göra.*

Mening A måste inte komma efter något, då den inte gör det. Mening B måste komma efter A, om inte vet läsaren inte vem det talas om. Vidare måste meningarna C,D och E också följa A, men inte nödvändigtvis någon annan mening. F måste följa C, G måste följa direkt efter F, H måste följa direkt efter G och I måste följa direkt efter H. Figur 3.1 beskriver hur detta skulle se ut om en graf. Med en sådan

[1]Exakt samma förfarande som för meningar bör gå att applicera på stycken. Informationsmängden är däremot ganska liten per tecken och exempel blir smått gigantiska och otympliga att hantera.



Figur 3.2. Sorterad meningsberoendegraf

beroendestruktur kan man koda in 60^2 olika meddelanden, nästan 6 bitar.

Om de enda relationerna som får uttryckas är *B måste följa A* och *B måste följa direkt efter A*[3], blir alla beroendegrafer riktade acykliska grafer. Att hitta mängden av alla representationer ($g\,(t, k)$ där t är texten och k beroendegrafen) är således ekvivalent med att finna alla topologiska sorteringar av beroendegrafen[4].

Eftersom A och B måste kunna återskapa samma totala ordning i mängden av alla möjliga permuterade meddelanden krävs det en sortering av meningarna i grafen. En sådan ordning kan enkelt skapas, exempelvis alfabetiskt:

> *0:Det var en oförvitlig och rättrådig man som fruktade Gud och skydde allt ont.*
> *1:Hans söner brukade turas om att bjuda hem varandra till fest, och de inbjöd då också sina tre systrar att komma och äta och dricka tillsammans med dem.*
> *2:Han tänkte: "De kan ju ha syndat och i sina tankar förbannat Gud." - Så brukade Job alltid göra.*
> *3:Han var den mäktigaste mannen i hela Östlandet.*
> *4:Han ägde 7 000 får, 3 000 kameler, 500 par oxar och 500 åsneston och hade många tjänare.*
> *5:I landet Us fanns en man som hette Job.*
> *6:När alla hade hållit sin fest såg Job till att hans barn blev renade.*
> *7:Sju söner och tre döttrar hade fötts åt honom.*
> *8:Tidigt på morgonen offrade han brännoffer för var och en av dem.*

Skapar man sedan en graf genom att ständigt lägga till minsta möjliga nod kommer A och B få samma graf; nämligen den som ses i figur 3.2. Den topologiska sorteringen

[2]Då C kommer efter FGHI i hälften av alla 5! möjliga permutationer av {B,C,D,E,FGHI} är antalet möjliga ordningar $\frac{5!}{2}$ som är 60.

[3]Det tidigare representeras av en riktad kant från A till B och det senare av en hopslagning av A och B till AB.

[4]En topologisk sortering är en ordnad mängd element av en riktad acyklisk graf där en kant från a till b betyder att $a < b$. Det finns ett flertal algoritmer för att finna alla sådana sorteringar.

Algorithm 1 Meningspermutation

1. Kryptering
    a) Skriv in stycket c att gömma meddelandet m i.
    b) Låt en människa skapa beroendegrafen k för c.
    c) "Sortera" grafen k.
    d) Skapa steganogramet som ges av den m:te topologiska sorteringen av den sorterade grafen.
2. Dekryptering
    a) Ta emot steganogram s
    b) Låt en människa skapa beroendegrafen k för s.
    c) "Sortera" grafen k. Den sorterade grafen är nu densamma som den sorterade vid kryptering.
    d) Avgör vilken topologisk sortering som ger upphov till s. Dess index är då meddelande m.

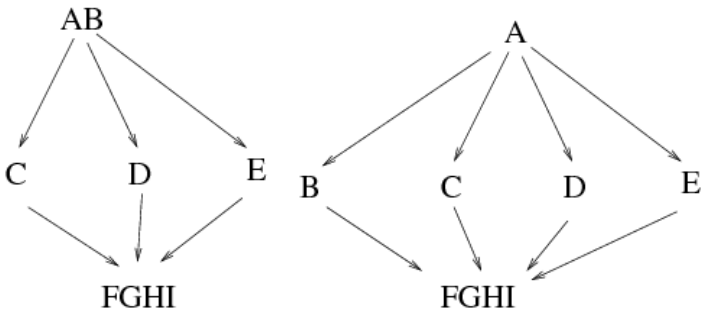
man får om man går från vänster till höger skulle således representeras av följande text:

> *I landet Us fanns en man som hette Job. Det var en oförvitlig och rättrådig man som fruktade Gud och skydde allt ont. Han var den mäktigaste mannen i hela Östlandet. Han ägde 7 000 får, 3 000 kameler, 500 par oxar och 500 åsneston och hade många tjänare. Sju söner och tre döttrar hade fötts åt honom. Hans söner brukade turas om att bjuda hem varandra till fest, och de inbjöd då också sina tre systrar att komma och äta och dricka tillsammans med dem. När alla hade hållit sin fest såg Job till att hans barn blev renade. Tidigt på morgonen offrade han brännoffer för var och en av dem. Han tänkte: "De kan ju ha syndat och i sina tankar förbannat Gud." - Så brukade Job alltid göra.*

Eftersom A och B kan återskapa samma ordnade mängd finns funktioner i, j och g från 2.5 och metoden kan användas för steganografi givet att A och B skapar samma beroendegraf. Förfarandet förklaras i algoritm 3.1.1.

3.1.2 Analys

Jag gjorde en mindre undersökning för att avgöra i hur stor utsträckning människor har en likartad uppfattning om hur beroendegrafen ser ut. Som testtext använde jag det första stycket i Job, alltså samma text jag använder som exempel. Av fyra testpersoner fick två samma graf som den som ses här, de två andra fick däremot



Figur 3.3. De två alternativa beroendegraferna från undersökningen

två annorlunda grafer, dessa syns i figur 3.3. Eftersom hela metoden bygger på att A och B återskapar exakt samma beroendegraf är detta ett mycket allvarligt problem. Att ens beroendegrafen kan representeras som en acyklisk graf är en mycket välbefogad och intressant fråga i sig. Förhoppningsvis kan man det, och även öka samstämmigheten genom träning eller överenskommelse mellan A och B.

En styrka med metoden är att det, i alla fall i dagsläget, inte finns någon metod för att extrahera beroendegrafen för ett stycke. Det är trots allt en uppgift som tycks kräva god semantisk förståelse för att lösa. Jag kan däremot inte med säkerhet säga att uppgiften är "AI-komplett", helt enkelt eftersom det, såvitt jag vet, inte finns några som försökt lösa den. Därför blir det väldigt dyrt för en avlyssnare att finna de texter som är steganogram genom att bara dekryptera alla texter. Det finns däremot närbesläktade problem, som exempelvis pronomenresolution, där språkteknologin har nått viss framgång; dessa kan troligtvis användas för att ta fram rimliga beroendegrafer automatiskt.

Eftersom metoden inte ändrar ordföljd, ordförekomst, meningslängd eller bokstavsdistribution är den säker mot många stilometriska steganalytiska angreppsmetoder. Meningar, som det finns nästintill oändligt många av, är i många avseenden en så pass abstrakt språklig konstruktion att språkteknologisk analys på meningsnivå är nästintill obefintlig.

Den stora svagheten är däremot den minimala mängd information som kan skickas i de flesta fall. Stycken är i regel skrivna som så att meningar måste följa direkt efter varandra.

3.2 Konjunktionspermutation

3.2.1 Metod

Ordningsoberoende konjunktioner som *och, eller* och *samt* kan användas för att koda en bit. *Jag åt ett äpple och drack mjölk* kan lika gärna skrivas om till *Jag drack mjölk och åt ett äpple.* Vid uppräkning av blir totala antalet sätt att ordna den $n!$, som i: *jag köpte mjölk, äpple, smör och ost;* som alltså kan koda 24 olika meddelanden. En ordning kan enkelt etableras, exempelvis alfabetiskt: *jag*

köpte (1:mjölk),(2:ost),(3:smör) och (4:äpple). Permutationerna av denna mening kan sedan skapas så att A och B får samma ordnade mängd av konjunktionspermutationer.

Genom att identifiera alla konjunktioner där delarna kan ändra ordning, och låta varje sådan konjunktion representera en siffra i ett tal med blandad bas[5], där basen är antalet möjliga ordningar, kan man permutera delarna i varje konjunktion för att koda ett visst tal. Tillåter man endast omflyttning i texten, och således inte andra ändringar i ordföljd och annat, ser första stycket i Job ut på följande vis[6]:

> *I landet Us fanns en man som hette Job. Det var en (oförvitlig) och (rättrådig)[2] man som (fruktade Gud) och (skydde allt ont)[2]. (Sju söner) och (tre döttrar)[2] hade fötts åt honom. Han (ägde (7 000 får), (3 000 kameler), (500 par oxar) och (500 åsneston)[24]) och (hade många tjänare)[2]. Han var den mäktigaste mannen i hela Östlandet. Hans söner brukade turas om att bjuda hem varandra till fest, och de inbjöd då också sina tre systrar att komma och (äta) och (dricka)[2] tillsammans med dem. När alla hade hållit sin fest såg Job till att hans barn blev renade. Tidigt på morgonen offrade han brännoffer för (var) och (en)[2] av dem. Han tänkte: "De kan ju ha syndat och i sina tankar förbannat Gud." - Så brukade Job alltid göra.*

Stycket kan alltså koda 1536 olika meddelanden, 10 bitar. Algoritm 3.2.1 förklarar förfarandet i pseudokod.

