2. Utgångspunkter och angreppssätt för automatisk språkgranskning

Detta kapitel beskriver utgångspunkter och angreppssätt för automatisk språkgranskning för svenska. I kapitlet presenteras också språkgranskningsprogrammet Granska, i vilket avhandlingsarbetets lingvistiska delar har implementerats. Jag placerar Granska i en språkteknologisk kontext, för att visa utifrån vilka grunder och beslut programmet konstruerades. Beskrivningen är koncentrerad till den lingvistiska delen av implementationen av Granska. Mer datalogiska och tekniska beskrivningar av Granska finns i (Carlberger & Kann, 1999; Carlberger, Domeij, Kann & Knutsson, 2000; Carlberger & Kann, 2000). En prototyp till Granskas grafiska gränssnitt finns beskriven i (Larsson, 1998).

Det är viktigt att göra klart att det är stora skillnader mellan att bygga ett språkgranskningssystem för svenska och för t.ex. engelska. Den mesta litteraturen om grammatikkontroll handlar om system för engelska. En viktig skillnad är att en del svenska feltyper inte ens förekommer i engelska och tvärtom. För det här arbetets del gäller det särskrivna sammansättningar, inkongruens i predikativ och till stora delar inkongruens i nominalfrasen. Att direkt översätta metoder och analysverktyg från engelska är därför inte möjligt.

Om intresset för grammatikkontroll för engelska de senaste åren har minskat på grund av teknikens kommersiella mognad, har det ökat för flera andra språk, t.ex. holländska (Vosse, 1994), spanska och grekiska (Bustamente & León, 1996), svenska (Birn, 2000; Arppe, 2000; Sågvall-Hein 1998; Domeij et al, 1996; Andersson et al, 1999), danska (Paggio, 2000), norska (de Smedt & Rosén, 2000) och tyska (Bredenkamp, 2000) för att ge några exempel. För användarna har utvecklingen medfört att fler och fler språk läggs till i ordbehandlarnas grammatikkontrollmoduler.

Det finns också ett ökat intresse för så kallade kontrollerade språk (se t.ex. Almqvist & Sågvall-Hein, 1996). Ett kontrollerat språk består ofta av en delmängd av ett språk, till exempel att endast vissa konstruktioner eller uttryck är godkända i språket. Begränsningarna införs dels för att möjliggöra en mycket täckande språkgranskning av språket och dels för att konstruktioner i det kontrollerade språket skall kunna hanteras av andra automatiska språksystem. Den främsta tillämpningen för kontrollerade språk är översättning, såväl manuell som maskinell. För översättning används grammatikkontrollen dels som en förprocess; programmet plockar bort oönskade konstruktioner som skribenten har konstruerat, och dels som en efterprocess efter den maskinella översättningen, programmet hittar och rättar fel i den maskingenererade texten. För svenska finns t.ex. det kontrollerade språket Scaniasvenska (Almqvist & Sågvall Hein, 1996).

Vid konstruktionen av ett kontrollerat språk bestämmer konstruktörerna vad som skall anses som grammatiskt eller ogrammatiskt. När det gäller utvecklingen av ett språkgranskningsprogram för ett riktigt språk måste konstruktörerna ha ett annat förhållningssätt till det grammatiska respektive det ogrammatiska, eftersom gränsen finns därute hos språkbrukarna och bara delvis i grammatikor och språkvårdsböcker. Detta medför ju också att utvecklarna av ett språkgranskningsprogram påverkar vilka feltyper som finns med i det slutgiltiga programmet. Urvalet av vilka feltyper som skall finnas med blir därför subjektivt och det är därför mycket viktigt med omfattande korpusstudier och insamling av texter med fel i. Urvalet i Granska är delvis baserat på en undersökning om inkongruens i substantivfraser och en undersökning om särskrivna sammansättningar (Domeij, Knutsson & Öhrman 1999).

Grammatikkontroll används med viss framgång inom andraspråksinlärning (se t.ex. Chen 1997; Bolt 1992; Yazdani, 1990). Granska har testats mot texter skrivna av personer med svenska som andra språk (se Öhrman, 2000). Resultaten är delvis lovande och en fortsatt inriktning med pedagogiska förtecken vore en intressant utveckling av denna typ av program. Inom detta område och språkinlärning överhuvudtaget kan grammatikkontroll eller teknik från grammatikkontroll säkerligen utvecklas vidare.

2.1 Övergripande utgångspunkter

Gränsen mellan grammatiskt och ogrammatiskt är en väsentlig del av all språkbeskrivning. En grammatik som gör anspråk på att beskriva grammatiska satser och meningar i ett språk måste på något sätt förhålla sig till de ogrammatiska. De flesta grammatiska beskrivningar brukar vara försedda med ogrammatiska exempel som ett slags motexempel till satser som grammatiken avser att beskriva. Studier och analysmetoder av ogrammatiska konstruktioner är alltså viktiga inte bara i ett språkgranskningsperspektiv, utan för analys av språk överhuvudtaget. Detta blir kanske än mer uppenbart när grammatiska analysverktyg skall implementeras, än det är för den rent teoretiska lingvisten.

I ett mer formellt grammatiskt perspektiv på språklig analys har Chomskys tidiga teorier om en grammatik som antingen accepterar eller förkastar satser varit dominerande (Chomsky, 1957)[3]. Med en generativ grammatik skulle man tala om att grammatiken antingen genererar en viss sats eller inte. Om grammatiken genererar satsen tillhör den språket. Om satsen tillhör språket får satsen en analys, om satsen inte tillhör språket genereras ingen analys alls. I viss mån har program för grammatikkontroll bidragit till en mer nyanserad syn på gränslandet mellan det grammatiska och det ogrammatiska.

