Benny Brodda
Institutionen för Lingvistik
Stockholms Universitet

81-09-30

"THE TAGGER"

The TAGGER är ett programsystem med vars hjälp man kan skapa och/eller uppdatera taggfiler associerade med en given text på ett semiautomatiskt och interaktivt sätt. The TAGGER är ett för taggningsändamål specialdesignat editeringsprogram där grundideen är att alla inoch utgående filer skall ha enkla, "raka" textformt utan konstiga superstrukturer i form av pekare eller dylikt. I avsnitt 1, nedan, ges en beskrivning av de olika format man har att välja mellan när det gäller taggfilernas utsende.

Den fundamentala frågan (ur teknisk synpunkt) om hur systemet håller reda på associationen mellan tagg och taggat objekt diskuteras i avsnitt 2.

När det gäller själva editeringsförfarandet har The TAGGER ett speciellt kommandospråk, med vars hjälp man enkelt kan ändra eller på annat sätt modifiera sina taggfiler. Detta kommandospråk beskrives i avsnitt 3.

Taggning med hjälp av The TAGGER kan ske på ett semiautomatiskt sätt, genom att man kan foga till ett lexikon till systemet (med regler av typ PÅ => PREP); detta lexikon kan interaktivt ändras under en taggninssession och sedan lagras undan; systemt kan alltså successivt "lära" sig den specifika användarens ideer om hur just hans taggning skall vara. Lexikonets användning och struktur behandlas i avsnitt 4.

Som ovan nämnts så förutsättes att alla input- och outputfiler är i rent textformat (detta gäller även lexikonet). Filerna kan ges godtyckliga namn (dvs olika användare kan använda sina egna filer), men i det följande kommer jag använda följande namn för de olika filerna; dessa namn är också systemets "defaultnamn": Originaltexten, dvs den text som man avser tagga, benämnes INFIL.TXT. Om man redan har taggat texten tidigare men vill ytterligare korr- igera de åsatta taggarna, så utgår systemet från att man har en redan existerande taggfil, OLDFIL.TAG kallad. Den nya (tagg)fil man skapar benämnes NEWFIL.TAG, och har man ett lexikon med "färdiga" taggningsförslag så heter det TAGGER.LEX.

Man bör observera att den ursprungliga originaltexten, INFIL.TXT, förblir alltid oförändrad, det är bara taggfilerna som ändras.

The TAGGER
Benny Brodda
Proceedings of NODALIDA 1981, pages 23-31

Jag vill avslutningsvis i denna inledning betona vad The TAGGER är och vad det inte är. Det är ett editeringsprogram med vars hjälp man bekvämt kan tagga en text, och den taggade filen som därvidlag åstadkommes har en vettig och enkel struktur, användbar, hoppas jag, för många olika syften. Jag vill dock betona att taggning i sig själv inte ger ett vetenskaplig resultat, det är vad man gör med sina taggade texter som (eventuellt) kan ha vetenskaplig värde. Det man gör med The TAGGER är alltså inte forskning i sig. Rätt använt kan dock program av den typ The TAGGER representerar utgöra värdefulla forskningsinstrument.

1. INTAGGNING OCH UTTAGGNING

Med en "tagg" menar jag fortsättningsvis en grammatisk, syntaktisk och/eller semantisk flagga eller etikett (eng. "tag") associerad med ett ord eller annat lingvistiskt objekt i en textmassa av något slag. Denna textmassa kan vara en vanligt löpande text, men den kan också utgöras av en textmassa som på ett eller annat sätt varit utsatt för någon annan typ av processning före själva taggningen, exempelvis en konkordans. "Taggning" är själva processen att åsätta sådana taggar, och The TAGGER är då ett programsystem som är avsett att underlätta denna process.

Som nästan alltid i datasammanhang är själva nyckeln till framgång det att man har genomtänkta och "rena" datastrukturer. I The TAGGER har jag valt att använda textformat i alla in- och utgående filer. Textfiler kan listas, ändras, lagras och flyttas med vanliga enkla standardprogram, och en välstrukturerad textfil kan alltid användas som utgångspunkt för, tex, ett databassystem; omvändningen är inte alltid sant. Vissa begränsningar ligger i en sådan filosofi; vilka jag strax tänker beröra; jag anser ändå att fördelarna med en sådan filosofi är helt överväldigande och jag anser att det skall bra starka skäl till i det enskilda fallet för att överge det enkla texformatet som primärt lagringsformat.

