

SATSLÖSNING I EN LEXIKONORIENTERAD PARSER FÖR SVENSKA

Mats Dahllöf

Februari 1989¹

1 Inledning: bakgrund och syfte

Det arbete som här framläggs är utfört inom projektet en Lexikonorienterad Parser för Svenska (LPS) (Sågvall Hein (1987a)), som bedrivs på Institutionen för språkvetenskaplig databehandling vid Göteborgs universitet. Projektets syfte är att utveckla en kraftfull parser för svenska. Den skall utföra satslösning av traditionell typ, d.v.s. redovisa den grammatiska funktionsstrukturen i de analyserade språkliga uttrycken. Icke grammatiska konstruktioner skall underkännas av parsern. Eftersom målet är att den resulterande parsern skall ha en bred täckning, betonas behovet av ett omfattande och välstrukturerat lexikon, därav projektets namn. Både grammatik och lexikon skall främst inriktas mot sakprosa. Parserns tillämpningsområde skall i första hand vara informationssökning i fri text, men de resultat som uppnås i samband med arbetet kring parsern har naturligtvis relevans även för andra datalingsvistiska tillämpningsområden.

Det föreliggande arbetet har bestått i utveckling av LPS-parserns grammatik och har resulterat i en parser för ett fragment av svenska. Denna parser, som omfattar ett lexikon av än så länge ringa omfattning, kan uppfattas som ett delmål på vägen mot den slutgiltiga LPS-parsern. Den föreliggande parsern — låt oss kalla den LPS-0 — har främst utvecklats för att kunna ta hand om olika typer av satskonstruktioner i svenskan. Övriga frastyper kan endast hanteras i begränsad omfattning. I projektets inledningsskede bedömdes det som lämpligt att utnyttja Uppsala Chart Processor (UCP) vid konstruktionen av LPS-parsern. Detta innebär att denna parser, liksom LPS-0, kommer att bestå av två komponenter, dels ett parsingmaskineri, som består av de rutiner som utför själva analysen och som i detta fall alltså utgörs av UCP, dels en språkbeskrivning, som utgör den språkliga kompetens som parsern besitter. UCP kan uppfattas som en interpretator till språkbeskrivningen. Denna, som består av grammatiska regler och en lexikal komponent måste därför vara formulerad i det

¹ Detta examensarbete på datalingsvistlinjen (Göteborgs universitet) handledes av Anna Sågvall-Hein. Denna version färdigställd i februari 1997. Nästan bara typografin är något modifierad.

språk i vilket en språkbeskrivning till UCP formuleras, d.v.s. i den procedurella UCP-formalismen.

Vid utvecklingen av LPS-0 har, förutom korrekthet hos de genererade analyserna, utbyggbarhet, klarhet och effektivitet eftersträvats. Dessa egenskaper har avgörande betydelse för en parsers värde. Vad gäller LPS-0 är naturligtvis dess utbyggbarhet speciellt viktig, eftersom den är avsedd att vara ett förstadium till en större parser. För att en parser skall vara utbyggbar måste de regler och tekniker dess språkbeskrivning innefattar vara användbara och korrekta även för större språkfragment. Eller, med andra ord, en parser är utbyggbar om man inte behöver skriva om stora delar av dess språkbeskrivning när man utvidgar den till att omfatta ett större språkfragment. Maximalt utbyggbar är en parser om man enbart genom att lägga till nya regler till dess språkbeskrivning kan utvidga dess kompetens till att omfatta ett helt språk. När en parser skall vidareutvecklas är en annan egenskap hos dess språkbeskrivning avgörande: den klarhet med vilken den är formulerad. Klarheten är naturligtvis viktig även vid utarbetandet av språkbeskrivningen. För att uppnå den eftersträvansvärda klarheten bör man, vid formuleringen av språkbeskrivningen, undvika alla onödiga "trix" och formulera regler på ett logiskt genomskinligt sätt, i så stor utsträckning som möjligt utnyttja välkända grammatiska begrepp och termer, strukturera språkbeskrivningen på ett genomtänkt sätt etc. Dessa principer kan ibland komma i konflikt med kravet att språkbeskrivningen bör formuleras på ett sådant sätt att parsern kommer att arbeta på ett så komputationellt effektivt, d.v.s. ekonomiskt sätt som möjligt. I sådana lägen har pragmatiska överväganden fått göras. För att effektivisera parsern har i samband med utvecklingen av LPS-0 en modifiering av UCP företagits. Denna förändring, som i betydande grad snabbar upp parsingprocessen, beskrivs i avsnitt 2.1. Dessförinnan kommer en översiktlig beskrivning av UCP och UCP-formalismen att ges.

2 UCP och UCP-formalismen — en översikt

Den svenska grammatik som här skall beskrivas är, som nämnts, formulerad som en mängd regler i UCP-formalismen, och bildar, interpreterad av chartparsingsystemet Uppsala Chart Processor (UCP), en parser för det fragment av svenska den beskriver. UCP är implementerad i Common-LISP och all kommunikation med UCP sker i form av funktionsanrop i LISP. De relevanta delarna av UCP och UCP-formalismen kommer här att beskrivas i ganska koncentrerad form. Senare, när grammatiken beskrivs, kommer exempel med mer utförliga förklaringar att ges. För grundligare redogörelser av UCP hänvisas till Sågvall Hein (1987) och Carlsson (1982).

UCP arbetar med en datastruktur som brukar kallas chart. En chart kan uppfattas som en riktad graf och består av en mängd noder och en mängd bågar. Två slag av bågar finns: aktiva och inaktiva. (De förra beskriver en partiellt igenkänd konstituent och de senare en fullständigt igenkänd.) En aktiv båge i UCP kan uppfattas som en ordnad kvintuppel som består av en begynnelsenod, en slutnod, en konstituentbeskrivning i form av en hierarkisk attribut-värdematrix, en kontrollsats formulerad i UCP-formalismen och ett filteruttryck. En inaktiv båge definieras av

en ordnad trippel: en begynnelsenod, en slutnod och en konstituentbeskrivning.

Ett bearbetningssteg (task) bildas när en aktiv båge stöter på en inaktiv (d.v.s. när den förras slutnod är identisk med den senares begynnelsenod), under förutsättning att den inaktiva bågen uppfyller det villkor som uttrycks av den aktiva bågens filteruttryck. Varje bearbetningssteg skall exekveras. I väntan på detta sätts det upp på en lista. Denna benämnes agendan och behandlas, alltefter användarens önskemål, som en stack eller en kö. Under parsingprocessens förlopp exekveras bearbetningsstegen ett efter ett tills agendan tömts. Därefter avstannar processen.

När UCP får en sträng att analysera kommer systemet automatiskt, d.v.s. oberoende av den språkbeskrivning som finns inladdad i systemet, att göra två saker. För det första kommer det att starta en process som, karaktär för karaktär, lägger in analyssträngen i charten. Därvid kommer, om analyssträngen består av n karaktärer, $n + 1$ noder att läggas in i charten och n bågar. Mellan första och andra noden spänns en inaktiv båge upp som representerar den första karaktären i analyssträngen, mellan andra och tredje en som representerar den andra karaktären, o.s.v. (Detta är något förenklat.) För det andra kommer systemet att ansätta en given regel vid första noden i charten. (Vad det innebär att "ansätta en regel vid en nod" förklaras strax.) Vad som kommer att ske i charten fortsättningsvis bestäms av den språkbeskrivning systemet arbetar med. En inaktiv båge från den första till den sista noden i charten representerar en fullständig analys (parse) av inputsträngen. Ligger ingen sådan båge i charten sedan processen avstannat har grammatiken inte känt igen inputsträngen. En ambiguös sträng ger upphov till multipla analyser.

En språkbeskrivning till UCP utgörs av en grammatik, ett antal morflexikon, och tre databaser, en tecken-, en lemma- och en lexembas. Morflexikon och baser bildar språkbeskrivningens lexikala komponent. Grammatiken utgörs i sin tur av en mängd namngivna regler, vilka består av en regelkropp, i form av en UCP-sats, och två optionella filteruttryck. Ett morflexikon består av en mängd ordnade par av en lexikoningång och en UCP-sats. En bas består av en mängd av atomer till vilka ett antal attribut-värdepar knutits.

Exekveringen av ett bearbetningssteg innebär att den till den aktiva bågen knutna kontrollsatsen evalueras. En UCP-sats kan inte evalueras annat än inom ramen för ett bearbetningssteg. För en UCP-sats är sålunda tre noder tillgängliga: 1) den aktiva bågens begynnelsenod, 2) den aktiva bågens slutnod, som är identisk med den inaktiva bågens begynnelsenod, och 3) den inaktiva bågens slutnod. UCP-uttrycken opererar på de attribut-värdematriser som finns knutna till de bägge bågarna, eller, rättare sagt, deras union. Innan vi kan säga så mycket mer måste vi därför definiera vad en attribut-värdematrix (AVM) är och vad som menas med unionen mellan två AVMer.

En AVM kan uppfattas som en mängd av attribut-värdepar (ordnade par av ett attribut och ett värde). Attributen är alltid atomer² och ett värde utgörs antingen av en atom eller av en AVM. Ett attribut förekommer i maximalt ett attribut-värdepar i en AVM och cirkulariteter är förbjudna: ingen AVM får ingå i värdet

² Med *atomer* åsyftas här samma objekt som kallas atomer i Lisp. I det här sammanhanget räcker det med att notera att en atom utgörs av en sträng av tecken (bokstäver, siffror och en del annat får ingå) och representerar ett atomärt värde. Några exempel på (grafiska representationer av) atomer: TYPE, PHR.CAT, 5 *.

till något av sina attribut. Om ett attribut-värdepar (a,v) är element i en AVM A , säger vi att a är definierad i A och att värdet av a är v i A . En distinktion mellan typ och förekomst görs ifråga om AVMer. Två attribut kan alltså ha samma AVM som värde (m.a.o. har de identiska värden) eller två olika, fast ekvivalenta AVMer som värden. (De kan naturligtvis också vara helt olika.) I detta avseende skiljer sig alltså AVMer från mängder, vilka inte tillåter en sådan distinktion. Ekvivalens mellan två AVM motsvarar det mängdteoretiska identitetsbegreppet. Hur en AVM representeras grafiskt visas enklast med ett exempel:

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              HEAD = (WORD.CAT = NOUN
                      LEX = HUND.NN.1)
              SPEC = DEF)
      SUBJ = <* FUND>
      PRED = (LEX = AVLIDA.VB.1)
      TENSE = PRES
      SEP = .))
```

Attribut-värdepar skrivs som synes ut med ett likhetstecken mellan attributnamnet och värdet och en AVM representeras som en mängd sådana par inneslutna mellan parenteser. Indenteringen görs så att attributnamnen i en AVM står vänsterjusterade under varandra. En AVM som förekommer flera gånger i strukturen skrivs ut på vanligt sätt första gången den nämns, för att i fortsättningen representeras av ett väguttryck (se nedan) som anger var den fått sitt explicita uttryck. I ovanstående struktur anges exempelvis att värdet av $\langle * \text{ SUBJ} \rangle$ är identiskt med värdet av $\langle * \text{ FUND} \rangle$. Av praktiska skäl får attribut-värdeparen en viss ordning i den grafiska representationen (och även i den interna representationen) av en AVM, men denna är logiskt insignifikant. (För läsligheten har den naturligtvis betydelse.)

Givet två AVMer, låt oss kalla dem A och B , kan unionen mellan dem, $A \cup B$, definieras på följande sätt: Om ett attribut-värdepar (a,v) är element i A och a är odefinierad i B , så ingår (a,v) i $A \cup B$, och tvärtom: Om ett attribut-värdepar (a,v) är element i B och a är odefinierad i A , så ingår (a,v) i $A \cup B$. Är (a,v) element i A och (a,w) element i B kan fem fall urskiljas: 1) v och w är identiska atomer, (a,v) är då element i $A \cup B$. 2) v och w är icke-identiska atomer, A och B kan i så fall inte unifieras (d.v.s. de har ingen union). 3) v och w utgörs av en atom och en AVM (eller vice versa), A och B kan i så fall inte unifieras. 4) v och w utgörs av två unifierbara AVMer, $(a,v \cup w)$ ingår då i $A \cup B$. 5) v och w är två icke unifierbara AVMer, A och B kan i så fall inte unifieras. Kan A och B unifieras är $A \cup B$ den minsta mängd som uppfyller ovanstående villkor. Två atomer sägs vara unifierbara då de är identiska. Schieber (1986) diskuterar olika unifieringsbaserade grammatikformalismen och användningen av AVMer (av honom kallade "feature

structures”) i lingvistiska sammanhang. Det ovan sagda kan lämpligen illustreras med ett exempel. Om vi har följande två AVMer:

$$\begin{aligned} (A = (A = a \\ \quad B = b) \\ B = b) \end{aligned}$$

och

$$\begin{aligned} (A = (A = a \\ \quad C = c) \\ C = c) \end{aligned}$$

så har deras union följande utseende:

$$\begin{aligned} (A = (A = a \\ \quad B = b \\ \quad C = c) \\ B = b \\ C = c) \end{aligned}$$

Två typer av uttryck finns i UCP-formalismen: 1) termer och 2) satser. Tre slag av av termer finns 1) väguttryck, 2) konstanter och 3) konkatenat. Ett väguttryck (path expression) består av ett antal attributnamn inneslutna mellan “<” och “>”. Dessa används för att referera till värden i en AVM. $\langle X \rangle$ för något attributnamn X denoterar värdet av attributet X i den aktuella AVMen. Ett väguttryck $\langle X_1 \dots X_n \rangle$, där $n > 1$, denoterar värdet av $\langle X_2 \dots X_n \rangle$ i den AVM som $\langle X_1 \rangle$ denoterar i den aktuella AVMen. Refererar $\langle X_1 \rangle$ i detta fall till en atom rapporteras ett fel. En annan typ av väguttryck refererar till värden i de olika baserna (tecken-, lemma- eller lexem-). Dessa väguttryck har strukturen $\langle X_1 \dots X_n : A \rangle$ och är meningsfulla endast om $\langle X_1 \dots X_n \rangle$ denoterar ett atomärt värde, låt oss kalla det v . I så fall denoterar $\langle X_1 \dots X_n : A \rangle$ det värde som någon av baserna knyter till attributet A hos v . En konstant utgörs av en (lisp-)atom föregången av “” (quote) och denoterar ett atomärt värde (sig själv). Ett konkatenat är en följd av juxtaponerade termer vilka denoterar atomära värden, hela uttrycket denoterar den atom som bildas genom att dessa konkateneras.

I de aktiva bågarnas konstituentbeskrivning ligger all information som värde av $\langle \& \rangle$ och de inaktiva bågarnas som värde av $\langle * \rangle$. Dessa konstituentbeskrivningar är tillgängliga då exekveringen av ett bearbetningssteg begynner.

En sats i UCP-formalismen returnerar ett sanningsvärde då det evalueras. I många fall är dock det väsentliga de sidoeffekter som uppstår vid evalueringen. Syntax och semantik hos de relevanta satstyperna kommer här att beskrivas. Följande metasymboler kommer därvid att användas (eventuellt med index): S betecknar en godtycklig sats. P betecknar ett godtyckligt väguttryck. X , liksom Y , betecknar ett godtyckligt väguttryck eller annan term. R betecknar ett godtyckligt regelnamn. D betecknar ett godtyckligt lexikonnamn. F betecknar ett godtyckligt filteruttryck.

“not S ” (negation) returnerar sant värde om S returnerar falskt, och vice versa.

“ S_1, \dots, S_n ” (konjunktion) evalueras tills någon av konjunkterna returnerar falskt värde, varvid konjunktionen returnerar falskt värde och de återstående konjunkterna förblir oevaluerade. Returnerar alla konjunkterna sant värde, så returnerar hela konjunktionen sant värde.

“ $S_1 / \dots / S_n$ ” (dependent disjunktion) evalueras tills någon av disjunkterna returnerar sant värde, varvid hela disjunktionen returnerar sant värde, och de återstående disjunkterna förblir oevaluerade. Returnerar alla disjunkterna falskt värde, så returnerar hela disjunktionen falskt värde.

“ $S_1 // \dots // S_n$ ” (independent disjunktion) evalueras på så sätt att för varje S_i , $1 \leq i \leq n$, bildas ett nytt bearbetningssteg, identiskt med det som just exekveras förutom det att den aktuella disjunktionen ersatts med S_i i den aktiva bågens kontrollrats. Varje bearbetningssteg som bildats på detta sätt läggs på agendan för att i sin tur exekveras i vanlig ordning. Exekveringen av det bearbetningssteg i vilket disjunktionen uppträdde avbryts därefter. Den independenta disjunktionen används alltså för att få en process att dela upp sig i flera parallella processer.

“if S then S_1 else S_2 ” returnerar vid evaluering värdet av S_1 om S returnerar sant värde, annars returneras värdet av S_2 .

För att de ovanstående logiska operatorerna ska binda samman satserna på rätt sätt kan parenteser användas.

“continue” och “false” är de båda sanningsvärdeskonstanterna, och returnerar sant, respektive falskt värde.

“ $X = Y$ ” returnerar sant värde om och endast om X och Y är ekvivalenta.

“ $X := Y$ ” returnerar sant värde om och endast om X och Y kan unifieras, eller om en av operanderna är odefinierad. Som en sidoeffekt ändras värdena av X och Y så att de efter operationen är identiska med $X \cup Y$, respektive den definierade operanden. (Råkar båda operanderna vara odefinierade får de värdet NIL efter unifieringen.) Unifieringsoperatoren “:=” kan alltså fungera både som predikat och tilldelningsoperator. Den är den enda operator som kan förändra en AVM.

“ P ”, bildar en sats som vid evaluering returnerar sant värde om och endast om denotationen hos P är definierad.

“ R ” bildar en sats som vid evaluering substitueras mot regeln R 's regelkropp, vilken sedan evalueras. Regeln sägs därvid bli aktiverad.

“ $R(X_1, \dots, X_n)$ ” bildar en sats som vid evaluering aktiverar regeln R . De formella parametrarna i regelkroppen till R substitueras mot de aktuella parametrarna X_1, \dots, X_n .

“do(P)” är en meningsfull sats endast under förutsättning att P denoterar en atom som är namnet på en regel. Denna aktiveras genom satsens evaluering.

“advance” åstadkommer vid evaluering att en aktiv båge som spänner från bearbetningsstegets aktiva båges begynnelseknod till den inaktiva bågens slutknod läggs in i charten. Denna båge ärver den “gamla” aktiva bågens konstituentbeskrivning samt filteruttryck och dess kontrollrats utgörs av det som finns kvar att evaluera av den “gamla” aktiva bågens kontrollrats. Därefter avslutas det aktuella bearbetningssteget.

“advance(R)” är nästan ekvivalent med “advance, R ”. Den enda skillnaden

är att i det första fallet kommer den nya aktiva bågen inte att ärva sitt filteruttryck från den gamla, utan få sitt filteruttryck från regeln R .

“`process(R)`” åstadkommer vid evaluering att en aktiv båge från och till den inaktiva bågens begynnelseod insätts i charten. Denna nya båge har en tom konstituentbeskrivning, R som kontrollats och samma filteruttryck som regeln R .

“`process(R,P)`” har nästan samma semantik som “`process(R)`”. Den enda skillnaden är att i den nya aktiva bågens konstituentbeskrivning kommer `<&>` att få samma värde som R har i det aktuella bearbetningssteget.

“`majorprocess(R)`” och “`majorprocess(R,P)`” uppför sig som “`process(R)`” respektive “`process(R,P)`”, med den skillnaden att den nya aktiva bågen insätts i charten från och till den gamla aktiva bågens begynnelseod. Vi säger i alla fyra fallen att regeln R ansätts vid den aktuella noden.

“`process(D)`” och “`majorprocess(D)`” insätter i charten en lexikonsökningsbåge för lexikonet D från och till den aktiva nodens slutnod, respektive begynnelseod.

“`store`” lägger in en inaktiv båge i charten från den aktiva bågens begynnelseod till den gamla inaktiva bågens slutnod. I den nya bågens konstituentbeskrivning får `<*>` värdet av `<&>` i det aktuella bearbetningssteget.

“`store(P)`” är som “`store`”. Enda skillnaden är att `<*>` i den nya bågen får samma värde som P i det aktuella bearbetningssteget.

“`minorstore`” och “`minorstore(R)`” liknar “`store`” respektive “`store(R)`”. Enda skillnaden är att den nya inaktiva bågen får samma spännvidd som den aktiva bågen.

En grammatik utgörs av en mängd regler. Dessa består av ett regelnamn, en regelkropp och två optionella filteruttryck. Det första kallas det inaktiva filtret och det andra det aktiva filtret. Det inaktiva filtret konsulteras när regeln ges som argument åt UCP-operatorerna `process`, `majorprocess`, och `advance`, på de sätt som tidigare beskrivits. Aktiva filter används mer sällan, i föreliggande grammatik inte alls, och kommer här att förbigås. En regel definieras i UCP med hjälp av ett LISP-uttryck av följande form:

```
(define G-entry R #u S; {#! F1; {#! F2;}})
```

Där G är namnet på den grammatik regeln skall in i, R regelnamnet, S regelkroppen, F_1 och F_2 det inaktiva respektive det aktiva filtret. Fakultativa element är placerade inom klamrar.

Syntax och semantik för filteruttrycken är okomplicerade. Ett attributnamn är ett filteruttryck som är sant om attributet i fråga är definierat i värdet av `<*>` (gäller endast inaktiva filter!). Två filteruttryck kan sättas samman till ett med `and` och `or`. Ett filteruttryck kan negeras med `not`. Parenteser får användas för att uttrycka hur dessa konnektiver ska binda samman uttrycken. `and`, `or`, och `not` har samma semantik som i satslogiken.

Ett morfleksikon består av en mängd lexikoningångar (karaktärsträngar) till vilka UCP-satser knutits. Sökningen i ett lexikon, som här beskrivs mycket förenklat, startas med hjälp av `process` eller `majorprocess`, så som tidigare beskrivits. Den sålunda insatta lexikonsökningsbågen, ett slags aktiv båge, kommer att bilda bearbetningssteg med de karaktärbågar den möter och om de hittills ingenkända karaktärerna tillsammans med den inaktiva bågens karaktär bildar början på en lexikon-

ingång avancerar lexikonsökningsbågen. Om detta inte är fallet avbryts naturligtvis sökningen. När lexikonsökningsbågen funnit den sista karaktären i en lexikoningång kommer den UCP-sats som finns knuten till denna sträng att aktiveras. En lexikoningång läggs in i ett lexikon med följande funktionsanrop i LISP.

```
(define D-entry E #u S;)
```

som leder till att lexikoningången E läggs in i lexikonet D och att UCP-satsen S knyts till E .

De tre baserna, tecken-, lemma-, och lexembasen, används för lagring av sådan information som kan knytas till ett enskilt tecken, lemma respektive lexem. Genom att denna information lagras separat avlastas konstituentbeskrivningarna. Informationen knyts till en symbol (för ett tecken, lemma eller lexem) i form av en mängd attribut-värdepar. Detta görs med någon av följande typer av funktionsanrop:

```
(define character s (a1 v1) ... (an vn))
```

```
(define lemma s (a1 v1) ... (an vn))
```

```
(define lexeme s (a1 v1) ... (an vn))
```

Detta leder till att attribut-värdeparen $(a_1 v_1) \dots (a_n v_n)$ knyts till symbolen s . Väguttryck används för att referera till attributvärden i baserna, enligt ovan.

2.1 En metod att effektivisera parsingprocessen i UCP

Förutsättningen för att ett chartparsingsystem skall vara helt indeterministiskt är att bågar i charten inte förändras under parsingprocessens förlopp. Den omfattande unifiering av konstituentbeskrivningar som normalt äger rum när UCP analyserar en sträng får alltså inte förändra någon av de bågar som finns inlagda i charten. Därför måste varje bearbetningssteg arbeta med kopior av de båda bågaras konstituentbeskrivningar. Då ett bearbetningssteg plockas från agendan för att exekveras måste därför kopior tas av dem. I de hittillsvarande versionerna av UCP har det varit fråga om en fullständig och helt strukturbevarande kopiering, vilket är en synnerligen resurskrävande procedur. Denna typ av kopiering är dock nödvändig om unifieringen används utan restriktioner i de grammatiker man använder.

En mindre fullständig kopiering kan dock göras, vilket leder till att UCP blir betydligt snabbare. Priset för en sådan förändring är att vissa restriktioner måste läggas på de slag av unifieringar som får ske. Detta leder till att grammatikerna måste skrivas i en "stil" som avviker en del från "stilen" hos de grammatiker som tidigare skrivits för UCP.

Sålunda har en procedur som utför en form av begränsad kopiering implementerats. Denna har utformats så att de restriktioner som måste läggas på grammatiken blir rimliga ur lingvistisk synpunkt. Enbart de översta nivåerna i konstituentbeskrivningarna kopieras³. Övriga delar förblir identiska i original och kopia. I den aktiva

³ Någoting bör kanske här sägas om implementationen av dessa. AVMer är implementerade som listor i Common Lisp. Att två konstituentbeskrivningar är identiska innebär att de representeras av en och samma lista. (Deras representationer är alltså eq.) Om C är en kopia av en AVM A , så är den underliggande representationen av C equal, men inte eq, med den struktur som representerar A .

bågen kopieras de två översta nivåerna. Värdet av $\langle \& X Y \rangle$, där X och Y är två godtyckliga attribut, blir således identiskt i original och kopia, däremot inte värdet av $\langle \& X \rangle$ eller $\langle \& \rangle$. I den inaktiva bågen kopieras översta nivån. Således blir värdet av $\langle * X \rangle$ identiskt i original och kopia, däremot inte värdet av $\langle * \rangle$. För att inte indeterminismen skall gå förlorad får alltså inte värdet av $\langle \& X Y \rangle$ eller $\langle * X \rangle$ modifieras. Låt oss som exempel anta att ett bearbetningssteg med följande konstituentbeskrivningar plockas från agendan.