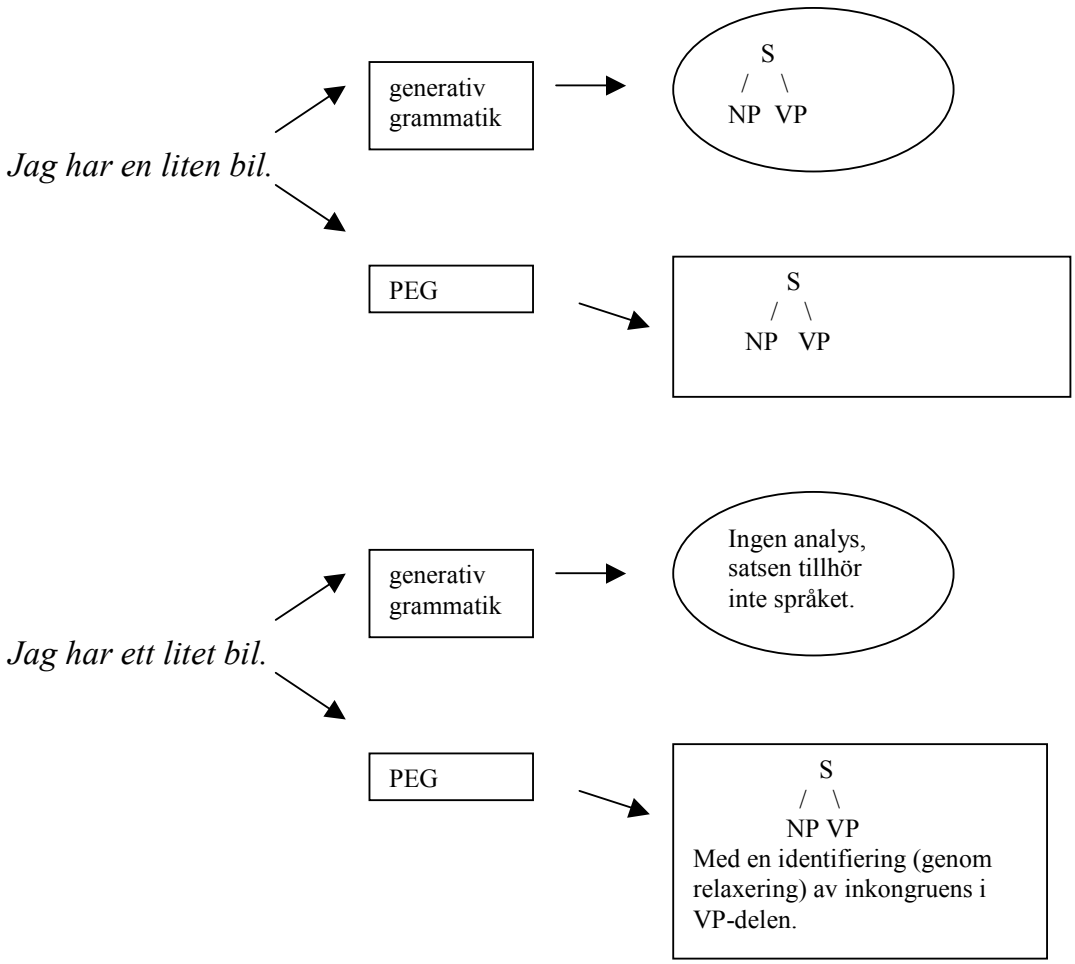
[3] Den senare Chomsky talar om att tilldela alla fonetiskt möjliga satser någon form av analys (Ramer, 1993).

Vid utvecklingen av språkgranskningsprogrammet Critique (Ravin, 1993; Richardson & Braden-Harder, 1993) användes en formalism/programspråk för automatisk grammatisk analys, kallat PLNLP (Jensen et al, 1993). Critique använder den engelska grammatiken English Grammar (PEG) implementerad i PLNLP, som på ett generellt sätt, det vill säga inte enbart för grammatikkontroll, kan hantera ogrammatiska satser (Jensen, 1993). Den grundläggande idén är att alla satser skall få något slags analys, även om de är ogrammatiska. Metoderna som används kallas *relaxering* och *parse fitting*, och beskrivs i avsnitt 2.3.

En analys av ogrammatiska satser hade inte varit möjlig med en klassisk generativ grammatik som endast genererar grammatiskt korrekta satser. PEG:s analys kan användas av ett språkgranskningsprogram för att ställa en diagnos på felet och ta fram ett lämpligt ersättningsförslag. Metoden blir sålunda mycket användbar för löpande text som i många fall både innehåller grammatiska fel och även satser som ligger på gränsen till det ogrammatiska. Dessa insikter verkar ha påverkat mer moderna grammatikformalismer för löpande text, t.ex. Constraint Grammar (Karlsson, 1995b) och Functional Dependency Grammar (Järvinen & Tapanainen, 1998). Figur 1 visar skillnaden mellan hur en hårdragen klassisk generativ grammatik och PEG hanterar ogrammatiska satser.[4]

PEG och dess användning i Critique är mycket tilltalande, men bygger på ett mycket omfattande arbete. Man bör också ha klart för sig, att det trots PEG:s generalitet behövs någon form av regler som ställer feldiagnoser och genererar ersättningsförslag. Det finns också risker med att använda ett generellt system; riskerna består framför allt av att klassiska strukturella flertydigheter eventuellt måste lösas av systemet. Om systemet misslyckas med att lösa dessa flertydigheter, kan dessa medföra att falska alarm uppstår eller att fel inte upptäcks. Konstruktörerna av ett system måste noggrant överväga hur pass omfattande analys som behövs för detektionen av varje enskild feltyp.

[4] Ramer (1993) hävdar att PEG tillhör s.k. transduktiv lingvistik (transductive linguistics).



Figur 2.1. Skillnaden mellan en hårdragen klassisk generativ grammatik och PEG för hantering av två satser. Om satsen är ogrammatisk får vi av den generativa grammatiken veta att satsen inte tillhör språket. PEG däremot tilldelar satsen en analys som visar att det förekommer inkongruens i frasen ”ett litet bil”.

2.2 Utgångspunkter för automatisk grammatisk analys av svenska

För datorlingvisten är grammatiska beskrivningar centrala vid en implementation. Om inte de grammatiska strukturer som skall implementeras finns väl beskrivna, måste grammatikern göra denna beskrivning, vilket ofta är en övermäktig uppgift. Detta gäller också för den som skall implementera ett system för språkgranskning; om inte de korrekta konstruktionerna finns detaljerat beskrivna, är det mycket svårt att skilja ut de felaktiga från de korrekta. För engelska finns det inte bara fler formella grammatiker implementerade än för svenska, de grammatiska beskrivningarna är också fler och mer omfattande. För svenska innebar Svenska Akademiens grammatik (1999) något av ett genombrott, vad det gäller den

grammatiska beskrivningen.[5] De enkelt implementerbara beskrivningarna i SAG är tyvärr inte så frekventa, utan detta arbete får grammatikern utföra på egen hand. SAG påpekar också i vissa fall hur språkvården förhåller sig till vissa språkliga konstruktioner; denna information har också varit mycket användbar. Det är rimligt att ett språkgranskningssystem försöker följa de riktlinjer som dras upp av en majoritet av svenska språkvårdare.