3.2.2 Analys

Som i meningspermutation gjorde jag här en mindre undersökning för att avgöra i hur stor utsträckning människor kommer fram till samma resultat. Jag bad fyra personer markera vilka satsdelar kring konjunktionerna *och, eller* och *samt* som kunde byta plats fritt runt varje konjunktion. Resultatet var, som i meningspermutation, allt annat än lovande. Av fyra tester fick jag tre olika lösningar, där alla var skilda från min egna. Olikheter uppstod kring *var och en,* då det är ett idiomatiskt uttryck, och *De kan ju ha syndat och i sina tankar förbannat gud* där två förslag på möjliga permutationer fanns: *De kan ju (ha syndat) och (i sina tankar förbannat gud)* och *De kan ju ha (syndat) och (i sina tankar förbannat gud).*

Som i alla metoder som bygger på ett samförstånd mellan A och B om egenskaper i texten så är det ett mycket allvarligt problem om konspiratörerna inte kommer fram till samma egenskaper. I likhet med meningspermutation är förhoppningen här att man kan låta träna konspiratörerna så för att minska chansen att de är av olika åsikter.

---

[5] På engelska kallas det ett sådant talsystem *mixed radix numeral system.* Detaljerna om hur man representerar ett sådant tal är egentligen ointressant och trivialt. Det viktiga är, om $s_0, s_1 \dots s_n$ är talen och b_i är basen för tal s_i, att antalet möjliga tal är $\prod_{i=0}^{n} b_i$, alltså produkten av baserna.

[6] Siffran inom hakparenteser är antalet möjliga ordningar av konjunktionen och inom vanliga parenteser är de satsdelar som kan byta plats kring konjunktionen.

Algorithm 2 Konjunktionspermutation

1. Kryptering
    a) Skriv in stycket c att gömma meddelandet m i.
    b) Identifiera möjliga konjunktionspermutationer kring konjunktionerna *och, eller* och *samt*. Låt p_i vara den i:te permutationsgruppen.
    c) "Sortera" de enskilda permutationerna.
    d) Representera m i basen som ges av de möjliga permutationerna som$m_0, m_1 \ldots$.
    e) Skicka meddelandet där varje konjunktionspermutation p_i representeras av den m_i:te permutationen.
2. Dekryptering
    a) Skriv in stycket c att gömma meddelandet m i.
    b) Identifiera möjliga konjunktionspermutationer kring konjunktionerna *och, eller* och *samt*. Låt p_i vara den i:te permutationsgruppen.
    c) "Sortera" de enskilda permutationerna, A och B bör nu kunna skapa samma ordnade mängd möjliga meddelanden.
    d) Identifiera vilket index varje konjunktionspermutation p_i har och sätt m_i till det indexet.
    e) m är således talet som, med baserna som ges av de möjliga permutationerna, representeras av talet $m_0, m_1 \ldots$

Metoden delar meningspermutationsmetodens styrka i att den varken ändrar ord- eller bokstavsdistribution. Den ordföljdsförändring som sker är i många avseenden obetydande då satsdelarna måste vara av samma typ för att ingå i konjunktionen. Detta gör den säker mot många typer av stilometriska angrepp.

Kapitel 4

Diskussion

4.1 Krav

Innan några av dessa metoder kan användas bör somliga krav uppfyllas.

4.1.1 Samstämmighet

För att konspiratörerna ska kunna kommunicera krävs det att funktionerna f och alltså också g från ekvation 2.4 måste vara identiska. Om inte så kommer B med största sannolikhet inte uppfatta samma meddelande som A skickade. Genom att A och B gemensamt tränar på samma texter för att öka samstämmigheten kan man troligtvis minimera risken för missförstånd. Andra alternativ är att skapa ett program som ger förslag på möjliga alternativ, beroendegrafer eller permutationsgrupper i fallet menings- respektive konjunktionspermutation, och sedan låter en människa markera vilket av alternativen som stämmer.