I det vanliga, enkla fallet utgör taggning egentligen inget större problem, nämligen i det fallet att man behöver sätta på enkla grammatiska taggar (el dyl) på de enskilda orden i en löpande text. Antag, t ex, att vi vill sätta på ordklassbeteckningar på orden i följande mening:

(1) KATTEN SITTER PÅ MATTAN.

På ett eller annat sätt vill vi alltså associera (1) med något i stil med

(2) NOM VBPRES PREP NOM

Den datamaskinella frågan vi då måste lösa är hur vi skall lagra taggsekvensen (2), i förhållande till (1), så att maskinen helt entydigt "ser" att det första NOM:et hör ihop

med KATTEN, att VBPRES hör ihop med SITTER, etc. Hur man väljer att lägga sina taggar beror ju delvis på vad man skall göra med sina analys. Kanske vill man göra en ren ordklass-statistik, och då är ju representationen i (2), där taggarna är helt lösgjorda från ursprungstexten, alldeles utmärkt som den är. Man kan betrakta en fil av poster av typ (2) och betrakta den som en text vilken som helst, och på den göra vanlig ordstatistik. Presto.

I andra sammanhang behöver man kanske ha kvar associationen ord-tagg, t ex om man vill kunna fråga sig vilken preposition som är den vanligaste i den text man håller på att analysera. Man behöver då snarare en representation av typ

(3) KATTEN/NOM SITTER/VBPRES PÅ/PREP MATTAN/NOM

Om man i en text, där "orden" ser ut som elementen i (3), gör en enkel konkordans (t ex) så får man just svar på frågor av den sistnämnda typen.

De två nyss angivna sätten att lagra taggarna utgör två av de grundkonventioner för lagring av taggar som The TAGGER utnyttjar. Format (2), där taggarna är lösgjorda från sina resp. ord, kallar jag för "uttaggning", och användes format (3), där taggarna är direkt fasklistrade på sina resp. ord, kallar jag för "intaggning".

Användes uttaggning som lagringsformat, kommer taggarna att existera i en slags "spökfil", parallell med den ursprungliga texten. Denna spökfil blir på sätt och vis en trogen kopia, eller rättare sagt en spegelbild, av ursprungsfilen, därigenom att radfallet bibehålles och att övriga skiljetecken (punkt, komma o dyl) läges på motsvarande ställen. Detta underlättar ganska mycket för läsaren (och delvis också för The TAGGER) när han/hon skall orientera sig i tagg-texten. Jag återkommer strax med en beskrivning över hur The TAGGER mer i detalj klarar av associationen ord-tagg, när uttagning användes.
De nyss angivna lagringsformaten är säkert ganska goda när det gäller att utföra den slutgiltiga analysen, men bägge är definitivt dåliga ur en synpunkt, nämligen ur presentationssynpunkt, antingen medan man håller på att jobba med sin taggning eller för den slutgiltiga presenta- tionen. The TAGGER har då ytterligare ett lagringsformat, som jag kallar (horisontell) display-taggning, som är mycket väl lämpat för sådan presentation:

(4)
```
KATTEN SITTER PÅ   MATTAN
NOM    VBPRES PREP NOM
```

Användes display-taggning så ser The TAGGER till att ord och tagg alltid matchas rakt ovanför varandra vid en utskrift. Detta underlättar korrekturläsning o dyl.

I The TAGGER kan man fritt växla mellan de olika taggnings-konventionerna för indata och utdata. Man kan t ex mata in en

display-taggad OLDFIL.TAG (taggningskonvention 4) och generera en uttaggad NEWFIL.TAG (taggningskonvention 2). På så vis har man inte mindre än 12 olika kombinationsmöjligheter: man kan ha ottaggad, uttaggad, intaggad och displaytaggad fil som input, och man kan generera en uttaggad, intaggad ller displaytaggad outputfil.
Hitinitlls har jag enbart berört enkel ordtaggning, där man hela tiden har att varje ord alltid tilldelas en tagg och att varje tagg alltid hör precis till ett ord. Tyvärr är det inte alltid så enkelt, och platsen här räcker inte för att idetalj gå genom hur man tekniskt kan lösa vidhängande problem. Problemet är, kortfattat, att man inte alltid kan vidmakthålla detta enkla ett-till-ett förhållande mellan ord och tagg. Antag att att man vill ha både syntaktisk och semantisk analys av sin text. Ordet KATTEN, t ex, i (1), kan associeras med en hel uppsättning taggar, av typ NOM+DEF+SUBJ, orden PÅ MATTAN tillsammans kan behövas åsättas beteckningen ADV+PLACE, och att ovanför kan sekvensen SITTER PÅ MATTAN åsättas beteckningen VP. Allt detta gör att vi kan komma att behöva kunna ange många-till-många relationer mellan taggar och ord på ett ganska intrikat sätt.