En grammatikkontroll kräver någon form av grundläggande grammatisk analys. Denna analys kan variera i nivå; från en morfosyntaktisk (ordklasser och morfologiska särdrag) till en full syntaktisk analys av satsen med full analys av frasstruktur (analys av fraser och hur de hänger ihop med varandra) och syntaktiska funktioner (subjekt och objekt osv.). För engelska finns det program som utför full analys av satser i löpande text, men för många andra språk, t.ex. svenska, saknas sådana program. För en bra översikt över olika system och metoder för syntaktisk analys främst för engelska se Karlsson (1995a).

Det finns idag inte några färdiga verktyg för att analysera hela satser och meningar vad det gäller svensk syntax, åtminstone inte med traditionell frasstrukturgrammatik. Gambäcks implementation av en unifieringsbaserad grammatik (Gambäck, 1997) klarar vid ett litet test med 2000 ord ur Stockholm-Umeå Corpus (Ejerhed et al, 1992) endast att analysera 28 % av de 131 meningarna i testtexten. Det måste sägas att Gambäcks grammatik och lexikon inte är konstruerade för löpande text. Men testresultaten för Gambäcks grammatik ger ändå en bra bild hur tillståndet är för implementationer av svenska grammatiksystem, och vad det finns att utgå ifrån när det gäller en implementation av en svensk grammatikkontroll. Att försöka sig på en fullständig frasstrukturanalys i Granska-projektet var därför uteslutet. Detta är ju något som görs i flera grammatikkontroller för engelska, t.ex. Critique.

Det finns flera olika system för morfosyntaktisk analys av svenska. Dessa program väljer vilken ordklass och morfologiska särdrag ett ord skall ha i dess språkliga kontext. Metoderna spänner från regelbaserade system (t.ex. Birn, 1998), via maskininlärning av regler (t.ex. Lindberg & Eineborg, 1998) till statistiskt baserad analys (t.ex. Carlberger & Kann 1999; Eriksson 1992; Åström 1998). Även om det är svårt att jämföra utvärderingar uppnår samtliga system en korrekthet runt 95 %. För en bra beskrivning av hur dessa metoder och algoritmer fungerar se t.ex. (Jurafsky & Martin, 2000).

För grammatikkontroll är det önskvärt med en mer omfattande analys än en morfosyntaktisk. Det vore till exempel önskvärt med en analys av fraser och syntaktiska funktioner. Om man vänder sig bort från frasstrukturgrammatiken

[5] Tyvärr återstod mindre än sex månader av Granska-projektet när Svenska akademinens grammatik (SAG) släpptes på bokdiskarna. Under de alltför få månader som jag har arbetat med SAG, har jag funnit den mycket användbar, och hittills har jag funnit alla eftersökta konstruktioner och även en mängd andra konstruktioner av relevans för automatisk språkgranskning.

finner man t.ex. ytsyntaktisk analys med Constraint Grammar (Karlsson, 1995b) som är konstruerad för analys av löpande text. Constraint Grammar fungerar reduktionistiskt, analysen utgår från flertydig lexikal analys. En omfattande samling regler plockar successivt bort de flertydigheter som finns. Detta förfarande fortgår tills det inte går att tillämpa fler regler. Vid svåra fall lämnas flertydigheter kvar. En svensk version av Constraint Grammar (Birn, 1998) innehåller dock i nuläget endast ett begränsat antal syntaktiska taggar och är fortfarande under utveckling. Den engelska versionen av Constraint Grammar, ENGlish Constraint Grammar (ENGCG) (Voutilainen et al, 1992) har en intressantare tagguppsättning, med flera syntaktiska taggar som skulle vara mycket användbara för grammatikkontroll. ENGCG har uppnått mycket bra resultat med avseende på såväl täckning som precision (Voutilainen, 1995). När en svensk version av Constraint Grammar har byggts ut med motsvarande tagguppsättning kommer den att ge möjligheter till en ganska omfattande grammatikkontroll.

Det verkar också finnas en Functional Dependency Grammar för svenska[6] under utveckling; denna bör vara mycket användbar i en grammatikkontroll eftersom en ytsyntaktisk analys kan ges inom ramen för en dependensgrammatik. Implementationer av dependensgrammtik används t.ex. i kommersiella grammatikkontroller för engelska, spanska, franska och portugisiska (Bourdon et al, 1998)[7].

Såväl Constraint Grammar som Functional Dependency Grammar har som uttalat mål att hantera ogrammatisk text, vilket gör dessa mer lämpade än system där ingen analys av ogrammatiska satser ges.

Ett annat ytsyntaktiskt analysverktyg har implementerats med goda resultat av (Kokkinakis & Johansson-Kokkinakis, 1999a) och det är möjligt att detta program skulle vara användbart som en förprocess för grammatikkontroll. En mer traditionell grammatisk analysator är Uppsala Chart Parser (Sågvall Hein, 1981) som har vidareutvecklats i Scarrie, en grammatikkontroll för svenska (Sågvall Hein, 1998). Hur ogrammatiska konstruktioner angrips med hjälp av Uppsala Chart Parser beskrivs kort i avsnittet *Grammatikkontroll för svenska*.

För en kort, men bra, överblick över implementerade svenska grammatiksystem se (Gambäck, 1997).

2.3 Automatisk språkgranskning för andra språk

Det finns många system för språkgranskning, men jag väljer att kort beskriva två kända och väldokumenterade system: Critique och CORRie. Det har tidigare framgått att många och omfattande system har utvecklats för engelska. Det är därför på sin plats med en kortfattad beskrivning av ett av de mest kända

[6] Se www.conexor.fi
[7] se också www.machinasapiens.com

systemen, som från början kallades Epistle (Jensen et al 1983), men sedermera bytte namn till Critique. Critique utvecklades först och främst för engelska, men tanken var att applicera tekniken på flera andra språk, varav åtminstone ett system för franska blev verklighet (Chanod, 1993).

Det kanske mest intressanta med Critique är det generella angreppssätt som utvecklarna hade med systemet och målet att alla satser skulle tilldelas något slags analys och en så omfattande analys som möjligt (fullständig syntaktisk analys).

Mer i detalj angriper Critique okända och ogrammatiska satser på följande sätt.