4.1.2 Analysbarhet

I dagsläget är det svårt att avgöra hur enkelt det är att analysera hur säkra dessa metoder är eftersom jag inte känner till någon forskning i ämnet. Det är däremot inte omöjligt att människor följer vissa mönster i sina konjunktioner som kan analyseras statistiskt eller stilometriskt. På meningsnivå tror jag däremot det blir svårare, just eftersom meningar är en så abstrakt språklig konstruktion korpuslingvistiska metoder inte kan ge större framgång. Sannolikheten att du skriver en mening som ingen annan före dig skrivit är förvånansvärt stor.

4.1.3 Snabbhet

Algortimerna för att kryptera och dechiffrera tar i sitt nuvarande naiva skick $\mathcal{O}(|M|)$ tid. För många användningar är en sådan metod oacceptabel. Jag är däremot övertygad om att det finns algoritmer som kan ta fram en $n : te$ topologisk sortering eller permutation, alltså en funktion i från metod 2.4, som går ansenligt snabbare

än $\mathcal{O}(|M|)$. Finns då en jämförelsefunktion $<$ kan en binärsökning tillämpas för att få fram funktion j, som då tillhör $\mathcal{O}(x \log |M|)$ där $i \in \mathcal{O}(x)$. Det bör å andra sidan gå att finna en funktion j som agerar bättre än så.

4.2 Egenskaper

Metoderna som lagts fram delar vissa egenskaper som skiljer sig från synonymsubstitution. Dessa redogörs för här.

4.2.1 Lokalitet

Förändringarna som görs med meningspermutation och konjunktionspermutation är, givet att de stämmer, lokala. Det är inte svårt att hävda att meningen *smör, mjölk eller ost* är helt synonym med meningen *smör, ost eller mjölk* oberoende av kontexten där *jag var på kalas* inte kan sägas vara helt synonymt med *jag var på fest.* Vidare påverkar inte val av ordning i en konjunktion dess kontext i samma utsträckning som en synonymsubstitution. Samma resonemang gäller för styckes- och meningspermutation. Av denna anledning är det inte, som i synonymsubstitution, en risk att kontexten, som alltså är en typ av nyckel, ska ändras vid kryptering.

4.2.2 Hierarkiskt

Eftersom alla metoder som lagts fram här rör olika abstraktionsnivåer i språket är det möjligt att använda dem samtidigt. Skickar du en text kan den krypteras med styckespermutation, meningspermutation, konjunktionspermutation och synonymsubstitution samtidigt. Det enda som krävs är att varenda del kan ordnas, en ganska enkel uppgift[1]. Antalet meddelanden som då kan skickas är produkten av antalet meddelanden som kan skickas med varje enskild metod. Finns det exempelvis 20 möjliga texter med synonymsubstitution, 24 med konjunktionspermutation och 4 med meningspermutation och 1 med styckespermutation kan man koda 1920 olika meddelanden, en uppenbar förbättring jämfört med att använda en enskild metod.

4.2.3 Stilometrisk analys.

En outforskad metod att finna steganogram som skapats med synonymsubstitution är att använda stilometri. Stilometri är konsten att, givet en text, utröna vem författaren till texten är. Stilometri utgår ifrån antagandet att människor har ett litterärt fingeravtryck, ett personligt sätt att skriva. Om ett sådant fingeravtryck finns kan en avlyssnare W träna en stilometrisk klassifikator på texter genererade med en steganografisk metod, för att sedan identifiera alla den misstänker. Alternativt kan W träna en klassifikator på en mängd misstänkta personer, och om de texter de skickar inte tillskrivs dem har W skäl att tro att de skickar steganogram.

[1] Som förut och alltid hittills i denna text kan den ordningen skapas alfabetiskt. Synonymgrupper ordnas alfabetiskt, konjunktioner likaså etc.

Många metoder finns för att finna ett litterärt fingeravtryck, och de används i regel samtidigt. Särdrag som ofta analyseras är bokstavsfördelning, ordfördelning, meningslängd och, i detta sammanhang särskilt intressant, synonymanvändning[14, 4, 5].

Synonymsubstitution ändrar ett litterärt fingeravtryck baserat på ord-, ordpars-,bokstavs- och synonymförekomst där styckes-, menings- och konjunktionpermutation inte gör det.