2. ASSOCIATION ORD-TAGG

I detta avsnitt är det lättast att rent tankemässigt utgå från ett uttaggat system. Som jag antydde tidi gare är själva grundideen i ett sådant system att åstakomma en fil som är så trogen kopia som möjligt av ursprungsfilen vad beträffar det rent yttre; i någon mening skall den "se" lika dan ut. Utgår man från en rättfram ordtaggning, är ju detta väldigt enkelt att åstadkomma, nämligen helt enkelt genom att använda ordliknande beteckningar för sin taggar (PREP, NOM, VERB, o dyl) Gör man så, så åstadkommer The TAGGER verkligen en trogen "spökfil" parallell till den ursprungliga texten.

Denna enkla grundprincip har jag försökt gene ralisera därhän att resultatet ur The TAGGERs synpunkt blir exakt på det sättet, dvs till varje "ord" i ursprungsfilen skall det finnas ett ordliknande objekt i taggfilen och tvärtom. För att åstadkomma detta måste jag dock precisera mer noggrant vad jag menar med ett "ord"; det är genom att generalisera det begreppet som man kan åstakomma mer komplicerade saker än den rena ordtaggningen.

I "vanliga" sammanhang så utgör ett (text)ord helt enkelt en sammanhängande sekvens av bokstäver, på ömse sidor omgiven av ordavkiljare (punkt, komma, mellanslag o dyl). Var och en som sysslat det minsta med datalingvistik är dock helt bekant med det faktumet att i datasammanhang är det inte alltid så klart vad som menas med "bokstav", det hör ju till den datorlingvistiska vardagen att få texter som innehåller de mest mystiska "diakritiska" tecken instoppade i orden (accentmarkering o dyl). Redan detta gör att man i varje generellt system för hantering av texter måste ha mekanismer

att ha ett kommando med innebörden "dagens alfabet utgöres av ...)

I The TAGGER är detta löst på det viset, att varje tecken i maskinens teckenförråd (i den aktuella versionen av The TAGGER ASCII-alfabetet) tilldelas ett typvärde, ett värde i intervallet 0, 1, 2, 3 och 4. "4:or" är det som definieras som "bokstav". (Graf)ord i The TAGGERS mening är då varje maximal, sammanhängande sekvens av "4:or". Grundideen vid uttaggningen (ja, alltid) är att textfilen och taggfilen skall ges exakt samma grafordsstruktur, med bibehållande av alla andra tecken (inkl vagnret.) på sina ursprungliga platser

I The TAGGER kan man ge ett kommando, "DEFTYP 4=" och där räkna upp "dagens alfabet". Det finns ett defaultalfabet i The TAGGER, nämligen

DEFTYP 4= 'A'-'Z', '0-9' (bokstävr o siffror)

dvs varje sammahängande sekvens av bokstäver och/eller siffror (t ex K2R) utgör "ord" i grundsystemet. Vill man betrakta engelska ord av typ CAN'T som ett enda ord så räcker det att föra till "'" till mängden "4:or", alltså med kommandot "DEFTYP 4= '''' (systemt förutsätter att man sätter apostrof kring de tecken man anger, även kring apostrofen).

Anledningen till att siffrorna också är "defaultade" som bokstäver är att det är mycket naturligt att använda taggbeteckningar av typ VB1, VB2, ADV1, ADV2,... för olika underklasser av verb, adverb o dyl. Vill man inte ha siffrorna behöver man bara typa om dem till, t ex, "2:or", vilket är normal värdet för "icke-bokstäver".

Genom att utnyttja denna teckentypning på ett övertänkt sätt kan man nu åstadkomma mycket generella taggningningssystem. Antag att man vill sätta på mer en än tagg på varje enskilt ord, dvs ge den till ordet hörande taggen en inre struktur. Säg att vi vill tilldela ordet KATTEN analysen NOM+DEF+SUBJ+ ANIM. Ja, det kan vi göra precis på det sättet bara genom att typa plustecknet som en 4:a. The TAGGER kommer då att uppfatta hela den nyss angivna sekvensen som ett enda graford, och kan alltså lätt hålla ihop associationen ord-tagg. Varje enskilt tecken i ASCII-alfabetet kan användas som "bokstav" i den här meningen.