- Om ett ord i satsen är okänt tilldelas det några förvalda värden; detta räcker i många fall för att få fram en rimlig analys.
- Om kraven i grammatiken är för stränga lättas dessa upp, för att åstadkomma en analys (s.k. relaxering). Relaxering innebär att vissa krav på syntaktisk korrekthet i grammatiken lättas upp och därmed kan en del ogrammatiska satser analyseras. Regler för lexikala sammanblandningar används också för att få bort vanliga sammanblandningar som t.ex. *whose – who's*.
- Om systemet trots allt inte lyckas kan det bero på svagheter i den konstruerade grammatiken eller att systemet stött på ett oförutsägbart fel, och då används i Critique metoden ”parse fitting”, som går ut på att med olika metoder välja ut en huvudkonstituent i satsen. Den konstituent som täcker mest i satsen från vänster till höger blir första kandidat t.ex. en verbfras. Omkring denna huvudkonstituent passas övriga konstituenter in enligt ett visst schema. När hela satsen täcks upp av delvis sammankopplade konstituenter är analysen klar.
- Om flera analysalternativ ges, finns det en metrik som tar fram den troligaste analysen. Metriken föredrar analyser där modifierade ord och fraser (fel) kopplas till närmaste lämpliga konstituent.

Algoritmer för feligenkänning arbetar mot alla stegen i analysen. Algoritmer som upptäcker grammatiska fel är inbyggda i den robusta analysen. Om en analys lyckas genom att lätta på kravet på t.ex. subjekt-verb kongruens så markeras orden i texten och ett ersättningsförslag med kommentar presenteras för användaren.

Critique hanterar 25 olika feltyper som är indelade i fem kategorier:

- Numerusinkongruens, t.ex. *He go* (He goes) och *Many book (*Many books). *The man who come to dinner* (The man who comes to dinner). *It clarifies and enforce* (It clarifies and enforces).
- Fel pronomenform, t.ex. *between you and I* (between you and me)
- Fel verbform, t.ex. *had expect* (had expected) och *seems to been* (seems to be)

- Skiljeteckensfel, t.ex. kommatecken används felaktigt istället för semikolon eller punkt; det kan också vara frågor som avslutas med punkt istället för frågetecken.
- Sammanblandningar av ord och uttryck, t.ex. who's - whose, you're - your, tangentsbordsfel, t.ex. *from* blir *form* som inte upptäcks av stavningskontrollen.

Critique upptäcker också 85 stilistiska problem. En viktig skillnad mot de grammatiska felen är att det inte finns några givna ersättningsförslag till stilproblemen. Critique särskiljer därför grammatik- och stilfel i presentationen till användaren.

Ett annat väldokumenterat program för språkgranskning är CORRie (Vosse, 1994). Vosses omfattande beskrivning är intressant dels för att programmet skall hitta språkfel i ett annat språk än engelska, nämligen holländska, och dels för att Vosse gör en övergripande klassificering av feltyper. I holländskan förekommer andra feltyper än i engelskan, varav en är särskrivna sammansättningar.

Klassificering av feltyper är användbar för att se förhållandet mellan det grammatiska och det ogrammatiska. Klassificeringen bör också vara användbar då man vill jämföra språkgranskningsprogram för olika språk och värdera komplexiteten hos felen. Vosse gör följande klassificering av olika typer av fel:

substitution: skribenten ersätter ord eller tecken med något som medför att ett fel uppstår, t.ex. stavfel som *sjribent* (skribent).

deletion: skribenten glömmer ett ord eller en hel fras, t.ex. *Tåget gått* (Tåget har gått)

insertion: skribenten sätter in ett tecken, ord eller en fras för mycket, t.ex. *Tåget har har gått.* Särskrivna sammansättningar som *hus bil* tillhör denna felkategori genom att skribenten har satt in ett mellanslag mellan *hus* och *bil*.

transposition: skribenten byter plats på ord, t.ex. *Tåget gått har*.

feature mismatch: skribenten väljer fel form av ett ord eller fras. Enligt Vosse kan feature mismatch ses som ett specialfall av substitution. Inkongruens i nominalfraser som t.ex. *den lilla huset* och inkongruens i predikativ t.ex. *Mannen är glada* tillhör denna felkategori.

Vosse hävdar vidare att **feature mismatch** är den grupp som elegantast kan hanteras i den grammatikformalism som används i CORRie, kallad Augmented Context-Free Grammars (ACFG), genom någon form av s.k. relaxering som i Critique. Övriga feltyper (strukturella fel) kräver speciella felregler som kan implementeras direkt i ACFG. För att hantera multipla fel eller val mellan flera möjliga fel i samma ordsekvens kan felreglerna viktas.

2.4 Automatisk språkgranskning för svenska

Under senare delen av 1990-talet har det bedrivits forskning och utveckling av grammatikkontroll för svenska av minst fyra grupper[8].

1. Språkgranskningprogrammet Granska utvecklas av en forskargrupp vid Nada, KTH (Domeij, Larsson & Knutsson, 1996; Domeij, Knutsson, Larsson, Severinsson Eklundh & Rex, 1998; Carlberger, Domeij, Kann & Knutsson, 2000). Granska presenteras mer utförligt i avsnitt 2.5.
2. Det finska språkteknologiföretaget Lingsoft släppte hösten 1998 den kommersiella grammatikkontrollen Grammatifix (Arppe, 2000; Birn, 2000).
3. Institutionen för lingvistik vid Uppsala Universitet utvecklar i samarbete med danska och norska grupper, inom ett EU-projekt, språkgranskningsprogrammet Scarrie (Sågvall-Hein, 1998).
4. Institutionen för lingvistik vid Göteborgs universitet utvecklar metoder för att med hjälp av positiva grammatikregler, det vill säga utan specifika felregler (negativa grammatikregler), detektera olika typer av grammatiska fel i svenskan (Andersson et al, 1999).

Det är intressant att var och en presenterat delvis en egen metod för att komma åt felen. Gemensamt för systemen är någon form av lokal analys, men den görs på olika sätt.

Grammatifix är en kommersiell produkt, medan Scarrie och Granska är forskningsprototyper. En jämförelse mellan systemen blir därför inte rättvis, men i stort hittar de tre olika programmen ungefär samma feltyper (se Arppe, 2000, för en jämförelse mellan system). Det stora undantaget är särskrivna sammansättningar, där Granska verkar eftersöka flest varianter av feltypen. Feltypen behandlas i någon mån av Scarrie, men inte alls av Grammatifix. Grammatifix och Granska ger också ersättningsförslag där så är möjligt, vilket inte Scarrie gör. Att göra en omfattande studie och jämföra täckning och precision är ett stort och komplicerat företag, som hittills inte utförts av någon grupp, åtminstone inte i publicerad form.