4.3 Fortsatt forskning

I fallet meningspermutation är det kanske viktigaste att utröna om en algoritm kan tas fram för att fastställa en beroendegraf. Finns en sådan algoritm som med säkerhet kan skapa en beroendegraf blir metoden långt säkrare för informationsöverföring, men långt osäkrare mot avlyssning. Det återstår också att utröna hur godtycklig meningsordningen egentligen är. Det är mycket möjligt att den följer stilmässiga eller personliga regler som enkelt kan analyseras som i fallet med synonymer. Där val av synonym knappast kan sägas vara helt godtycklig för en individ. Hur godtyckliga konjunktionsordningar är är också en viktig fråga av exakt samma anledning. Om förekomsten av dessa ordningar i stort kan liknas vid brus är de ett väldigt bra val av medium för steganografi, långt bättre än synonymsubstitution, på grund av deras lokalitet.

Vidare bör det vara möjligt att utöka konjunktionspermuterandet till subjunktioner och val av specifik konjunktion. Att skapa en sorts grammatik för konjunktionssubstitution helt enkelt. *Vi gick från festen eftersom det var sent* kan skrivas på många olika, synonyma, sätt: *Vi gick från festen, då det var sent; Eftersom det var sent gick vi från festen; Det var sent, varför vi gick från festen; Det var sent, därför gick vi från festen* etc. Att skapa en grammatik för sådana konjunktionssubstitutioner och sedan låta en människa eller ett program avgöra vilka av de möjliga substitutionerna som är rimliga givet sammanhanget (och alltså realisera en funktion g) bör kunna ge goda resultat inom lingvistisk steganografi.

Kapitel 5

Slutsats

Uppsatsen kanske bristfälliga fokus kan sägas bero på dess ambition: att kort redogöra för lingvistisk steganografi, men även lägga fram nya och välformulerade metoder. Om jag lyckats är det nästan mer än vad jag såhär i efterhand kan hoppas; mycket av det jag lagt fram kräver ansenligt arbete för att få i fungerande skick och kanske än mer för att med någon större korrekthet analysera. Jag känner ändå att jag på något vis bidragit till området lingvistisk steganografi med de nya metoder jag lagt fram, något som kanske var det viktigaste målet med projektet. Finns effektiva metoder för att främja samstämmighet är metoderna lovande.

Eftersom kunskapen om de lingvistiska strukturer som används i de metoder jag lagt fram är ytterst begränsad är det svårt att säga om metoderna uppfyller kravet på att vara AI-kompletta.[1]Det gör å andra sidan området nästan än mer intressant och lockande för vidare forskning.

[1]Däremot kan de användas tillsammans med AI-komplett synonymsubstitution, vilket tillåter en att överföra en nyckel genom AI-kompletta kanaler och koda ett meddelande med tillhörande nyckel i de delar som inte är det. Av denna anledning kan man resonera som så att deras steganalytiska egenskaper är av större vikt än deras AI-svårhet.

Litteraturförteckning

[1] Steganography, definition: http://www.merriam-webster.com/dictionary/steganography

[2] Key disclosure law: http://en.wikipedia.org/w/index.php?title=Key_disclosure_law&oldid=408591

[3] Cryptography laws survey: http://rechten.uvt.nl/koops/cryptolaw/cls2.htm#co

[4] Stylometry: http://en.wikipedia.org/w/index.php?title=Stylometry&oldid=421465732

[5] Document Author Classification using Generalized Discriminant Analysis http://www.siam.org/meetings/sdm06/workproceed/Text%20Mining/moon20.pdf

[6] Topological sorting: http://en.wikipedia.org/w/index.php?title=Topological_sorting&oldid=417795

[7] An algorithm to generate all topological sorting arrangements: http://comjnl.oxfordjournals.org/content/24/1/83.full.pdf+html

[8] On the generation of all topolical sortings: http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6WH3-4D7JMS3-1B&_user=4478132&_coverDate=06%2F30%2F1983&_rdoc=1&_fmt=high&_orig=ga

[9] Natural Language Steganography and an AI-complete Security Primitive: richard.bergmair.eu/pub/towlingsteg-21c3-proc.pdf

[10] Towards Linguistic Steganography: A systematic investigation of approaches, systems and issues: richard.bergmair.eu/pub/towlingsteg-rep-ino-b5.pdf

[11] Linguistic steganography: survey, analysis, and robustness concerns for hiding information in text: www.cerias.purdue.edu/tools_and_resources/bibtex_archive/archive/20013.pdf

[12] Watermarking of natural language texts: http://projects.cerias.purdue.edu/wmnlt/

[13] Lexical steganography: http://alumni.imsa.edu/~keithw/tlex/

[14] A Classifier System for Author Recognition Using Synonym-Based Features: http://www.springerlink.com/content/9644x51751863188/