Om man vill kunna åsätta hela grupper av ord en skilda taggar blir det genast lite mer kom plicerat, men i princip kan man göra på det sättet att man editerar in paranteser i ursprungsfilen kring de ord man anser utgöra en enhet. Dessa paranteser typas sedan som "4:or", och blir, om de editeras in med luft runt omkring, ur The TAGGERs mening självständiga ord, vilka alltså också blir åtkomliga för taggning. Den således åstadkomna taggningen svarar då precis mot en "labelled bracketing".

3. KOMMANDOSPRÅKET

Själva påförandet eller korrigerandet av taggarna till en text med hjälp av The TAGGER sker med hjälp av ett kommandospråk. Kommandona i detta språk utgöres dels av lokaliseringskommandon, sådana som innebär att man går en fram och tillbaka i den aktuella filen, och dels de rena edite ringskomandona, sådana kommandon som direkt ändrar (eller låter bli) att ändra taggar. I den mån man för på eller ändrar saker så är det uteslutande object (taggar) i taggfilen som ändras, själva ursprungsfilen, INFIL.TXT, påverkas inte av The TAGGER.

Kommandona i The TAGGER har följande syntax

(5) (n) OP (arg)

där n är antalet ggr operationen OP skall utföras, OP är en operator, ett tecken ur den specifika uppsättningen TAGGER-operatorer som systemet ut nyttjar, (f n tecknen /, :, ;, <, =, >, ?, @), och som har det defaultade typvärdet DEFTYP 3. Normalt kan man ge godtyckligt många kommandon på en gång, så länge de ryms inom en rad (om kombi nationskommandon skall användas bör inte typningen av operatorerna ändras). Argumentet <arg> till en operator bör vara ett "graford" enligt avsn. 2. Varje kommando(rad) som innehåller mer än ett tecken totalt lagras automatiskt undan, och kan refereas till genom kommandot "=", vilket kommando alltså innbär "repetera senaste något så när komplexa kommando".

När man kör systemet presenteras posterna en efter en på skärmen i display-format (4), enligt avsnitt 1. Om taggningssessionen innebär en uppdatering av en existerande taggfil (OLDFIL.TAG), så presenteras däri förekommande taggar en efter en, och man kan antingen låta dem förbli som de är, eller ändra dem (med komandot "ändra aktuell tagg"; se nedan). Om taggfilen man håller på att ändra till väsentliga delar är OK kan man hoppa antingen framåt i posten eller ta längre kliv i hela filen med hjälp av lokaliseringskommandona (väsentligen ":").

Om taggningen rör en "fräsch" textfil, dvs en som man inte har någon taggfil till, så presenterar systemet antingen tillverkade taggar (a1, b1, c1, ..., a2, b2,...) vilka fungerar som "dummies" - ur The TAGGERs synpunkt är det enbart viktigt att de rent formellt ser ut som taggar - eller så presenterar systemet förslag till förmodat mer vettiga taggar, enligt det lexikon, TAGGER.LEX, som man optionellt kan ha fogat till systemet. Dessa föreslagna taggar kan man antingen acceptera (kommandot ">"), eller förkasta ("@"), varvid systemet föreslår andra taggar för samma ord, så länge alternativ finnes i lexikonet. Man kan också manuellt editera in det rätta alternativet, om inget av de presenterade dög.

Här följer en kort översikt över de kommandon som f n är implementerade i The TAGGER (version 8110):

OP	ARGUMENT (typ)	BETYDELSE
/	lexikonregel	För in lexikonregeln i TAGGER.LEX
:	ID-begrepp	Lägg ut post, och stega fram till angiven post (förutsätter rad-id)
;	sträng	Ändra akt. tagg till "sträng"
<	tagg	Backa till angiven tagg
=		Repetera senast komplexa instruktion.
>	tagg	Acceptera akt. tagg, och stega ett steg eller till angiven tagg.
?		Displaya akt post en gång till
@		Förkasta föreslagen tagg och begär fram en ny (om sådan finnes)
<CR>		Stega en tagg utan att ändra akt.

Obs! Argumenten är i samtliga fall optionella. Om man vid lokaliseringskommandona inte ger en sträng, "går" systemet ett steg. Ett rent ":", t ex, betyder "Skriv ut akt. post och tag in nästa". Om man vid "/" avstår från argument, sättes en regel in i lexikonet med innebörden "Ge i fortsättningen akt tagg som förslagsalternativ till akt. ord".