Grammatifix använder sig av morfologiskt taggad text från den morfologiska analysatorn SWETWOL (Karlsson, 1992), som i de allra flesta fall är entydiggjord (disambiguerad) med regler skrivna i Constraint Grammar. Disambigueringen utgår från Lingsofts svenska version av Constraint Grammar (SWECG, Birn 1998), men är något modifierad för att hantera ogrammatiska indata (Birn, 2000).

Språkgranskningen i Grammatifix bygger på felspecifika regler skrivna i Constraint Grammar (CG). Det vill säga varje regel söker efter en viss feltyp. CG är framförallt ett system för taggning och disambiguering av text i olika nivåer,

[8] Arppe ger en historisk tillbakablick i Arppe (2000).

men i Grammatifix används formalismen för att detektera grammatiska fel, vilket i sig är intressant. Man använder helt enkelt CG för att tagga upp de konstruktioner som enligt en uppsättning regler bör vara fel. Varje regel i sig avgör med kontextuella villkor om en sekvens av ord verkar vara grammatiskt fel. En jämförelse mellan två regler i Grammatifix och Granska görs i kapitel 3.

Scarrie eller ScarCheck[9] som språkgranskningsmotorn i den svenska versionen egentligen heter (Sågvall-Hein, 1998), är det system för svenska som mest påminner om Critique och Corrie genom att bygga på en parser för fullständig analys, Uppsala Chart Parser (UCP) (Sågvall-Hein, 1981). En av finesserna med chart-parsing är att även delanalyser är möjliga i systemet, även om inte hela satsen kan analyseras. I Scarrie används dels s.k. relaxering, kraven i de grammatiska reglerna lättas upp för att identifiera fel, och dels specifika felregler som appliceras på den information som kommer ut från UCP. De flesta meningar som analyseras av UCP får endast ett antal delanalyser som inte byggs ihop till ett fullt syntaxträd. ScarCheck bygger därmed på s.k. partiell parsning. Men satser som *Det bli förmodligen det slutgiltiga siffrorna* täcks fullt ut av grammatiken i ScarCheck.

En forskargrupp vid institutionen för lingvistik vid Göteborgs universitet har arbetat med något annorlunda metoder (Andersson et al, 1999). Metoden bygger på Karttunens algoritm (Kartunnen et al, 1996) som upptäcker felaktiga datum utan att explicit beskriva hur ett felaktigt datum ser ut. Denna metod har Andersson et al (1999) överfört till grammatikgranskning och lyckas till viss del upptäcka inkongruens i nominalfrasen. Den stora fördelen med metoden är att lingvisten aldrig behöver skriva specifika felregler utan kan skriva ”vanliga” grammatikregler som anger hur t.ex. en enkel nominalfras ser ut. Genom en subtraktionsmekanism mellan regler med olika krav på grammatisk korrekthet kan inkongruens i nominalfraser upptäckas. Denna typ av regler kan implementeras i Granska genom det samarbetsprojekt som Nada och forskargruppen vid Göteborgs Universitet deltog i; detta har dock inte genomförts i någon större skala (se vidare i kapitel 3).

2.5 Granskas angreppssätt

Om PEG (se avsnitt 2.1) inriktar sig på att analysera såväl grammatiska som ogrammatiska satser, så inriktar sig Granska på en del av de satser som verkar innehålla ogrammatiska sekvenser av ord. På detta sätt blir Granska inte ett generellt system för språklig analys, men när det kommer till den specifika tillämpningen, automatisk språkgranskning, kan Granska nå goda resultat.

[9] Den svenska version av Scarrie använder en egenutvecklad grammatikkontroll som bygger på Uppsala Chart Parser (Sågvall-Hein, 1981). Svenska Scarrie skiljer sig därmed från de danska och norska versionerna som bygger på CORRie (Vosse, 1994).

Vosse rekommenderar att man är försiktig med en fullständig analys av satser och meningar (Vosse, 1994) eftersom det är en svår och tung process, och endast när det behövs skall den användas. I Granska används en partiell parsning med s.k. hjälpregler när feltypen kräver en mer omfattande analys. På detta sätt behöver inte en fullständig grammatik implementeras och därmed undviks många av de problem som följer med en mer fullständig analys av meningen, det vill säga många flertydigheter som är mycket svåra att lösa upp (se t.ex. Voutilainen, 1994).

Granska undersöker endast de meningar som verkar vara fel och i dessa görs en mer omfattande analys av meningen, med t.ex. identifikation av nominalfraser, prepositionsfraser och satsgränser, se vidare 2.5.4.

Delar av den presentation av Granska som följer är hämtad från artikeln ”Granska – an efficient hybrid system for Swedish grammar checking” skriven tillsammans med Rickard Domeij, Johan Carlberger och Viggo Kann (2000).

2.5.1 Feltyper som detekteras av Granska

Denna avhandling är koncentrerad på tre feltyper: särskrivna sammansättningar, inkongruens i nominalfrasen och inkongruens i predikativ, men regler för andra feltyper har också implementerats. Detta har dock inte utförts med samma noggrannhet som de tre feltyper som avhandlingen behandlar, utan mer med syftet att visa att även andra typer av fel kan implementeras i Granska med gott resultat.

Granska har funnits i flera olika versioner och har sitt ursprung i Plita (Domeij, 1994). I tidigare versioner av Granska (Domeij et al, 1998) fanns de flesta reglerna från Plita med; dessa regler sökte främst efter ortografiska fel, t.ex. dubbla mellanslag och efter stilistiska problem t.ex. *medans*. Nuvarande version har koncentrerats på mer grammatiska fel som presenteras kortfattat nedan.

Feltyp	**Exempel**
Felaktiga sammansättningar, enligt Svenska skrivregler (1991).	*undertiden*
Särskrivna sammansättningar	*Han orkade inte flytta **sten bumlingarna**.*
Kontamination, sammanblandning av uttryck	*tillhör en av* (ska vara *är en av* eller bara *tillhör*)
Felaktig preposition, signalerar för fel i fasta uttryck enligt Svensk handordbok (1966).	*med utgångspunkt från* (bör vara *med utgångspunkt i*)
Objektsform efter preposition	*Han skickade en rad vänliga brev till **de**.*
Inkongruens i nominalfraser	*Han såg **de gröna villor**.*
Inkongruens i predikativ	***Äktenskapet är baserad** på kärlek mellan man och kvinna*
Böjningsfel vid främmande ord	*Han köpte **ett data**.*
Tautologi	*orsaken till anledningen*
Böjningsfel i verbfras	*Vi **kommer spela** en låt av Ebba Grön*
Ordföljdsfel	*Han sa att **han sparkade inte** bollen*

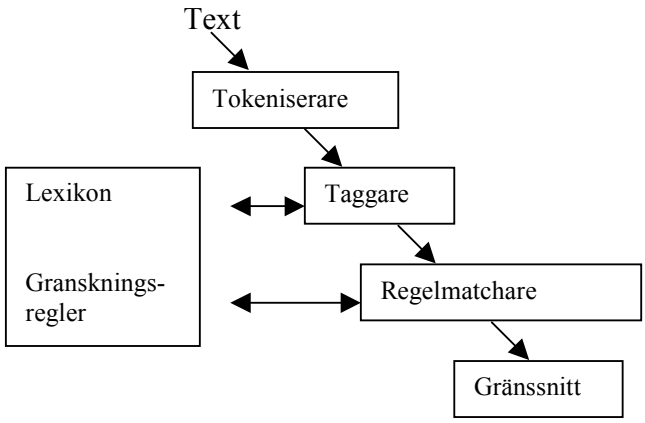
Tabell 2.1. Ett urval av de feltyper som Granska kan detektera, diagnostisera och korrigera.

2.5.2 Översikt över Granskasystemet

Granska är ett hybridsystem som använder ytgrammatiska regler för att kontrollera svensk grammatik. Systemet kombinerar probabilistiska och regelbaserade metoder för att uppnå hög effektivitet och robusthet. Detta är en nödvändig förutsättning för en grammatikkontroll som ska fungera i realtid i direkt interaktion med användaren (se t.ex. Kukich, 1992). Med hjälp av särskilda granskningsregler kan ett antal grammatiska fel i svensk text detekteras och förses med korrektionsförslag.

Hur granskningen är uppbyggd i moduler visas i figur 2.2. Först görs en igenkänning av alla textens ord i tokeniseraren. I nästa steg tilldelas orden information om ordklasser och morfologiska särdrag med hjälp av en s.k. taggare som slår upp ord i lexikon och på statistisk grund räknar ut vilken av flera möjliga ordklasstilldelningar ett ord får i sin kontext. Den taggade texten skickas sedan vidare till regelmatcharen som genomsöker texten och matchar den mot de förväntade grammatiska avvikelser som definierats i granskningsreglerna.

Granskningsreglerna genererar också felbeskrivningar och rättningsförslag som slutligen presenteras för användaren i ett grafiskt gränssnitt. Systemet innehåller även en rättstavningsmodul, kallad Stava, som kan hantera svenska sammansättningar (Kann et al, 1999) och som används i granskningsreglerna för att kontrollera felaktiga särskrivningar.



Figur 2.2. Granskasystemets uppbyggnad

2.5.3 Grundläggande analys – morfologisk analys och statistisk disambiguering

Granska använder en gömd markovmodell (Carlberger & Kann, 1999) för att göra en grundläggande språklig analys och taggning av texten. Varje ord tilldelas en tagg som beskriver ordets ordklass och morfologiska särdrag. Taggningen görs på basis av ett lexikon på 160 000 ordformer som är konstruerat utifrån en handtaggad korpus på en miljon ord (SUC – Stockholm Umeå Corpus, Ejerhed et al, 1992) och kompletterat med ord från SAOL (Svenska akademiens ordlista, 1986). Markovmodellen bygger på statistik om ords och taggars förekomster i kontexten från texterna i SUC. På så sätt kan den välja den mest sannolika taggen för varje ord i texten som finns i lexikon. Okända ord taggas med hjälp av probabilistisk ordslutsanalys. All denna grundläggande analys görs av ett program som i fortsättningen kallas *Granskas taggare*.

Trots att Voutilainen och Samuelsson (Samuelsson & Voutilainen, 1997) har hävdat att regelbaserad taggdisambiguering är bättre än statistisk, åtminstone för engelska, finns det ännu så länge inte några bevis för att detta gäller även för svensk text med ogrammatiska inslag. Frågan om vilka metod som är bäst är fortfarande öppen, men Birn (2000) hävdar att disambiguering är nödvändigt för att uppnå hög precision vid detektion av grammatiska fel. I en mindre utvärdering (Carlberger & Kann, 1999) med 50 000 löpord från SUC, fann vi att dåvarande version av Granska (Domeij, et al, 1998) genererade 94 falska alarm med

morfosyntaktiskt flertydig indata, men endast 8 falska alarm med Granskas taggare som förprocess. I en annan mindre utvärdering utförd av (Knutsson, 1996) med en uppsättning språkgranskningsregler för inkongruens i nominalfrasen, jämfördes den flertydiga analysen från SWETWOL med den entydiga från en svensk version av den statistiska taggaren XPOST (Cutting et al, 1992). Det framkom att precisionen ökade avsevärt med entydiggjord indata från en uppsättning studentuppsatser.

Å andra sidan påverkas disambigueringen av de grammatiska felen; en gemensam konsekvens för regelbaserad och statistisk disambiguering är att det uppstår disambigueringsfel, t.ex. att en determinerare tolkas som ett pronomen, som *det* i *Jag såg det mannen redan igår*. Jämför med den korrekta satsen *Jag såg det mannen talade om redan igår*. Metoderna skiljer sig genom att den regelbaserade disambigueringen (åtminstone Constraint Grammar) lämnar tvetydighet kvar om ordsekvensens flertydighet inte kan lösas upp av reglerna. Den statistiska taggningen (åtminstone Granskas taggare) försöker alltid att lösa flertydigheter och endast en morfosyntaktisk analys av ordet ges. Vilken metod som egentligen är bäst är ännu så länge inte klart, och här återstår många forskningsfrågor. Den grundläggande analysen är mycket viktig för ett gott slutresultat eftersom efterföljande regler bygger på resultatet från dessa.