Det är många operativa finesser med systemet som jag här inte alls har plats att beröra. Det finns speciella körinstruktioner som tar upp allt det mer i detalj. Låt mig bara nämna en facilitet, som har visat sig mycket användbar, nämligen att man kan köra The TAGGER i vad man kan kalla "run off mode". Antag att man har en "fräsch", otaggad text och ett taggningslexikon. Run off mode betyder att man låter The TAGGER köra igenom hela texten i ett enda svep, varvid den sätter på taggar på de ord den har i sitt lexikon (alltid första alternativet), och låter "dummy"-taggarna stå kvar på övriga ord. Utfilen, som då lämpligen ges display-format, kan man sedan ta ut som en radskrivarutskrift, vilken man i lugn och ro kan sitta hemma och korrekturläsa.

4. LEXIKONET

Jag har ovan upprepade gånger refererat till att man optionellt kan foga ett lexikon till The TAGGER, ett lexikon som systemet utnyttjar till att komma med mer eller mindre vettiga förslag till taggar. Default-namnet på detta lexikon är TAGGER.LEX.

Formatet på lexikonet är att under ordet "REGLER" kommer lexikonreglerna en efter en. Lexikonreglerna skall ha följande utseende:

(6) xxxx; yyy; C

där "xxxx" markerar det ord (el motsvarande) i originalfilen som skall taggas av lexikonet, "yyy" är den tagg man vill

asätta "xxxx", och "C", slutligen, är ett enkelt kontextvillkor, vars betydelse jag strax skall ge.

Samma element "xxxx" kan förekomma flera gånger i lexikonet. Vid en aktuell taggningssituation föreslår systemet taggarna i den ordning man lagt in dem i lexikonet; det är alltså viktigt att försöka Lägga in lexikonorden i rätt ordning.

Exempel: ordet PÅ är säkert i 80 % av alla förekomster en vanlig preposition, och i återstående fall en partikel, och kanske i något enstaka fall något slags adverb (kan vi antaga för exemplets skull). Då bör man i lexikonet skriva

(7) PÅ; PREP; 3
PÅ; PTCL;

och inte tvärtom. Det tredje, lågfrekventa fallet behöver man inte alls ta med, efter som man hela tiden har möjlighet att helt förkasta systemets inbyggda förslag och i stället för hand föra in egna taggar.

Talet "3" som står i exemplet är ett exempel på hur kontextvillkoren skrives, och "3" betyder helt enkelt att det är ord i sin helhet som lexikonet tittar på. Mer specifikt betyder "3" "Ordavgränsare både till vänster och till höger", eller, med andra ord, ord i sin helhet. Övriga tillåtna kontextvillkor är "1", som betyder "Ordavsklijare till vänster", (dvs i ord början) och "2" = "ordavskiljare till höger" (dvs ord slut). Med denna konvention kan man också ha regler av typ:

(8) ARNA; SBPLUR; 2
AR; SBPLUR;
AR; VBPRES
BE; VB; 1
PÅ; VB;

etc. Med den konventionen så skulle ordet POJKARNA få ett förslag SBPLUR, likaså ordet POJKAR. Verbet HOPPAR skulle först ges samma förslag, men efter ett "@" (förkasta förslag) ges man det korrekta VBPRES. Ett ord som SOMMAR finge man tagga för hand, t ex genom kommandot ";SBSG" (ändra föreslagen tagg till SBSG). Kon villkoren ("2" resp. "1" i exemplet) behöver inte upprepas om de är samma.

Ordbörjankontexten "1" kanske är mindre användbar, men i princip kan det ju vara en god gissning att ord som börjar på BE eller PÅ är verb. Sådant får man prova sig fram till.

Jag vill påpeka att man med hjälp av programsystemet BETA kan göra betydligt mer avancerad analys än den man kan åstadkomma inom ramen för The TAGGER, och därför finns det knappast någon anledning att bygga ut själva taggningsprogrammet med än mer sofistikerade analysmekanismer. Observera också att man mer sofistikerade inbyggda analysmekanismer. Med Beta kan man nämligen också se till att utfilen får antingen ett

uttagat eller ett intaggat format, dvs utfilen från Beta kan ges sådant format att den fungerar som input till The TAGGER, i vilket program man sedan manuellt kan göra slutjusteringarna.