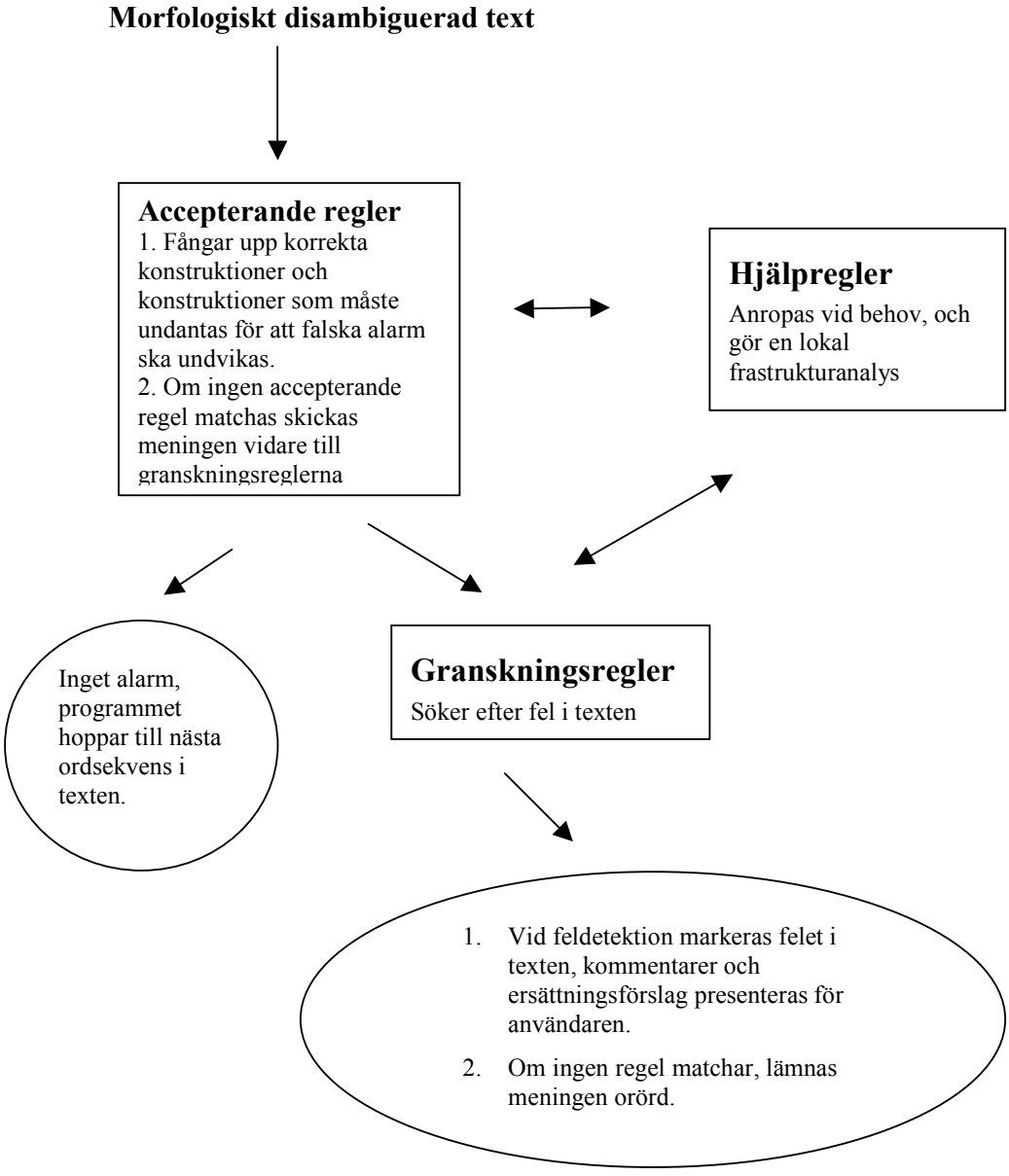
2.5.4 Regler och regelspråk

Granska har särskilda granskningsregler som genomsöker resultatet från den morfosyntaktiska taggningen på jakt efter grammatiska felaktigheter. Eftersom markovmodellen även taggar morfosyntaktiskt avvikande ord med endast en tagg behöver i normalfallet ingen ytterligare disambiguering göras i granskningsreglerna för att detektera t.ex. ett kongruensfel som ***en*** *hus*. Dessa granskningsregler kommer att beskrivas i detalj för respektive feltyp och dessutom i kapitlet *Granskas regelspråk* som beskriver den syntax som reglerna implementeras i.

I Granskas regelspråk kan s.k. hjälpregler som utför delanalyser av meningen konstrueras. Med hjälpreglerna närmar sig analysen en frasstrukturgrammatik. Poängen med hjälpreglerna är att det går att göra en ytsyntaktisk frasanalys lokalt, utan att systemet behöver analysera de delar av satsen som inte är relevanta för lokaliseringen av den aktuella feltypen. Därmed bevaras systemets höga effektivitet samtidigt som den språkliga analysen får det djup som behövs i det enskilda fallet.

För att undvika falska alarm finns det en grupp regler som plockar bort de konstruktioner som sannolikt skulle matchas av granskningsreglerna. Dessa regler kallas accepterande regler och har visat sig vara ett effektivt vapen mot falska alarm. Fördelar är främst att kritiska korrekta konstruktioner inte behöver granskas med granskningsreglerna, samt att granskningsreglerna i sig blir renare och inte

fulla av undantag. Nackdelen är att en granskningsregel inte ensamt anger hur ett fel realiseras. Figur 2.3 visar hur de olika reglerna samverkar för att hitta så många fel som möjligt, utan att generera falska alarm.



Figur 2.3. Figuren visar hur accepterande regler, hjälpregler och granskningsregler samverkar för att hitta fel och undvika falska alarm.

2.5.4.1 Accepterande regler

Accepterande regler hanterar många undantag som hade behövt implementeras i varje enskild felregel. Konstruktioner som hanteras med accepterande regler är t.ex. *en massa arbetare*, där *massa arbetare* skulle kunna vara en tänkbar

särskriven sammansättning av *massaarbetare* (en arbetare som arbetar med pappersmassa). De accepterande reglerna medför att granskningsreglerna inte kan läsas och förstås utan att man först betraktar de regler som accepterar konstruktioner. Detta angreppssätt bör löna sig genom att man får en beskrivning av det som är korrekt och först när det till exempel inte finns några korrekta nominalfraser kvar i texten appliceras felreglerna. Det är också ett sätt att förenkla för regelskrivaren som först kan koncentrera sig på korrekta konstruktioner och därefter skriva regler för konstruktioner som ofta betraktas som fel. Uppdelningen mellan accepterande regler och felregler blir emellertid inte riktigt så precis i verkligheten, vilket de regelexempel som senare presenteras kommer att visa. Antalet matchningar som måste göras minskas också radikalt. Den enda nackdelen är att en felregel inte ensam anger hur ett fel realiseras, utan kombinationen felregel och accepterande regel måste hanteras. Flera exempel på hur accepterande regler konstrueras ges i efterföljande kapitel.

2.5.4.2 Hjälpregler

Tanken med hjälpregler är att analysera tillräckligt mycket av meningarna för att kunna applicera felreglerna på ett mer precist sätt. Uppbyggnaden av den samling hjälpregler som nu finns och som kommer att presenteras i den följande texten har vuxit fram i takt med att felreglerna har blivit mer avancerade och att det är onödigt att försöka uttrycka den språkliga variationen i varje enskild felregel. Ett kontextvillkor för en nominalfrasregel och för en särskrivningsregel kan vara de samma. Hjälpreglerna är ett sätt att återanvända lingvistisk kunskap i systemet. Man kan också se hjälpreglerna som en sorts förprocess för felreglerna; denna förprocess initieras dock endast om någon felregel appliceras på en mening i texten. Hjälpreglerna ligger alltså inte som en bakomliggande process, utan de används endast när felreglerna appliceras.

Samtidigt som hjälpreglerna förbättrar den grundläggande lingvistiska analysen, inför de nya flertydigheter med flera olika analyser av en ordsekvens, t.ex. om *den man* i meningen *den man såg igår ser man inte idag*, skall betraktas som en (*den man*) nominalfras eller som två nominalfraser (*den* och *man*).

2.5.4.3 Detektion av nominalfraser

En viktig drivkraft för hjälpreglernas utveckling är en önskan om ökad täckning och förbättrad precision för felreglerna. Utvecklingen av felreglerna för framför allt inkongruens i predikativ och de accepterande reglerna har drivit fram en kraftig utveckling av framförallt regler för nominalfrasigenkänning. Att detektera inkongruensfel i predikativ kräver i ganska många fall att nästan hela meningen analyseras, eller åtminstone satsen; det kan handla om en ordsekvens på mer än tio ord, t.ex. ***Problemet*** *med den kommunala fördelningen från inkomstskatter för säsongsanställda är* ***svår*** *att lösa inom befintligt skattesystem*, där *Problemet* och *svår* är inkongruenta. Nominalfrasen ***Problemet*** *med den kommunala fördelningen*

från inkomstskatter för säsongsanställda måste identifieras för att inkongruensen i predikativ skall upptäckas.

Hjälpreglerna är skrivna på ett ganska grovt sätt; en tanke är de skall klara felaktiga konstruktioner, eftersom det är möjligt att sådana finns i kontexten till andra fel. Det är t.ex. ganska vanligt hos andraspråksinlärare, se t.ex. (Öhrman, 2000) och även hos språkbrukare med sverigesvenska som modersmål att t.ex. ett särskrivningsfel förekommer i kontexten av ett predikativfel. Reglernas grovhet var också tänkt att användas inom området språklig redigering[10] (se t.ex. Severinson Eklundh, 1991; Dijkstra & Huls 1992), där den lingvistiska analysen måste kunna hantera fraser och satser som är under revidering, och därmed i många fall innehåller fel. Det är framförallt nominalfrasreglerna som är skrivna avsiktligt grova och detta drar ner precisionen för varje enskild nominalfrasregel, men kan öka såväl täckning som precision för felreglerna; felreglerna berörs genom grovheten inte på samma sätt av felaktigheter i satsen som om felregeln utgick från att övriga konstituenter var korrekta. Ett sätt att höja precisionen för varje nominalfrasregel vore att de ingående konstituenterna skall kongruera. Hittills har kongruens inte använts i hjälpreglerna och en orsak har redan nämnts; den andra är att den svenska nominalfrasen innehåller en rad undantag där korrekta fraser innehåller lokal inkongruens t.ex. e*tt överklassens attribut*, där *ett* inte kongruerar med *överklassen*, dock med *attribut* som det i själva verket syftar på. För att kunna avgöra att denna konstruktion får innehålla lokal inkongruens, måste programmet kunna avgöra att det rör sig om en komplex nominalfras. Konstruktioner som *ett överklassens attribut* hanteras dock av accepterande regler och ett framtida mål bör vara att bygga ihop accepterande regler, hjälpregler och granskningsregler för att få fram en bättre grundläggande analys.

För att ge en överblick över de nominalfraser (NP) som identifieras med hjälpregler och kan anropas av de accepterande reglerna och granskningsreglerna presenterar jag ett urval av dessa i tre tabeller nedan. Hjälpreglerna kan också användas av andra hjälpregler.

[10] Språklig redigering handlar inte om att korrigera språkliga fel i texten, utan att hjälpa användaren att redigera texten utifrån språkliga grunder, t.ex. att byta tempus på verb eller bestämdhet på nominalfraser, antingen lokalt eller globalt i texten.

Klassificering	Exempel
Minimal NP med artikel, adjektiv och substantiv	*en liten bil*
Minimal NP med possessiv	*min lilla båt*
Minimal NP med egennamn	Wayne Gretzky
Minimal NP med pronomen	*Han*
NP utan substantiv eller egennamn	***unga*** *som bor på landet* ***De gröna*** *kämpar för att ta många platser i parlamentet*

Tabell 2.2. Enkla nominalfraser

Klassificering	Exempel
NP med måttsapposition	*en massa hus, en grupp studenter*
NP med NP-apposition	*min vän generalen, Staden Kalmar*
NP med konjunktion	*små och stora båtar*

Tabell 2.3. Sammansatta nominalfraser

Klassificering	Exempel
NP med relativ bisats som efterställt attribut	*Den lagstiftning som nu införs. Den avhandling vilken Jonsson försvarade*
NP med prepositionsfras som efterställt attribut	*mannen med hatten*

Tabell 2.4. Nominalfraser med efterställda attribut

En preliminär uppsättning av dessa hjälpregler (dock ej hjälpregler med efterställda attribut) utvärderades av Johansson (2000). Johansson utförde en relativ liten utvärdering, men fann att såväl precision och täckning hamnade ungefär på 80 %.

Det finns även hjälpregler för andra frastyper som har utvecklats för att alla typer av regler ska förenklas och förbättras. En särskild uppsättning hjälpregler har skapats för identifiering av satsgränser i meningar. Satsgränsreglerna bygger på Ejerheds satsigenkänningsalgoritm (Ejerhed, 1999). Satsgränsreglerna är i nuläget implementerade som vanliga hjälpregler, men man bör överväga om inte dessa bör implementeras som en förprocess, eftersom nästan alla granskningsregler kan bli bättre genom att endast appliceras på satser. En mängd falska alarm har på detta sätt eliminerats.