```

(& = (GD = (PHR.CAT = CL
            TYPE = MAIN
            MODE = DECL
            FUND = (PHR.CAT = NP
                    GENDER = UTR
                    NUMB = SING
                    HEAD = (WORD.CAT = NOUN
                            LEX = RABIES.NN.1)
                    SPEC = INDEF)
            SUBJ = (PHR.CAT = NP
                    GENDER = UTR
                    NUMB = SING
                    HEAD = (WORD.CAT = NOUN
                            LEX = HUND.NN.1)
                    SPEC = DEF))
    REG = (PVA = <& GD FUND>
           V1.LEM = AVLIDA.VB
           V1.TENSE = PRET))
* = (PHR.CAT = PP
     PREP = (LEX = I.PP.0)))

```

Kopian av denna struktur har samma grafiska representation som originalet. Ovanstående figur fungerar därför som bild av båda två. Värdet av $\langle \& \rangle$, en AVM med attributen GD och REG, är icke-identiskt i original och kopia. Inte heller värdet av $\langle \& GD \rangle$, som är en AVM med attributen PHR.CAT, TYPE, MODE, FUND o.s.v., är identiskt i original och kopia. Samma sak gäller värdet av $\langle \& REG \rangle$. Däremot är värdet av exempelvis $\langle \& GD FUND \rangle$, med attributen PHR.CAT, FORM, GENDER o.s.v., identiskt i original och kopia. Värdet av $\langle * \rangle$, vari attributen PHR.CAT och PREP ingår, är inte identiskt i original och kopia. Av dessa anledningar går det bra att lägga till och definiera nya attribut i de AVM som utgör värden till $\langle \& GD \rangle$, $\langle \& REG \rangle$ och $\langle * \rangle$. Sådana operationer får inga konsekvenser utanför bearbetningssteget, d.v.s. de kommer inte att förändra någon båge som är inlagd i charten. Däremot får värdet av $\langle \& GD FUND \rangle$ på inga villkor modifieras, eftersom detta skulle få omfattande återverkningar i charten: Den inaktiva bågen skulle förändras. Alla bågar som ärvt sin konstituentbeskrivning från denna skulle också förändras, liksom den inaktiva båge från vilken värdet av $\langle \& GD FUND \rangle$ kommer, och följaktligen alla bågar i vilka dess konstituentbeskrivning inlemmats. Det är alltså viktigt att inga sådana operationer görs. Detta garanteras om grammatikskrivaren följer de regler

för unifieringsoperatorns användning som skall redovisas nedan. I de experiment jag utfört visar sig den versionen av UCP där denna begränsade form av kopiering införts vara betydligt, omkring 4–5 gånger, snabbare än den tidigare versionen (tiden för garbage-collection borträknad). Effektivitetshöjningen är ett gott argument för det restriktiva användande av UCP-formalismen som här skall beskrivas, men även ett lingvistiskt argument ges nedan.

2.1.1 Regler för unifieringsoperatorns användning

Förutom att garantera att ovanstående restriktioner följs kommer de principer som här skall beskrivas att definiera en enhetlig stil i vilken grammatiker kan skrivas för UCP. I redogörelsen av dessa kommer jag att tala om primära, sekundära, och tertiära attribut. Primära är attribut på den översta nivån i konstituentbeskrivningarna, d.v.s. främst * och &. Ett sekundärt attribut är ett attribut i en AVM som utgör värdet av ett primärt attribut, och ett tertiärt attribut är ett attribut i en AVM som utgör värdet av ett sekundärt attribut.

Endast tre primära attribut får användas: Förutom de självklara * och &, får PROP förekomma. Detta attribut används endast när en konstituentbeskrivning skall propageras som argument till process eller minorprocess. Hur detta går till kommer att beskrivas längre fram.

Endast två sekundära attribut får förekomma under <&>: GD och REG. Värdet av <& GD> är den grammatiska beskrivning som håller på att byggas upp. <& REG> används som ett register där man kan ”stoppa undan” konstituenters vars funktion inte kan avgöras förrän senare och där man kan ha diverse flaggor etc. Fördelen med detta är att man slipper överlasta den grammatiska beskrivningen med information som enbart är av intern relevans för parsingprocessen, och som man inte vill ha med i de färdiga grammatiska beskrivningarna.

När en konstituent är fullbordad lagras <& GD> (med `store` eller `minorstore`). De tertiära attributen under <& GD> blir därmed sekundära attribut under <*>. Dessa attribut kommer fortsättningsvis att kallas grammatiska attribut. Grammatiska attribut betecknar dels delkonstituent, som namnges av de grammatiska funktioner de har, och då och endast då utgörs deras värden av en AVM, dels drag hos den aktuella konstituenten. Deras värden är då atomära. Regeln är alltså den att en AVM alltid svarar mot en konstituent, och vice versa. Denna regel definierar ett ur lingvistisk synpunkt tilltalande sätt att organisera konstituentbeskrivningarna, eftersom strukturen hos dessa kommer att motsvara strukturen hos de konstituent de beskriver.

Unifieringsoperatorn används endast för att tilldela värden till nya grammatiska attribut i den aktiva och inaktiva bågens konstituentbeskrivningar, samt för att definiera nya attribut i registret. Om X är ett godtyckligt attribut så är det alltså <& GD X >, <* X > och <& REG X > som får tilldelas värden. Den odefinierade operanden placeras till vänster. Ex. <& GD NUMB>:=: 'SING. Detta innebär att det är tillåtet att lägga till ett drag eller en delkonstituent i den beskrivning som håller på att byggas upp, vilket är en av de viktigaste operationerna i en parser. Dessutom är det tillåtet att göra ett tillägg i den konstituentbeskrivning som finns knuten till den inaktiva bågen. En fullständigt igenkänd konstituent kan alltså kom-

pletteras i efterhand. Denna möjlighet utnyttjas vid diskontinuiteter för att hantera fall där en delkonstituent flyttats ut ur en konstituent, exempelvis vid analysen av diskontinuerliga prepositionsfraser. Det faktum att alla andra modifieringar av konstituentbeskrivningarna är otillåtna utgör en begränsning av grammatikernas uttryckskraft.

Att modifiera de grammatiker som tidigare skrivits för UCP (Sågwall Hein (1987) och Sågwall Hein & Ahrenberg (1985)) att överensstämna med dessa principer leder i de flesta fall inte till några betydande förändringar. Några undantag bör nämnas: Vissa drag hos konstituenterna har tidigare samlats under ett attribut (NOUN.FEAT, VERB.FEAT, CL.FEAT, etc.) och dessa har, exempelvis för att garantera kongruens, unifierats i ett svep. Detta får inte längre göras, utan procedurer måste definieras där de aktuella attributen jämförs var för sig och tilldelningar görs explicit. Det speciella attribut CHARS under vilket attributen 1,2 o.s.v. (bildade med :NEW) placeras kan inte längre finnas. De numeriska attributen placeras istället själva som sekundära attribut (under *). Nackdelen med min "reform" är att möjligheten att gruppera ihop sammanhörande attribut genom att placera dem under ett överordnat attribut går förlorad. Detta kan dock avhjälpas genom att utskriftsrutinen utrustas med en procedur som ordnar attributen efter användarens önskemål.

Den grammatik som här ska redovisas är skriven i enlighet med de ovan beskrivna principerna. Redogörelsen för denna kommer därför att innehålla åtskilliga exempel på vilken inverkan dessa har på grammatikens utformning.

3 Språkbeskrivningen till LPS-0

3.1 Allmänna principer

Den föreliggande grammatiken tilldelar språkliga uttryck analyser i enlighet med en modell som i sig förenar den traditionella grammatikens principer och kategorier med en valensgrammatiskt orienterad begreppsapparat. De grammatiska beskrivningar som byggs upp redovisar den funktionella strukturen i de aktuella uttrycken utan att frasstrukturen anges explicit. Ett minimalt krav är dock att alla de (kommunikativt) relevanta aspekterna av frasstrukturen hos det analyserade uttrycket på något sätt avspeglas i den grammatisk beskrivningen.

Eftersom LPS-0 är implementerad i UCP kommer de analyser (grammatiska beskrivningar) av ett språkligt uttryck (tecken, morf, ord, eller fras) som parsern genererar att redovisas som en hierarkisk attribut-värdematrix. I LPS-0's analyser förekommer två slag av attribut. Den första typen svarar mot grammatiska drag. Värdena hos dessa är alltid atomära. Exempelvis anges att draget numerus har värdet singularis med NUMB=SING. Ett attribut av det andra slaget namnges av en grammatisk funktion och har som värde en beskrivning av den delkonstituent som innehar denna funktion i frasen. Sålunda har attributet SUBJ (som förekommer i beskrivningar av satser) som värde en beskrivning av subjektet.

Två typer av lexikala enheter urskiljs i den grammatiska beskrivningen, dels lemman, dels lexem. Denna indelningen är övertagen från Svensk Ordbok (1986), som tjänar som grund för den lexikala beskrivningen i den föreliggande parsern. Ett

lemma definieras där som “en grupp ordformer som tillhör samma ordklass, ingår i samma böjningsmönster (paradigm) och har (så långt möjligt) samma uttal” (s. XII). I ordboken urskiljs en eller flera kärnbetydelser⁴ hos varje lemma. För varje sådan upprättas ett lexem. Till varje lemma kommer sålunda ett eller flera lexem att knytas. Orden uppdelas alltså primärt på formella grunder i lemman, och dessa i sin tur uppdelas utifrån semantiska kriterier i lexem. Lemman betecknas i den föreliggande modellen med tvådelade namn, där första delen utgörs av grundformen och den andra av en beteckning för ordklassen. Dessa båda delar skiljs åt av en punkt. Ordklassbeteckningen består av två tecken, normalt den första och den sista bokstaven i det engelska ord som benämner den aktuella ordklassen. Sålunda har vi:

NN	för	substantiv
PM	för	proprier
AV	för	adjektiv
PN	för	pronomen
RP	för	relativpronomen
NL	för	räkneord
VB	för	verb
AB	för	adverb
CN	för	konjunktioner
SN	för	subjunktioner
PL	för	verbpartiklar
PP	för	prepositioner
IN	för	interjektioner

Följaktligen utgör MAN.NN, FINLAND.PM, GRÖN.AV, HAN.PN, VILKEN.RP, TIO.NL, ANTYDA.VB, OFTA.AB, OCH.CN, ATT.SN, AV.PL, PÅ.PP, och USCH.IN exempel på olika lemmabeteckningar. De lexem som finns knutna till ett lemma är numrerade från ett och uppåt. Namnet på ett lexem erhålles genom att en punkt följd av dess nummer adderas till namnet på det lemma det tillhör. Exempelvis är ANTYDA.VB.1 och ANTYDA.VB.2 namnen på de två lexem som finns knutna till lemmat ANTYDA.VB.

Språkbeskrivningen är grundad på valensteoretiska principer. Det innebär att vissa led i en fras betraktas som argument till huvudordet i denna och att det är lexikaliskt styrt vilka argumentstrukturer som tillåts. Dessa led sägs vara valensbundna till huvudordet. Den regel som säger vilka dessa får vara kallas för huvudordets valensram. De övriga leden betraktas som underordnade hela komplexet huvudord-argument, och deras närvaro som reglerad av allmänna (icke-lexikala, icke-specifika) grammatiska principer. I satsen, exempelvis, anses verbet vara huvudord. Subjekt, olika typer av objekt etc. räknas som argument till verbet, medan tids-, rumsadverbial o.a. betraktas som tillhöriga hela satsen, vars kärna utgörs av

⁴ Begreppet *kärnbetydelse* ges ingen definition i Svensk Ordbok. Uppenbarligen tänker man sig att ett ord (lemma) kan användas dels i någon av sina bokstavliga, egentliga, lexikalt givna betydelser, eller m.a.o. i någon av sina kärnbetydelser, dels i betydelser som på något sätt (“överfört”, “utvidgat”, “speciellt”, eller “bildligt”) är härledda ur kärnbetydelser hos ordet. Kärnbetydelser är alltså synnerligen intuitiva entiteter. De därmed förknippade teoretiska och praktiska svårigheterna är givetvis stora.

verbet och dess valensbundna led. Denna dualistiska åtskillnad mellan valensbundna och icke valensbundna led är i längden svår att upprätthålla och ersätts därför i många valensgrammatiska system, dock ej här, av ett mer skalärt betraktelsesätt, enligt vilket de olika leden klassificeras efter hur hårt knutna de är till huvudordet. Somers (1987), för att ta ett exempel, urskiljer sex olika grader av bundenhet till verbet. En rik litteratur behandlar de teoretiska frågorna kring valensgrammatiker. Se exempelvis Somers (1987) och Toporowska-Gronostaj (1987). Här kommer de inte att behandlas närmre.

I den föreliggande grammatiken har ett slags valensramar för verb (verbramar) implementerats. Dessa spelar, som vi skall se, en central roll vid analysen av olika typer av satser.

Eftersom de olika lexemen inom ett och samma lemma mycket väl kan ha olika valens, knyts valensinformationen till lexemen och inte till lemmat. Eftersom denna information behövs vid analysen av fraser, och därvid bidrar till att fastställa huvudordets lexemtillhörighet, anges alltid lexemtillhörigheten för de i en fras ingående lexikala enheterna.

3.2 Metod

Språkbeskrivningen till LPS-0 har utarbetats enligt en tämligen informell metod, som torde vara vanlig i grammatiska sammanhang. Den ingående lingvistiska kunskapen har framförallt hämtats ur två källor, av vilka den första är redan formulerade grammatikor, främst Thorells (1973) Svensk grammatik, och den andra är författarens egna språkliga intuition. Vid språkbeskrivningens verifiering, d.v.s. kontrollen av att den förmår analysera de konstruktioner för vilka den är konstruerad och att den tilldelar dessa de avsedda analyserna, har den givna metoden varit testkörningar.

3.3 Några anmärkningar av mer teknisk art

Om ett blanktecken följer efter ett ord (vilket i de flesta fall är fallet) så spänns den inaktiva båge som representerar analysen av det aktuella ordet över detta blanktecken, så att detta s.a.s. inkorporeras i analysen. En ordbåge kommer således att gå från det analyserade ordets begynnelse till det nästföljande ordets begynnelse. Bågar som representerar skiljetecken följer också denna princip. Förfarandet har den uppenbara fördelen att man vid formuleringen av de syntaktiska reglerna inte behöver ta hänsyn till närvaron av blanktecken.

Den språkliga analysprocessen initieras i UCP genom att regeln `start.rule` ansätts vid första noden i charten. I LPS-0's språkbeskrivning har denna regel följande definition:

```
(define sve.gram-entry start.rule
  #u process(sve.dic),
      process(cl.decl),
      process(cl.imp),
      process(cl.interr);
  #! CHAR;)
```

`start.rule` initierar en lexikonsökning i stamlexikonet `sve.dic` samt ansätter de tre regler som analyserar huvudsatser i deklarativt, imperativt respektive interrogativt modus.

3.4 Analysen av andra kategorier än satser

Denna uppsats är avsedd att främst redogöra för hur analysen av några olika sats-typer i svenskan kan gå till. Modulariteten i en UCP-grammatik tillåter oss att bortse från hur analysen av ord och andra frastyper utförs. Däremot är naturligtvis innehållet i de konstituentbeskrivningar som byggs upp för dessa kategorier i många fall relevant vid analysen av satser. Några av dem kommer därför kortfattat beskrivas.

För analysen av verben använder sig föreliggande grammatik av det system som utvecklats av Sågwall Hein (1988). Följande drag förekommer alltid i verbbeskrivningarna: `WORD.CAT`, som har värdet `VERB`, och `LEM`, som anger vilket lemma vi har att göra med. Draget `TENSE` anger vilket tempus verbet har. Vid presens är dess värde `PRES`, vid preteritum `PRET`, och om verbet är en supinumform `SUP`. Verb i imperativform får särdraget `IMP=+` i sina beskrivningar och de i infinitiv får särdraget `INF=+`. Att ett verb är i passivum anges likaledes med ett särdrag `PASS=+`. Dragen `TENSE`, `IMP`, `INF` och `PASS` är i icke angivna fall odefinierade. Följande verbanalyser exemplifierar detta:

```
BOR :  
  (* = (WORD.CAT = VERB  
        LEM = BO.VB  
        TENSE = PRES))
```

```
BOTT :  
  (* = (WORD.CAT = VERB  
        LEM = BO.VB  
        TENSE = SUP))
```

```
SJUNG :  
  (* = (WORD.CAT = VERB  
        LEM = SJUNGA.VB  
        IMP = +))
```

Nominalfrasreglerna i den föreliggande grammatiken är mycket ofullständiga. De utgör ett urval ur en nominalfrasgrammatik som utvecklats av Anna Sågwall Hein. I nominalfrasbeskrivningarna anges att `PHR.CAT=NP`. Attributet `HEAD` har en beskrivning av huvudordet som värde. I denna ingår dragen `WORD.CAT` och `LEX`, som anger huvudordets ordklass- och lexemtillhörighet. I beskrivningen av nominalfrasens determinerare, som placeras under attributet `DETER`, anges endast lexemtillhörighet. Attributet `NUMB` (vi har återgått till nominalfrasbeskrivningarna) anger frasens numerus och kan anta värdena `SING` och `PLUR`, vars innebörder är uppenbara. Frasens species anges av draget `SPEC`, som kan anta värdena `DEF` och `INDEF`, vilka står för bestämd respektive obestämd species. Genus anges av draget `GENDER`, vars värde

är antingen NEUTR eller UTR, för neutrum respektive utrum (reale). Attributet CASE anger nominalfrasens kasus. Beskrivningarna av nominalfraser vars huvudord är ett substantiv eller ett proprium innehåller en angivelse av kasus endast om detta är genitiv, varvid CASE=GEN. I annat fall är CASE odefinierad. Är nominalfrasens huvudord ett pronomen med olika subjeks- och objektsform får CASE i det första fallet värdet NOM och i det andra värdet OBL (oblikt kasus). Nedan ges några exempel på nominalfrasbeskrivningar:

EN HUND :

```
(* = (PHR.CAT = NP
      GENDER = UTR
      NUMB = SING
      SPEC = INDEF
      DETER = (LEX = EN.AL.1)
      HEAD = (WORD.CAT = NOUN
              LEX = HUND.NN.1)))
```

HUNDARNAS :

```
(* = (PHR.CAT = NP
      CASE = GEN
      GENDER = UTR
      NUMB = PLUR
      HEAD = (WORD.CAT = NOUN
              LEX = HUND.NN.1)
      SPEC = DEF))
```

DE :

```
(* = (PHR.CAT = NP
      CASE = NOM
      NUMB = PLUR
      HEAD = (LEX = DE.PN.1
              WORD.CAT = PRON)))
```

DEM :

```
(* = (PHR.CAT = NP
      CASE = OBL
      NUMB = PLUR
      HEAD = (LEX = DE.PN.1
              WORD.CAT = PRON)))
```

Grammatiken tillåter endast enordiga adverbfraser. I deras beskrivningar ingår attributen PHR.CAT, vars värde är ADVP, och HEAD, vars värde är en beskrivning av huvudordet. I denna beskrivning ingår endast draget LEX som anger huvudordets lexemtillhörighet. Ett exempel på en adverbfrasbeskrivning ges nedan:

INTE :

```
(* = (PHR.CAT = ADVP
      HEAD = (LEX = INTE.AB.1)))
```

Prepositionsfrasers beskrivningar innehåller tre attribut. Det första är PHR.CAT som har värdet PP. De två andra anger vilka delkonstituenterna prepositionsfrasen består av. Värdet av PREP är en beskrivning av frasens preposition, vars enda attributvärdepar anger dess lemmatillhörighet. Beskrivningen av prepositionens rektion lagras under attributet RECT. En prepositionsfrasbeskrivning kan sålunda få följande utseende:

```
I HUSET :
  (* = (PHR.CAT = PP
        PREP = (LEX = I.PP.1)
        RECT = (PHR.CAT = NP
                GENDER = NEUTR
                NUMB = SING
                HEAD = (WORD.CAT = NOUN
                        LEX = HUS.NN.1)
                SPEC = DEF)))
```

Beskrivningarna av verbpartiklar, liksom beskrivningarna av subjunktioner, innehåller endast angivelser av ordkategori och lemmatillhörighet:

```
AV :
  (* = (WORD.CAT = PARTICLE
        LEM = AV.PL))
```

```
ATT :
  (* = (WORD.CAT = SUBJU
        LEM = ATT.SN))
```

Beskrivningarna av relativpronomen måste ibland, förutom angivelser av kategori och lemmatillhörighet, innehålla information om vilket genus och numerus ordet är böjt efter. Detta gäller inte *som*, utan former av *vilken*:

```
SOM :
  (* = (WORD.CAT = REL.PRON
        LEM = SOM.RP))
```

```
VILKA :
  (* = (WORD.CAT = REL.PRON
        LEM = VILKEN.RP
        NUMB = PLUR))
```

Skiljetecken (separatorer) tilldelas analyser som enbart anger kategori. En punkt har alltså denna analys:

```
(* = (SEP.CAT = .))
```

SEP.CAT (separatorkategori) anger vilken kategori av skiljetecken det är fråga om. Eftersom kategorierna i detta sammanhang (punkt, komma, frågetecken, utrops-tecken, etc.) enbart har ett element låter vi dem namnges av detta. Sålunda har kategorin punkt enbart elementet “.” och får namnet “.”.

3.5 Analysen av olika satstyper

Som satser räknas här alla fraser vars mest centrala beståndsdelar är ett verb och argument till detta verb. I svenska språket kan tre huvudtyper urskiljas, huvudsatser, bisatser och infinitivfraser. Huvudsatser förekommer i tre modus: deklarativt modus (påståenden), imperativt modus (befallningar) och interrogativt modus (frågor). Bisatserna bildar en relativt heterogen grupp av satser, medan infinitivfrasen är den minst varierande gruppen bland huvudtyperna. Den föreliggande språkbeskrivningen har utformats för att kunna ta hand om huvudsatser i alla tre modus. Vad gäller interrogativa huvudsatser kan endast ja-nej-frågor hanteras. Ifråga om bisatser har regler formulerats för två typer, dels explikativa bisatser, dels relativsatser. Grammatiken kan även ta hand om infinitivfraser. Vissa idéer om hur satser bör analyseras med hjälp av UCP, har övertagits från en språkbeskrivning av svenska som tidigare skrivits för detta chartparsingsystem kallad Sve.Ucp (Sågwall Hein (1987), Sågwall Hein & Ahrenberg (1985)). De viktigaste av dessa är användandet av diderichsenska satsscheman och analysen av fundamentet som en grammatisk funktion, samt bruket av verbaktionsregler för att fånga verbens valens.

3.5.1 Analysen av deklarativa huvudsatser

Regeln `cl.decl` analyserar svenskans deklarativa huvudsatser. Den ansätts initialt (av `start.rule`) i varje sträng som analyseras. Stommen i regeln är ett huvudsatsschema av den typ som Diderichsen (1966) angivit för danskan. Den har följande utseende:

```
(define sve.gram-entry cl.decl
  #u <& GD PHR.CAT> ::= 'CL,
  <& GD TYPE> ::= 'MAIN,
  <& GD MODE> ::= 'DECL,

  fundament,
  advance(v.fin),
  (advance(subj) // advance(subj.form)),
  adverbials1,
  do(<& REG V1.LEM :VERBCOMP>),
  do(<& GD PRED LEX :VERBACTION>),
  adverbials2,

  (if <& GD SUBJ.FORM>
    then <& GD SUBJ.PROPER>
    else continue),
  adv.in.pva,
  pva.used,
  end.of.sent,
  store*;

  #! PHR.CAT;)
```

De tre första konjunkterna i regelkroppen svarar för att dragen för fraskategori, typ respektive modus, tilldelas värden. Dessa anger att den fras som är under uppbyggnad är en deklarativ huvudsats. Därefter följer den del av regeln som utgör själva satsschemat och sålunda svarar för igenkänningen av de i satsen ingående leden. Den första frasen i satsen kallas fundamentet (och känns igen av regeln `fundament`). Därpå följer satsens finita verb (`v.fin`), och därefter subjektet, såvida inte detta är identiskt med fundamentet. Två slag av subjekt är här aktuella, dels "vanligt" subjekt, dels formellt subjekt (`det`), som alltid förekommer tillsammans med ett egentligt subjekt. (Dessa känns igen av reglerna `subj` respektive `subj.form`.) I nästa position förekommer fakultativt ett antal adverbial (`adverbials1`). När leden i det fram till denna punkt ganska konventionella satsschemat identifierats, anropas den verbkompositionsregel som finns knuten till det finita verbet. Denna har till uppgift att fullborda igenkänningen av predikatet. Är finitet ett hjälpverb, så tar verbkompositionsregeln hand om den infinita verbformen och den därpå eventuellt följande verbpartikeln. Då det finita verbet är ett huvudverb⁵, har verbkompositionsregeln enbart att ta hand om eventuell verbpartikel. Resten av predikatets argument, d.v.s. alla utom subjektet eller det formella subjektet, återfinns i de följande positionerna, möjligen med undantag av ett som står i fundamentalspositionen. Dessa identifieras av den verbaktionsregel som finns knuten till predikatsverket. Därefter, i satsens sista position, kan ett antal adverbial förekomma (vilka känns igen av `adverbials2`). När så hela analyssträngen konsumerats av satsschemat återstår några operationer. Först görs en kontroll av att det finns ett egentligt subjekt (`SUBJ.PROPER`) om det finns ett formellt subjekt. Proceduren `adv.in.fund` tilldelar fundamentet funktionen adverbial, om så är möjligt. `pva.used` kontrollerar att fundamentet tilldelats en funktion, och `end.of.sent` att satsen avslutas med ett skiljetecken.

Huvudsatsregeln `c1.decl` kan jämföras med Thorells (1973) formulering av svenskans huvudsatsschema. Han delar in huvudsatsen i tre fält: för-, mitt-, och slutfält. Förfältets enda led är fundamentet. Mittfältet innehåller tre delfält, för respektive finita verbformen, subjektet och eventuella adverbial. Slutfältet är likaledes indelat i tre delfält, ett för infinita verbformer, ett för objekt och ett för adverbial. Grovt sett motsvarar alltså vart och ett av dessa sju delfält ett subregelansrop i ovanstående regel. Skillnaderna mellan Thorells huvudsatsschema och Diderichsens danska original är obetydliga.

Fundamentsregeln är mycket enkel:

```
(define sve.gram-entry fundament
  #u  <* PHR.CAT>,
      <& GD FUND>:=:<*>,
      <& REG PVA>:=:<& GD FUND>;)
```

Det enda krav på fundamentet som här uttrycks är att det ska utgöras av en fras. Inga andra regler refererar direkt till fundamentet, utan istället till `<& REG PVA>` (PreVerbalt Argument), som här görs identisk med `<& GD FUND>`. Fördelen med detta förfarande kommer att klargöras i samband med redogörelsen för analysen av relativa bisatser.

⁵ Med huvudverb avser jag varje verbform som inte fungerar som ett hjälpverb.

Regeln `v.fin` identifierar det infinita verbet i satsen, och har följande formulering:

```
(define sve.gram-entry v.fin
  #u <* WORD.CAT>='VERB,
    (<* TENSE>='PRES / <* TENSE>='PRET),
    <& REG V1.LEM>:=:<* LEM>,
    <& REG V1.TENSE>:=:<* TENSE>,
    unif-atom(<& GD PASS>,<* PASS>);
  #! TENSE;)
```

Ett finit verb definieras som ett verb vars tempus är antingen presens eller preteritum. Återfinns ett sådant, lagras dess lemmabeteckning och tempus i registret (som `<& REG V1.LEM>` respektive `<& REG V1.TENSE>`), eftersom satsens tempus och predikat ännu inte slutgiltigt kan bestämmas. Däremot kan vi redan nu säga att om verbet är i passiv form så måste också satsen vara det. Detta anges i regeln.

Ett diderichsenskt satsschema fastställer en grundläggande ordning mellan de av sina grammatiska funktioner definierade leden i satsen. I svenskan är denna ordning identisk med ordningen mellan de olika leden i en ja-nej-fråga. Schemat för huvudsatsen kombineras med principen att i en huvudsats måste något led flyttas bort från den plats schemat anvisar, till positionen främst i satsen, fundamentet. Man säger ibland att ledet topikaliseras. Tillsammans bildar satsschemat och denna princip ett kraftfullt sätt att beskriva huvudsatser av den typ som finns i svenskt och danskt skriftspråk. I princip kan alla led utom predikatets komponenter (till vilka jag räknar verbet plus eventuellt hjälpverb och eventuell verbartikel) topikaliseras. Reglerna som identifierar dessa led kommer att få väsentligen samma logiska struktur. De består av en independent disjunktion. Den första disjunkten motsvarar fallet att ledet står i sin normalposition⁶. Uppfyller den konstituent som representeras av den inaktiva bågen de krav som ställs på det sökta ledet, så kommer det att inlemmas som det sökta ledet i den konstituentbeskrivning som är under uppbyggnad, varefter ett avancemang sker. Den andra disjunkten tar hand om fallet att det sökta ledet står i fundamentalspositionen: Om fundamentet ännu inte tilldelats någon funktion och om det uppfyller de krav som ställs på det sökta ledet, så kommer det att tilldelas den aktuella funktionen. Här används de två procedurerna `pva.free` och `assign.pva.to`. Den första kontrollerar att fundamentet (det Pre-Verbala Argumentet) inte tidigare tilldelats någon funktion och den andra tilldelar fundamentet den funktion som givits som argument till proceduren. De har följande definitioner:

```
(define sve.gram-entry pva.free
  #u not <& REG PVA.FUNC>;)
```

```
(define sve.gram-entry assign.pva.to
  #u <?1>:=:<& REG PVA>,
    <& REG PVA.FUNC>:=:'+;)
```

⁶ Jag kommer att kalla den position som huvudsatsschemat anvisar åt ett led för detta leds normalposition. Detta skall inte tolkas som att denna är ett leds vanligaste placering. Subjektet förekommer tvärtom oftast i fundamentalsposition.

I dessa regler används `<& REG PVA.FUNC>` som en flagga som när den är definierad anger att fundamentet tilldelats en funktion i satsen. Regeln för igenkänning av subjektet är ett utmärkt exempel på en regel med den uppbyggnad som ovan beskrivits:

```
(define sve.gram-entry subj
  #u (legal.subj(<*>),
      <& GD SUBJ>:=:<*>,
      advance
      // pva.free,
      legal.subj(<& REG PVA>),
      assign.pva.to(<& GD SUBJ>));
  #! PHR.CAT or WORD.CAT or SEP.CAT;)
```

Analysen av en mening som *I ett hus bor en man.*, där subjektet står i normalposition, tas alltså om hand av första disjunkten, medan den andra svarar för analysen av meningar som *En man bor i ett hus.*, där subjektet förekommer i fundamentalsposition. Regeln `legal.subj`, som logiskt sett bildar ett predikat som tar en konstituentbeskrivning som argument, definierar vilka konstitenter grammatiken godkänner som subjekt:

```
(define sve.gram-entry legal.subj
  #u <?1 PHR.CAT>='NP,
      not <?1 CASE>='GEN,
      not <?1 CASE>='OBL;)
```

Grammatiken tillåter alltså bara subjekt som utgörs av nominalfraser vilka inte står i oblikt kasus eller genitiv. Regeln för igenkänning av det formella subjektet har precis samma struktur som regeln för det “vanliga” subjektet:

```
(define sve.gram-entry subj.form
  #u (legal.subj.form(<*>),
      <& GD SUBJ.FORM>:=:<*>,
      advance
      // pva.free,
      legal.subj.form(<& REG PVA>),
      assign.pva.to(<& GD SUBJ.FORM>));
  #! PHR.CAT or WORD.CAT or SEP.CAT;)
```

Den första disjunkten i regelkroppen tar hand om meningar som *I huset bor det en man.* och den andra meningar av typen *Det bor en man i huset.*. Förutom att det formella subjektet ges namnet `SUBJ.FORM` istället för `SUBJ`, är den enda skillnaden att andra krav ställs på det formella subjektet. Dessa definieras av predikatet `legal.subj.form`.

```
(define sve.gram-entry legal.subj.form
  #u <?1 PHR.CAT>='NP,
      <?1 HEAD LEX>='DET.PN.1;)
```

Ett formellt subjekt utgörs alltså av en nominalfras vars huvudord utgörs av pronomen `DET.PN.1`. Det får endast finnas i kombination med ett egentligt subjekt. Detta villkor uttrycks i huvudsatsregeln `c1.decl`. (Se ovan!)

Efter subjektet kommer fakultativt ett antal adverbial, som tas om hand av proceduren `adverbials1`. En samlad redogörelse för analysen av adverbial kommer att ges senare (i avsnitt 3.5.7). I de närmast följande positionerna finns "resten" av predikatet. Grammatiken kan i sin nuvarande form ta hand om två fall. I det första fallet utgörs finitet av ett huvudverb, i det läget återstår det att ta hand om en eventuell verbpartikel. I det andra fallet utgörs finitet av verbet *ha*. Detta kan fungera både som huvud- och hjälpverb. I det senare fallet är satsens tempus perfekt eller pluskvamperfekt och det återstår att finna ett huvudverb i supinum, plus en eventuell verbpartikel. De regler som tar hand om dessa två verbtyper, jag kallar dem verbkompositionsregler, aktiveras genom ett variabelt subregelanrop, `do(<& REG V1.LEM :VERBCOMP>)`. I första fallet aktiveras `vc.ta`, som är den verbkompositionsregel lemmabasen knyter till alla huvudverbslemman (som ett defaultvärde hos attributet `VERBCOMP`), i det andra `vc.ha`, som endast finns knuten till lemmat `HA.VB`. Verbkompositionsregeln `vc.ta` har följande utseende:

```
(define sve.gram-entry vc.ta
  #u comp.pred(<& REG V1.LEM>),
  unif-atom(<& GD TENSE>,<& REG V1.TENSE>);)
```

Först sker alltså att anrop av regeln `comp.pred` som fullbordar igenkänningen av predikatet, som argument tar den huvudverbet, vilket i detta fall är identiskt med finitet. Därefter tilldelas huvudsatsen det tempus som verbet har. `comp.pred` definieras på detta sätt:

```
(define sve.gram-entry comp.pred
  #u if <* WORD.CAT>='PARTICLE
  then <& REG PRED.LEM>::'?1'+<* LEM>,
  assign.lex(<& GD PRED LEX>,<& REG PRED.LEM>),
  advance
  else assign.lex(<& GD PRED LEX>,'?1');)
```

Regeln testar om den inaktiva konstituenten utgörs av en verbpartikel. Om så är fallet bildas partikelverbslemmat genom att huvudverbslemmat (den formella parametern `?1`), ett plustecken och partikkellemmat konkateneras. Därvid bildas namnet på ett partikelverbslemma som lagras under `<& REG PRED.LEM>`. Med hjälp av proceduren `assign.lex` splittras processen upp så att varje lexem som finns knutet till lemmat ansätts som predikat i en egen process. Om inget lexem finns knutet till lemmanamnet, vilket innebär att ett sådant lemma inte finns i lexikonet, avstannar analysen. Detta sker om exempelvis verbet `DÖ.VB` kombineras med partikeln `PÅ.PL` till `DÖ.VB+PÅ.PL`, eftersom ett sådant partikelverb saknas i svenskan. Om den inaktiva konstituenten utgörs av någonting annat än en verbpartikel kommer huvudverbet att betraktas som fullständigt och de till huvudverbslemmat knutna lexemen kommer att parallellansättas som predikat (på samma sätt som med partikelverben, med hjälp av `assign.lex`). Den till verbet `HA.VB` knutna verbaktionsregeln har en enkel logisk struktur:

```
(define sve.gram-entry vc.ha
  #u (vc.ta // vc.aux.ha);)
```

Regelkroppen utgörs av en independent disjunktion. Den första disjunkten *vc.ta* tar hand om fallet att verbet fungerar som ett huvudverb, då det alltså hanteras som vilket huvudverb som helst. Den andra konjunkten *vc.aux.ha* står för analysen av sammansatta tempus i vilka *ha* fungerar som ett tempusbildande hjälpverb (perfekt och pluskvamperfekt). *vc.aux.ha* är formulerad på följande sätt:

```
(define sve.gram-entry vc.aux.ha
  #u <* TENSE>='SUP,
    <& REG V2.LEM>:=:<* LEM>,
    unif-atom(<& GD PASS>,<* PASS>),
    advance,
    comp.pred(<& REG V2.LEM>),
    (<& REG V1.TENSE>='PRES,
     <& GD TENSE>:=:'PERF
    / <& REG V1.TENSE>='PRET,
     <& GD TENSE>:=:'PLUPERF
    / <& REG V1.INF>='+,
     <& GD TENSE>:=:'PERF.INF);)
```

Regeln kräver alltså först ett verb i supinum. Dess lemmabeteckning lagras i registret som *<& REG V2.LEM>*. Är detta verb i passivum är också hela satsen det, vilket anges i regeln. Ett avancemang sker sedan, varefter *comp.pred* anropas med huvud verbet, d.v.s. *<& REG V2.LEM>*, som argument. Ansättningen av predikatet och hanteringen av en eventuell verbpartikel går alltså till på samma sätt som vid enkla tempus. Därefter sker ansättningen av satsens tempus. Om finitet är i presens (*har*) är satsens tempus perfekt, och om finitet är i preteritum (*hade*) är satsens tempus pluskvamperfekt. I det tredje fallet, som bara är aktuellt vid analysen av infinitivfraser, är satsens första verb i infinitiv (*ha*). Infinitivfrasens tempus är då perfekt infinitiv. (Analysen av infinitivfraser beskrivs närmre i avsnitt 3.5.6.)

När predikatet identifierats fullständigt, har av dess argument enbart ett subjekt eller ett formellt subjekt återfunnits. Det återstår alltså att finna de eventuella argument ännu inte identifierats. Dessa har sina normalpositioner i de positioner som följer. Vilka argument som måste och vilka som kan förekomma beror på det verb som utgör predikatet i satsen. Därför anropas den verbaktionsregel som finns knuten till detta verb. Denna identifierar verbets obligatoriska och fakultativa argument.

Mot varje typ av verbargument svarar en speciell regel som känner igen instanser av denna. Den föreliggande grammatiken innehåller ett dussin sådana regler, argumentidentifieringsregler skulle vi kunna kalla dem. Verbaktionsreglerna utgörs väsentligen av sekvenser av anrop till dessa regler. Före genomgången av verbaktionsreglerna är det därför lämpligt att närmre beskriva dessa argumentidentifieringsregler. Den enklaste av dessa är regeln som känner igen det egentliga subjektet:

```
(define sve.gram-entry subj.proper
```

```
#u  <& GD SUBJ.FORM>,
    <* PHR.CAT>='NP,
    <* SPEC>='INDEF,
    <& GD SUBJ.PROPER>:::<*>,
    advance;)
```

Det egentliga subjektet förekommer bara tillsammans med ett formellt subjektet. Närvaron av ett sådant kontrolleras först i regeln. Den motsatta implikationen, att det formella subjektet bara förekommer tillsammans med ett egentligt subjekt, anges i huvudsatsregeln. (Se ovan!) Som egentligt subjekt tillåts bara obestämda nominalfraser. *Det bor en man i huset.* är exempelvis grammatisk, medan *Det bor mannen i huset.* är klart ogrammatisk. Regeln tillåter inte att det egentliga subjektet topikaliseras. *En man bor det i huset.* betraktar grammatiken sålunda som ogrammatisk. Regeln för det indirekta objektet har väsentligen samma struktur som den "vanliga" subjektsregeln:

```
(define sve.gram-entry obj.indir
  #u (legal.obj(<*>),
      <& GD OBJ.INDIR>:::<*>,
      advance
    // pva.free,
      legal.obj(<& REG PVA>),
      assign.pva.to(<& GD OBJ.INDIR>));)
```

Det indirekta objektet får flyttas från sin normalposition till fundamentet, både *Han ger henne en bok.* och *Henne ger han en bok.* tillåts alltså. Som indirekt objekt tillåts endast nominalfraser vars kasus inte är genitiv eller nominativ⁷. Detta sägs i predikatet `legal.obj`:

```
(define sve.gram-entry legal.obj
  #u <?1 PHR.CAT>='NP,
      not <?1 CASE>='GEN,
      not <?1 CASE>='NOM;)
```

`legal.obj` används även för att definiera vilka fraser som kan bilda ett direkt objekt. Regeln för det direkta objektet har följande utseende:

```
(define sve.gram-entry obj.dir
  #u (<& GD OBJ.DIR>,
      <& REG OBJ.DIR.VER>:::'+'
    /(legal.obj(<*>),
      <& GD OBJ.DIR>:::<*>,
      advance
    // pva.free,
      legal.obj(<& REG PVA>),
      assign.pva.to(<& GD OBJ.DIR>));)
```

⁷ Observera här att det bara är nominalfraser vars huvudord är ett pronomen som kan ha nominativt kasus. För en nominalfras vars huvudord är ett substantiv är kasus antingen genitiv eller odefinierat. *Hunden* accepteras således av `legal.obj`.

Regelkroppen består av den dependent disjunktion. Den första disjunkten svarar mot fallet att ett direkt objekt redan finns och detta kan bli aktuellt endast vid analysen av infinitivfraser. Närmre detaljer om detta ges därför senare (s. 43 f.). Den andra disjunkten motsvarar i princip regelkroppen hos `obj.indir`, och utgörs sålunda av en independent disjunktion med två disjunkter. Den första tar hand om fall där det direkta objektet står i sin normalposition, exempelvis *Han ger en bok åt mannen..* De fall där detta topikaliserats, som i *En bok ger han åt mannen.* hanteras av den andra disjunkten. Regeln `obj.prep` känner igen prepositionsobjekt:

```
(define sve.gram-entry obj.prep
  #u (<* PHR.CAT>='PP,
      <* PREP LEX>=?1,
      <& GD OBJ.PREP>:=:<*>,
      advance
  // pva.free,
      <& REG PVA PHR.CAT>='PP,
      <& REG PVA PREP LEX>=?1,
      assign.pva.to(<& GD OBJ.PREP>)
  // pva.free,
      <* PHR.CAT>='PP,
      <* PREP LEX>=?1,
      not <* RECT>,
      <& REG PVA PHR.CAT>='np,
      <& GD OBJ.PREP>:=:<*>,
      assign.pva.to(<& GD OBJ.PREP RECT>),
      advance);)
```

Ett prepositionsobjekt utgörs alltid av en prepositionsfras, där valet av preposition inom snäva gränser är styrt av verbet. Den här använda prepositionsobjektsregeln tillåter bara den preposition som ges som argument till regeln. (Motsvarande formella parameter är ?1.) Regelkroppen består av en independent disjunktion med tre disjunkter. Den första motsvarar fallet att prepositionsobjektet står i sin normalposition. Ett exempel på detta är *Hunden dog i rabies..* Den andra disjunkten tar hand om meningar där prepositionsobjektet är topikaliserat, som i *I rabies dog hunden.* De två första disjunkterna motsvarar alltså de båda disjunkterna i subjeks- och objektsreglerna. Den tredje disjunkten i prepositionsobjektsregeln hanterar den avvikande typ av topikalisering som innebär att prepositionsobjektet splittras genom att dess nominala led (rektion) placeras i fundamentalsställning, medan dess preposition står kvar i normalpositionen. En mening som *Rabies dog hunden i.* exemplifierar detta. I detta fall krävs det av den konstituent som står i prepositionsobjektets normalposition att den är en prepositionsfras med "rätt" preposition och att den saknar rektion samt att fundamentet utgörs av en nominalfras. Om dessa villkor är uppfyllda ansätts den rektionslösa prepositionsfrasen som prepositionsobjekt, varefter fundamentet ansätts som dess rektion. Lokationsobjektsregeln uppvisar stora likheter med prepositionsobjektsregeln:

```
(define sve.gram-entry obj.loc
```

```

#u (loc.expr(<*>),
    <& GD OBJ.LOC>:=:<*>,
    advance
// pva.free,
    loc.expr(<& REG PVA>),
    assign.pva.to(<& GD OBJ.LOC>)
// pva.free,
    <*> PHR.CAT>='PP,
    not <*> RECT>,
    <& REG PVA PHR.CAT>='NP,
    <& REG PVA HEAD LEX :LOC>='+,
    <& GD OBJ.LOC>:=:<*>,
    advance,
    assign.pva.to(<& GD OBJ.LOC RECT>));)

```

Predikatet `loc.expr` definierar vilka fraser som kan fungera som lokationsobjekt, vilka är desamma som utgör möjliga rumsadverbial. Det beskrivs närmare i samband med diskussionen av dessa. De två första disjunkterna i `obj.locs` regelkropp tar hand om fallen att lokationsobjektet står i normalposition respektive fundamentställning. Som exempel kan vi ge *Mannen bor här.* för det första fallet, och *Här bor mannen.* för det andra. Den tredje disjunkten tar hand om lokationsobjekt som utgörs av splittrade prepositionsfraser, som i exempelvis *Finland bor mannen i.* Dessa analyseras i princip på samma sätt som splittrade prepositionsobjekt. Här ställs dock inga krav på prepositionen, däremot på dess rektion: Huvudordet i denna måste i lexembasen ha särdraget LOC, vilket anger att ett substantiv kan denotera en "plats". Temporalobjektsregeln är till sin utformning helt parallell med lokationsobjektsregeln:

```

(define sve.gram-entry obj.temp
  #u (temp.expr(<*>),
      <& GD OBJ.TEMP>:=:<*>,
      advance
// pva.free,
      temp.expr(<& REG PVA>),
      assign.pva.to(<& GD OBJ.TEMP>)
// pva.free,
      <*> PHR.CAT>='pp,
      not <*> RECT>,
      <& REG PVA PHR.CAT>='np,
      <& REG PVA HEAD LEX :TEMP>='+,
      <& GD OBJ.TEMP>:=:<*>,
      advance,
      assign.pva.to(<& GD OBJ.TEMP RECT>));)

```

Klassen av möjliga temporalobjekt, som är identisk med klassen av möjliga tidsadverbial, definieras av predikatet `temp.expr`. Detta beskrivs närmare vid redogörelsen av tidsadverbialen. Första disjunkten i regelkroppen tar hand om temporalobjekt i normalposition som exempelvis i *Det börjar nu.* Satser där detta är

topikaliserat, som i *Nu börjar det.* analyseras av den andra disjunkten. Den tredje tar hand om temporalobjekt som utgörs av splittrade prepositionsfraser, som i *Elvatiden börjar det vid..* Dessa hanteras som i lokationsobjektsregeln, med den skillnaden att rektionens huvudord här måste ha särdraget TEMP, som markerar att ett substantiv kan stå för en tidpunkt eller ett tidsintervall.

Expplikationsobjekt kallar jag sådana verbargument som består av explikativa bisatser. Dessa förekommer främst tillsammans med anföringsverb samt verb som betecknar vissa typer av mentala aktiviteter och uttrycker innehållet i en utsaga respektive i en "tanke". I åtminstone det första fallet brukar man tala om "indirekt anföring". Expplikationsobjektregelns formulering är ganska enkel. Regeln har samma logiska uppbyggnad som subjektsreglerna:

```
(define sve.gram-entry obj.expl
  #u (<* PHR.CAT>='CL,
      <* TYPE>='SUBORD,
      <* SUBTYPE>='EXPL,
      <* INTROD LEX>=?1,
      <& GD OBJ.EXPL>::<*>,
      advance
  // pva.free,
      <& REG PVA PHR.CAT>='CL,
      <& REG PVA TYPE>='SUBORD,
      <& REG PVA SUBTYPE>='EXPL,
      <& REG PVA INTROD LEX>=?1,
      assign.pva.to(<& GD OBJ.EXPL>));)
```

Den första disjunkten i regelkroppen tar hand om explikationsobjekt i normalposition och den andra sådana som är topikaliserade. Valet av bisatsinledare INTROD är styrt av verbet och anges som argument till regeln. (Motsvarande formella parameter är ?1). Om denna är *att* kan vi som exempel på de båda fallen ge *Han antyder att hon har tagit pengarna.* och *Att hon har tagit pengarna antyder han..*

Verbargument som utgörs av infinitivfraser, som i *Han börjar sjunga.*, kallar jag infinitivobjekt. Vid analysen av dessa måste man ta hänsyn till att ett led i den överordnade huvudsatsen också, i kraft av grammatiska regler, fungerar som ett argument till verbet i infinitivfrasen. Vi säger att det propageras till infinitivfrasen. Det vanligaste fallet är att subjektet i den överordnade satsen också fungerar som subjekt i infinitivfrasen, som i *Han börjar sjunga.*, men det kan också propageras som direkt objekt till infinitivet, som i *Boken finns att läsa i biblioteket..* Även ett direkt objekt i den överordnade satsen kan spela en roll i infinitivfrasen, som i *Hon tvingar honom att sjunga.*, där det propageras som subjekt till infinitivet. Vilket led som skall propageras och vilken funktion det får i den underordnade infinitivfrasen bestäms av det styrande verbet. Regeln som analyserar infinitivfraser ansätts därför prediktivt, med rätt led i huvudsatsen propagerat som rätt led i infinitivfrasen, efter det att predikatet känts igen. Med denna teknik kan inte meningar med topikaliserade infinitivobjekt analyseras. Även om dessa inte kan utdömas som helt ogrammatiska, torde de inte förekomma i någon större utsträckning i normalt språkbruk, varför denna svaghet kanske kan accepteras. Några exempel på sådana

meningar, *Sjunga börjar han., Att sjunga tvingar hon honom.* etc., visar sådana konstruktioners otymplighet. Infinitivobjektsregeln får följande formulering:

```
(define sve.gram-entry obj.inf
  #u <PROP GD ?2>:=:<& GD ?1>,
  process(c1.inf,<PROP>),
  <* PHR.CAT>='CL,
  <* TYPE>='INF,
  <* ?2>==<& GD ?1>,
  <& GD OBJ.INF>:=:<*>,
  advance;)
```

Två argument ges till regeln; motsvarande formella parametrar är ?1 och ?2. Det första anger vilket led i huvudsatsen som skall propageras till infinitivfrasen och det andra argumentet anger vilken funktion det skall ha där. Denna propagering ombesörjes av de två första konjunkterna i regelkroppen. Det första som görs är att <PROP GD ?2> tilldelas värdet av <& GD ?1>. Därefter ansätts regeln `c1.inf`, som analyserar infinitivfraser, med <PROP> som andra argument till `process`. I den nyansatta bågen kommer därmed <&> att få samma värde som <PROP> har i moderbågen, och alltså, vilket är det väsentliga, kommer <& GD ?2> i den aktiva båge som försöker bygga infinitivfrasen att ha samma värde (beskrivningen av en fras) som <& GD ?1> har i huvudsatsbågen. Resten av regelkroppen står för själva igenkännandet av infinitivobjektet. Där kontrolleras att den i-tur-stående frasen är av rätt typ och att leden propagerats på rätt sätt. Om så är fallet ansätts den som infinitivobjekt i satsen, varefter ett avancemang sker. `obj.inf` har alltså två funktioner, dels att ansätta infinitivregeln och dels att känna igen ett infinitivobjekt. Men dessa funktioner utförs inte samtidigt. Först kommer `obj.inf` att evalueras i ett bearbetningssteg där den inaktiva bågen utgörs av första ordet (ett *att* eller ett infinitiv) i infinitivfrasen. Därvid kommer infinitivregeln att ansättas med rätt led propagerat. Därefter kommer evalueringen av `obj.inf` att avbrytas, eftersom den inaktiva konstituenten inte är en sats. Om den ansatta regeln leder till att en infinitivfras känns igen, så kommer ett nytt bearbetningssteg att bildas där den inaktiva konstituenten utgörs av denna infinitivfras. Vid exekveringen av detta kommer `process`-satsen i `obj.inf` inte att få någon effekt, eftersom infinitivregeln redan är ansatt, och två identiska bågar inte får förekomma i charten. Däremot kommer den inaktiva frasen att passera de kontroller som görs i infinitivregeln och följaktligen att infogas i huvudsatsens beskrivning som infinitivobjekt.

Ovanstående regel, `obj.inf` säger inget om närvaron av ett infinitivmärke, d.v.s. *att*, i infinitivfrasen och godkänner därför infinitivobjekt både med och utan ett *att*. Denna används för verb där *att*et är fakultativt, som exempelvis *börja*. Man kan säga *Han börjar sjunga.* lika väl som *Han börjar att sjunga.* I de fall då inget fritt val föreligger används istället någon av reglerna `obj.inf.mark` och `obj.inf.no.mark`. Den första kräver av infinitivobjektet att ett infinitivmärke skall ingå i det och och sådana infinitivobjekt förekommer till exempelvis verbet *hata*, i meningar som *Han hatar att sjunga.*, där *att*et inte kan strykas utan att meningen blir ogrammatisk. Den andra regeln, `obj.inf.no.mark`, används för igenkänning av infinitivobjekt i vilka ett *att* inte får finnas, som t.ex. infinitivobjektet till *kunna* i en mening som

Han kan läsa.. Han kan att läsa. är därför en fullständigt ogrammatisk sats. Dessa två regler är, bortsett från att ett test på när- respektive frånvaron av infinitivmärke tillagts, identiska med `obj.inf`:

```
(define sve.gram-entry obj.inf.mark
  #u <PROP GD ?2>:=:<& GD ?1>,
    process(cl.inf,<PROP>),
    <* PHR.CAT>='CL,
    <* TYPE>='INF,
    <* INF.MARK>,
    <* ?2>==<& GD ?1>,
    <& GD OBJ.INF>:=:<*>,
    advance;
)
```

```
(define sve.gram-entry obj.inf.no.mark
  #u <PROP GD ?2>:=:<& GD ?1>,
    process(cl.inf,<PROP>),
    <* PHR.CAT>='CL,
    <* TYPE>='INF,
    not <* INF.MARK>,
    <* ?1>==<& GD ?2>,
    <& GD OBJ.INF>:=:<*>,
    advance;
)
```

Slutligen finns en regel som känner igen agenter. Den är sålunda aktuell endast vid analysen av passiva satser. Eftersom agenten i många avseenden uppför sig som ett prepositionsobjekt, kommer agentregeln att få nästan samma formulering som prepositionsobjektsregeln:

```
(define sve.gram-entry agent
  #u (<* PHR.CAT>='PP,
    <* PREP LEX>='AV.PP.1,
    <& GD AGENT>:=:<*>,
    advance
  // pva.free,
    <& REG PVA PHR.CAT>='PP,
    <& REG PVA PREP LEX>='AV.PP.1,
    assign.pva.to(<& GD AGENT>)
  // pva.free,
    <* PHR.CAT>='PP,
    <* PREP LEX>='AV.PP.1,
    not <* RECT>,
    <& REG PVA PHR.CAT>='NP,
    <& GD AGENT>:=:<*>,
    assign.pva.to(<& GD AGENT RECT>),
    advance);)
```

Förutom ifråga om namngivningen av det eftersökta ledet, är skillnaden att endast en preposition godkänns i agentregeln, medan vilken preposition som helst kan ges som argument till prepositionsobjektsregeln. Den preposition som agentregeln kräver är givetvis *av*. Den tillåter agenter i normalposition, som i *Hunden ägs av honom*. och topikaliserade agenter, som i *Av honom ägs hunden*.. Dessutom accepteras splittrade agenter, som i *Honom ägs hunden av*.. Sådana konstruktioners grammatikalitet är dock inte helt självklar, förmodligen därför att de bara bildar meningslöst tillkrånglade varianter av motsvarande aktiva sats och därför är sällsynta. *Honom ägs hunden av*. är ju i syntaktiskt (vad gäller de betydelsebärande enheternas ordning) och semantiskt hänseende så gott som likvärdig med den enklare *Han äger hunden*..

När den behövliga repertoaren av argumentidentifieringsregler finns tillgänglig kan verbaktionsregler relativt lätt formuleras. Dessa har, som tidigare sagts, till uppgift att känna igen de argument utöver subjekt eller formellt subjekt som predikatet konstrueras med. Eftersom olika verblexem konstrueras med olika argumentuppsättningar, är en verbaktionsregel lexikaliskt bunden (via lexembasen) till varje verblexem. Förutom på vilket detta är, beror argumentuppsättningen på vilken diates verbet har. Om både aktiv och passiv diates är tillåtna för ett visst verblexem, uttrycks det explicit i dess verbaktionsregel med vilka argument det konstrueras i de båda fallen. I samband med detta används de båda hjälpreglerna *act* och *pass*. Den förstnämnda testar om satsen är i aktiv diates, den andra om den är i passiv. De har följande definitioner:

```
(define sve.gram-entry act
  #u not <& GD PASS>);
```

```
(define sve.gram-entry pass
  #u <& GD PASS>);
```

I grammatiken har ett antal verbaktionsregler definierats. Dessa exemplifierar några olika typer av argumentuppsättningar, men ger på inget sätt en uttömmande beskrivning av vilka argumentuppsättningar som är möjliga. De skall här beskrivas i stigande komplexitetsordning. Först den som tillåter maximalt ett argument till verbet, sedan de som tillåter maximalt två argument o.s.v. Den enklaste verbaktionsregeln är den som tar hand om intransitiva verb. När den anropas har, om satsen är grammatisk, ett subjekt återfunnits. Därefter återstår inga ytterligare argument att finna. Det enda som måste göras är att testa att satsen är i aktiv diates, eftersom de intransitiva verben inte kan förekomma i passivkonstruktioner. Regeln får sålunda följande utseende:

```
(define sve.gram-entry va.glittra
  #u act;)
```

En kommentar om namngivningen av verbaktionsreglerna kan vara på sin plats. Deras namn är av formen *va.X*, där *X* är grundformen av ett verb som är ett typiskt exempel på ett verb som kan konstrueras enligt den aktuella regeln. I ovanstående regel har exempelvis *glittra* valts ut som ett typiskt exempel på ett intransitivt verb.

Två olika regler finns för verb som vid aktiv diates konstrueras med ett subjekt och ett direkt objekt som enda argument. Den första heter `va.tillhöra`, och knyts till verb av detta slag som inte kan konstrueras i passivum. Den andra heter `va.plundra` och tillåter passivering. `va.tillhöra` har detta utseende:

```
(define sve.gram-entry va.tillhöra
  #u act, obj.dir;)
```

Regelkroppen består av två konjunkter: Den första ser till att satsen är i aktiv diates. Den andra är ett anrop till regeln som känner igen det direkta objektet. `va.plundra` måste även kunna ta hand om fallet med passiv diates och får sålunda följande utseende:

```
(define sve.gram-entry va.plundra
  #u (act, obj.dir /
      pass, (agent // continue));)
```

Som synes har vi här att göra med en dependent disjunktion med två disjunkter, och detta är den generella formen för regelkroppen hos de verbaktionsregler som tillåter båda typerna av diates. Den första tar hand om det fall att satsen är i aktiv diates och inleds med ett test på detta. Den andra disjunkten inleds med ett test på motsatsen, och tar följaktligen hand om passiva satser. I regeln `va.plundra` krävs i första disjunkten (den som tar hand om aktiva satser) att ett direkt objekt skall finnas. Om satsen är passiv är agenten fakultativ. Detta uttrycks i den andra disjunkten med hjälp av en independent disjunktion med två disjunkter. Den första svarar mot fallet att en agent finns, och består av ett anrop till agentregeln. Den andra består bara av ett `continue` och tar hand om fallet att agenten saknas, helt enkelt genom att inte göra någonting. Detta är det generella sättet att hantera fakultativa argument.

Många verb konstrueras med ett prepositionsobjekt som enda argument utöver subjektet. Sådana konstruktioner kan inte passiviseras. Ofta är prepositionen i prepositionsobjektet entydigt bestämd av verbet. En grupp verbaktionsregler krävs för att ta hand om dessa fall — en för varje preposition —. I den föreliggande grammatiken finns tre exempel på sådana regler:

```
(define sve.gram-entry va.ansvara
  #u act, obj.prep('FÖR.PP.1);)
```

```
(define sve.gram-entry va.polemiserar
  #u act, obj.prep('MOT.PP.1);)
```

```
(define sve.gram-entry va.prenumererar
  #u act, obj.prep('PÅ.PP.1);)
```

Olika prepositioner kan också alternera i ett prepositionsobjekt, exempelvis i det argument till verbet *dö* som anger dödsorsak. Det kan heta *Hunden dog i rabies*. lika väl som *Hunden dog av sorg*. Prepositionen kan sålunda utgöras av både *i* och *av*. Valet är dock inte fritt, utan styrs av prepositionens rektion. Om denna

denoterar en sjukdom föredras *i*, i annat fall *av*. Den föreliggande grammatiken tar dock inte hänsyn till detta, utan accepterar även *Hunden dog av rabies.* och *Hunden dog i sorg.* Till skillnad från prepositionsobjekten i ovanstående tre regler är prepositionsobjektsargumentet till *dö* fakultativt, och motsvarande verbaktionsregel får följande formulering:

```
(define sve.gram-entry va.dö
  #u act, (obj.prep('I.PP.1) // OBJ.PREP('AV.PP.1) // continue);)
```

Regelkroppen säger, som synes, att satsen skall vara i aktiv diates och att det utöver subjekt bland predikatets argument finns ett prepositionsobjekt med *i* som preposition, eller ett med *av*, eller inget alls.

Verb som konstrueras med ett infinitivobjekt som enda argument utöver subjekt tas om hand av reglerna *va.börja* och *va.kunna*. Skillnaden mellan dem är att den förstnämnda tillåter infinitivobjekt både med och utan *att*, medan den sistnämnda inte tolererar närvaron av ett infinitivmärke. I båda reglerna propageras subjektet i den överordnade satsen som subjekt i den underordnade infinitivfrasen. Passivering är inte möjlig. Dessa verbaktionsregler formuleras sålunda:

```
(define sve.gram-entry va.börja
  #u act, obj.inf(<SUBJ>,<SUBJ>);)

(define sve.gram-entry va.kunna
  #u act, obj.inf.no.att(<subj>,<subj>);)
```

En grupp verb, för vilken *hata* är en representativt exempel, konstrueras vid aktiv diates med antingen direkt objekt, infinitivobjekt, eller explikativobjekt. I det andra fallet är ett *att* obligatoriskt och den överordnade satsens subjekt propageras som subjekt i infinitivfrasen. Passivering är möjlig. Jag räknar därvid agenten som ett obligatoriskt argument. *va.hata* får följande definition:

```
(define sve.gram-entry va.hata
  #u (act, (obj.dir // obj.inf.att(<subj>,<subj>) // obj.expl) /
    pass, agent);)
```

Verbaktionsregeln *va.starta* tar hand om verb som förutom subjekt tar antingen ett lokationsobjekt eller ett temporalobjekt som argument. Den tillåter inte passivering:

```
(define sve.gram-entry va.starta
  #u act, (obj.loc // obj.temp);)
```

Ett verb som *antyd*a kan ta ett explikativobjekt som argument, som i *Hon antyder att mannen kan sjunga.* En sådan konstruktion passiveras på ett avvikande sätt till *Mannen antyds kunna sjunga.* Sådana konstruktioner hanteras av regeln *va.antyd*a:

```
(define sve.gram-entry va.antyd
  #u (act, obj.expl('ATT.SN.1) /
    pass, obj.inf.no.att(<SUBJ>,<SUBJ>));)
```

Verb som vid aktiv diates obligatoriskt konstrueras med ett subjekt, ett direkt objekt och ett prepositionsobjekt är ganska vanliga. Sådana konstruktioner passiveras på normalt sätt. Agenten är fakultativ i detta sammanhang. En regel som tar hand om denna typ av konstruktion finns i den föreliggande grammatiken. Det är *va.förse*, som kräver att prepositionen i prepositionsobjektet är *med*:

```
(define sve.gram-entry va.förse
  #u (act,  obj.dir,
      obj.prep('MED.PP.1) /
      pass, obj.prep('MED.PP.1),
      (agent // continue));)
```

Verb som vid aktiv diates utöver subjekt konstrueras med ett obligatoriskt direkt objekt och ett fakultativt prepositionsobjekt tas om hand av regler som *va.avsätta*. I just den regeln är kravet på prepositionsobjektets preposition att det utgörs av *från*:

```
(define sve.gram-entry va.avsätta
  #u (act,  obj.dir,
      obj.prep('FRÅN.PP.1) /
      pass, obj.prep('FRÅN.PP.1),
      (agent // continue));)
```

Verbaktionsregeln *va.sända* är av samma typ, skillnaden är endast att prepositionsobjektets preposition i denna är *till*:

```
(define sve.gram-entry va.sända
  #u (act,  obj.dir,
      obj.prep('TILL.PP.1) /
      pass, obj.prep('TILL.PP.1),
      (agent // continue));)
```

Verb som tar ett obligatoriskt lokationsobjekt, kan normalt konstrueras med ett "vanligt" subjekt lika väl som med ett formellt subjekt och ett egentligt. *En man bor i huset.* och *Det bor en man i huset.* är båda exempel på välformade meningar. Passivering är omöjlig. Motsvarande verbaktionsregel får följande formulering:

```
(define sve.gram-entry va.bo
  #u act, (subj.proper // continue),
      obj.loc;)
```

Ett verb som *finnas* kan förekomma i konstruktioner av ovanstående typ. Till detta verb kan fogas ytterligare ett argument, en infinitivfras, som i *Boken finns att läsa i biblioteket.*, i vilken subjektet eller det egentliga subjektet i den överordnade satsen propageras som direkt objekt. Passivering av detta verb är inte tillåten. Dess verbaktionsregel definieras på detta sätt:

```
(define sve.gram-entry va.finns
  #u act, (subj.proper,
          (obj.inf.att(<SUBJ.PROPER>,<OBJ.DIR>) // continue)
          // (obj.inf.att(<SUBJ>,<OBJ.DIR>) // continue)),
      obj.loc;)
```

Verbaktionsregeln *va.förvara* knyts till verb som utöver subjekt tar ett direkt objekt och ett lokationsobjekt som argument och som tillåter passivering:

```
(define sve.gram-entry va.förvara
  #u (act, obj.dir,
      obj.loc /
      pass, obj.loc,
      (agent // continue));)
```

Den verbaktionsregel som knyts till verbet *ge* måste kunna ta hand om s.k. “dative shift”, som består i att ett indirekt objekt placerat före det direkta objektet också kan realiseras som ett prepositionsobjekt placerat efter det direkta objektet. Detta exemplifieras av satsparet *Hon ger honom boken.* och *Hon ger boken till honom.* Regeln *va.ge* formuleras sålunda:

```
(define sve.gram-entry va.ge
  #u (act, (obj.dir,
           obj.prep('TILL.PP.1)
           // obj.indir,
           obj.dir) /
      pass, obj.prep('TILL.PP.1),
      (agent // continue));)
```

Denna regel analyserar, som synes, *till honom* i *Hon ger en bok till honom.* som ett prepositionsobjekt. Många grammatiker, exempelvis Thorell (1973), skulle dock kalla även detta för ett indirekt objekt. Vid passiv diates tillåter regeln endast sådana prepositionsobjekt och skulle underkänna exempelvis *Boken ges honom.* Däremot skulle *Boken ges till honom.* godkännas.

Verbaktionsregeln *va.svara* knyts till verb som utöver subjekt tar ett fakultativt direkt objekt och ett explikativobjekt som argument. Sådana konstruktioner kan inte passiveras:

```
(define sve.gram-entry va.svara
  #u act, (obj.dir // continue),
  obj.expl('ATT.SN.1);)
```

Verb som utöver subjekt tar ett direkt objekt och ett infinitivobjekt i vilket det direkta objektet propagerats som subjekt hanteras av verbaktionsregeln *va.tvinga*. Sådana konstruktioner passiviseras på vanligt sätt:

```
(define sve.gram-entry va.tvinga
  #u (act, obj.dir,
      obj.inf.att(<OBJ.DIR>,<SUBJ>) /
      pass, obj.inf.att(<SUBJ>,<SUBJ>));)
```

För att återgå till regeln *main* så ser vi att när verbaktionsregeln exekverats, och verbets alla argument återfunnits, anropas *adverbials2*. Den kommer, om den analyserade meningen tillåts av grammatiken, att konsumera de eventuella adverbial som återstår. *adverbials2* liksom *adv.in.fund*, som tar hand om topikaliserade adverbial, beskrivs i avsnitt 3.5.7. Efter det sistnämnda regel anropats görs ett test på att fundamentet blivit tilldelat en funktion. Detta görs av *pva.used*:

```
(define sve.gram-entry pva.used
  #u if <& REG PVA> then <& REG PVA.FUNC> else continue;)
```

Villkorligheten i regeln kommer sig av att den används också i bisatsregeln. Värdet av <& REG PVA> är alltid definierat vid analysen av huvudsatsen och regeln säger att då måste också <& REG PVA.FUNC> vara definierad. Denna flagga tilldelas ett värde av `assign.pva.to` samtidigt som fundamentet tilldelas en funktion. Därefter försäkras att anrop av regeln `end.of.sent` att ett skiljetecken avslutar meningen:

```
(define sve.gram-entry end.of.sent
  #u <* SEP.CAT>,
  <& GD SEP>:=:<* SEP.CAT>;
  #! SEP.CAT;)
```

Normalt är det en punkt som avslutar en mening, men även andra kan förekomma, som frågetecken och utropstecken. Det är av relevans för analysen vilket det anträffade skiljetecknet är. Det infogas därför i den grammatiska beskrivningen av meningen, som SEP (separator).

3.5.2 Analysen av imperativa huvudsatser

De tekniker som används vid analysen av av deklarativa huvudsatser är i stor utsträckning också tillämpbara vid analysen av imperativa och interrogativa huvudsatser. Regler för dessa kan således formuleras relativt lätt.

I imperativa huvudsatser, d.v.s. befallningar, är ordningen mellan de ingående leden i princip densamma som i deklarativa huvudsatser. Fundamentet och varje slag av subjekt saknas dock i de imperativa satserna. På verbet ställs kravet att det skall ha imperativ form och det avslutande skiljetecknet skall vara ett utropstecken. Regeln för imperativa huvudsatser `cl.imp`, som ansätts i `start.rule`, får således följande utseende:

```
(define sve.gram-entry cl.imp
  #u <& GD PHR.CAT>:=:'CL,
  <& GD TYPE>:=:'MAIN,
  <& GD MODE>:=:'IMP,

  v.imp,
  advance(adverbials1),
  do(<& REG V1.LEM :VERBCOMP>),
  do(<& GD PRED LEX :VERBACTION>),
  adverbials2,

  mark.of.exclamation,
  store*;

  #! IMP;)
```

Efter att fraskategori, satstyp och modus angivits, sker ett anrop till `v.imp` som känner igen det imperativa verbet. Därefter kommer den första adverbialsgruppen, varefter den till verbet knutna verbkompositionsregeln aktiveras. Den har i det här sammanhanget enbart att ta hand om eventuell verbpartikel, eftersom en imperativ sats är obestämd för tempus och, följaktligen, sammansatta tempus inte är möjliga. (Eller vad skall man säga om befallningen *Ha sjungit!?*) När så satsens predikat fastställts, aktiveras dess verbaktionsregel, som känner igen de argument som predikatet tar, utöver subjekt. (Subjektet förblir odefinierat i en imperativ sats.) Därefter kommer den andra (finala) adverbialsgruppen. Det avslutande utropstecknets närvaro kontrolleras av `mark.of.exclamation`. Hjälpregeln `v.imp` har givits följande definition:

```
(define sve.gram-entry v.imp
  #u <* WORD.CAT>='VERB,
  <* IMP>='+,
  <& REG V1.LEM>:=:<* LEM>;)
```

`v.imp` arbetar på samma sätt som sin parallellregel `v.fin`, men är enklare, enär den inte behöver ta hänsyn varken till tempus eller passiv diates (som är omöjlig i imperativt modus). Utropsteckensregeln `mark.of.exclamation` är synnerligen enkel:

```
(define sve.gram-entry mark.of.exclamation
  #u <* SEP.CAT>='%!;)
```

Eftersom utropstecknet är obligatoriskt anges det inte i beskrivningen av en imperativ huvudsats (till skillnad från det skiljetecken som avslutar en deklarativ huvudsats).

3.5.3 Analysen av ja-nej-frågor

En ja-nej-fråga består, syntaktiskt sett, av en deklarativ huvudsats, där inget led topikaliserats. Regeln som analyserar ja-nej-frågor, `cl.interr`, kommer således att uppvisa stora likheter med huvudsatsregeln `cl.decl`, och ansätts liksom denna i `start.rule`:

```
(define sve.gram-entry cl.interr
  #u <& GD PHR,CAT>:=:'CL,
  <& GD TYPE>:=:'MAIN,
  <& GD MODE>:=:'INTERR,
  <& GD QUESTIONTYPE>:=:'YES-NO,

  v.fin,
  (advance(subj) // advance(subj.form)),
  adverbials1,
  do(<& REG V1.LEM :VERBCOMP>),
  do(<& GD PRED LEX :VERBACTION>),
  adverbials2,
```

```
(if <& GD SUBJ.FORM>
  then <& GD SUBJ.PROPER>
  else continue),
questionmark,
store*;
```

```
#! TENSE;)
```

Förutom fraskategori, satstyp och modus, ansätts initialt i regeln också frågetyp (QUESTIONTYPE), som här anger att vi har att göra med en ja-nej-fråga (YES-NO) (och inte en frågeordsfråga). Regeln kräver att ett frågetecken avslutar frågan. Denna kontroll utförs av regeln `questionmark`:

```
(define sve.gram-entry questionmark
  #u <* SEP.CAT>='%?;)
```

Den har som synes stora likheter med `mark.of.exclamation`.

3.5.4 Analysen av relativa bisatser

Medan huvudsatserna bildar en relativt homogen kategori, finns det ett flertal typer av bisatser, som till form och funktion väsentligen skiljer sig från varandra. Den föreliggande grammatiken kan hantera två typer: relativsatser och explikativa bisatser.

Relativsatser förekommer som bestämningar i nominalfraser och är knutna till ett korrelat. Normalt är detta identiskt med nominalfrasens huvudord. Som bisatsinledare förekommer normalt ett relativpronomen. Enligt den traditionella synen bland grammatiker är detta koreferent med korrelatet och har, förutom att vara bisatsinledare, ytterligare en funktion i bisatsen. I exempelvis nominalfrasen *mannen som sjunger*, anses *som* vara subjekt i relativsatsen. Här kommer dock denna typ av konstruktioner att analyseras på ett annat sätt: Vi kommer att anse att det är korrelatet som utgör ett satsled i relativsatsen, medan relativpronomenet aldrig fungerar som mer än bisatsinledare. I analysen av *mannen som sjunger* kommer sålunda *mannen* att anges som subjekt i relativsatsen. Två argument kan anföras till försvar för detta val. För det första får relativpronomenet under vissa förutsättningar utelämnas. (Se nedan!) Detta leder dock inte till att något satsled blir odefinierat i relativsatsen. Den analys vi valt får exempelvis inga problem med relativsatsen i nominalfrasen "hunden han äger". *Hunden* är det direkta objektet i denna. För det andra är den analys vi valt mer informationsrik. Angivelsen att ett *som* eller *vilken* har en viss funktion är mindre informativ än en specifikation där det anges vilken nominalfras som innehar den aktuella funktionen. Denna ytterligare information som vår analys ger är t.ex. användbar om man skulle vilja utnyttja selektionsrestriktioner i grammatiken.

Vi kan uppfatta relativsatsen som en bisats där ett led flyttats bort från sin normala position och realiserats som dess korrelat, på i princip samma sätt som ett led i huvudsatsen flyttas från sin normalposition till fundamentet. Det visar sig vid närmre eftertanke att ett led som kan uppträda som korrelat till en relativsats också

kan topikaliseras i en motsvarande huvudsats, och vice versa. Följande nominalfras-huvudsatspar exemplifierar detta:

<i>mannen som sjunger</i>	<i>Mannen sjunger.</i>
<i>huset som han bor i</i>	<i>Huset bor han i.</i>
<i>boken som han läser</i>	<i>Boken läser han.</i>

Likheten, i detta avseende, mellan fundamentet i en huvudsats och korrelatet till en bisats förklarar vitsen med att argumentidentifieringsreglerna inte refererar direkt till fundamentet utan till `<& REG PVA>`. (PVA uttyds som det PreVerbala Argumentet.) Samma argumentidentifieringsregler kan därmed användas vid analysen av relativsats. Detta blir möjligt genom att korrelatet görs identiskt med `<& REG PVA>`. Begreppet preverbalt argument inbegriper alltså både korrelat och fundament.

Relativsatsen känns igen av regeln `cl.rel`. Vid ansättningen är korrelatet definierat. Denna måste alltså göras prediktivt. Ansättningen görs i regeln `attr.rel`, som anropas vid analysen av en nominalfras, just när huvudordet känts igen. `attr.rel` har följande formulering:

```
(define sve.gram-entry attr.rel
  #u <PROP GD CORR PHR.CAT>:=:<& GD PHR.CAT>,
  <PROP GD CORR HEAD>:=:<& GD HEAD>,
  nn-compatible(<PROP GD CORR>,<& gd>),
  process(cl.rel,<PROP>),
  <* PHR.CAT>='CL,
  <* SUBTYPE>='REL,
  <* CORR PHR.CAT>==<& GD PHR.CAT>,
  <* CORR HEAD>==<& GD HEAD>,
  nn-compatible(<* CORR>,<& GD>),
  <& GD ATTR.REL>:=:<*>,
  store*;)

```

Korrelatet byggs upp som ett utsnitt av den nominalfras som är under uppbyggnad. Detta utsnitt ärver fraskategori och huvudord från denna och har samma grammatiska drag. Dessa kopieras av proceduren `nn-compatible`. Därefter ansätts relativsatsregeln med `process(cl.rel,<prop>)`. I den bildade aktiva bågen ligger sålunda korrelatet under `<& GD CORR>`. Resten av regeln aktiveras då en färdig-analyserad relativsats finns tillhanda och infogar denna som ett relativattribut `attr.rel` i nominalfrasens beskrivning, som därefter lagras. På följande sätt formuleras Relativsatsregeln `cl.rel`:

```
(define sve.gram-entry cl.rel
  #u <& GD PHR.CAT>:=:'CL,
  <& GD TYPE>:=:'SUBORD,
  <& GD SUBTYPE>:=:'REL,
  <& REG PVA>:=:<& GD CORR>,
  (introd.rel // continue),
  cl.sub;
  #! PHR.CAT or WORD.CAT or SEP.CAT;)

```

Först, som synes, definieras fraskategori, typ, och SUBTYPE, vilket drag anger vilken typ av bisats vi har att göra med. Därefter ansätts korrelatet, av nyssnämnda skäl, som det preverbala argumentet. `introd.rel` känner igen ett relativpronomen och infogar det som bisatsinledare i relativsatsens beskrivning. Relativpronomenet anges i regeln som fakultativt. Restriktionen att om korrelatet är subjekt i bisatsen så är relativpronomenet obligatoriskt, uttrycks i den regel som därefter anropas, `cl.sub.introd.rel` definieras på följande sätt:

```
(define sve.gram-entry introd.rel
  #u <* WORD.CAT>='REL.PRON,
    nue(<& GD CORR SPEC>,<* FORM>),
    nue(<& GD CORR NUMB>,<* NUMB>),
    <& GD INTROD>:=:<* LEM :DLEX>,
    advance;)
```

Regeln kräver att korrelatet och relativpronomenet kongruerar: Korrelatets species och relativpronomenets form får inte vara motstridiga. Inte heller får de ha olika numerus. (Predikatet `nue` (Not Unequal) är sant om de båda argumenten inte är icke-ekvivalenta, d.v.s. om de är ekvivalenta eller om något av dem är odefinierat.) Regeln `cl.sub` tar hand om själva bisatskroppen hos båda de typer av bisatser som grammatiken förmår hantera. Den har givits följande definition:

```
(define sve.gram-entry cl.sub
  #u (subj // subj.form),
    (if <& GD SUBJ>==<& REG PVA>
      then <& GD INTROD>
      else continue),
    adverbials1,
    v.fin,
    advance,
    do(<& REG V1.LEM :VERBCOMP>),
    do(<& GD PRED LEX :VERBACTION>),
    adverbials2,

    (if <& GD SUBJ.FORM>
      then <& GD SUBJ.PROPER>
      else continue),
    adv.in.pva,
    pva.used,
    minorstore*;
  #! PHR.CAT or WORD.CAT or SEP.CAT;)
```

Likheterna mellan denna regel och huvudsatsregeln `cl.decl` (s. 19) är uppenbara. Skillnaden ligger främst däri att i bisatsen kommer subjektet eller det formella subjektet och första adverbialsgruppen före det finita verbet, medan ordningen i huvudsatsen är att subjektets eller det formella subjektets normalposition är efter det finita verbet och den första adverbialsgruppens placering därefter. Efter de parallella

anropen till subjeksreglerna (`subj` och `subj.form`), uttrycks det nyssnämnda villkoret att en bisatsinledare måste finnas i de fall det preverbala argumentet fungerar som subjekt. I övrigt, förutom att regeln inte räknar med att ett skiljetecken avslutar bisatsen, fortskrider analysen av en bisats på samma sätt som analysen av en huvudsats. De flesta av huvudsatsregelns hjälpprocedurer kan sålunda återanvändas i bisatsregeln.

3.5.5 Analysen av explikativa bisatser

Explikativa bisatser, den andra kategorin av bisatser som kan hanteras av den föreliggande grammatiken, förekommer vid s.k. indirekt anföring och deras semantiska funktion är att uttrycka innehållet i ett yttrande (påstående eller fråga) eller ett faktum. De inleds vanligen med ett *att* eller ett *om* (Dessa är i lexembasen markerade med `EXPL=+`. Vi kan kalla dem för explikativa subjunktioner.) Som exempel på en explikativ bisats kan *att hon sjunger* i *Han antyder att hon sjunger.* nämnas. Bisatsinledaren i en explikativ bisats har ingen grammatisk funktion i denna utöver att vara bisatsinledare. Regeln som analyserar explikativa bisatser, `cl.expl` får sålunda följande utseende:

```
(define sve.gram-entry cl.expl
  #u <& GD PHR.CAT> ::= 'CL,
      <& GD TYPE> ::= 'SUBORD,
      <& GD SUBTYPE> ::= 'EXPL,
      <* WORD.CAT> = 'SUBJU,
      <& GD INTROD LEX> ::= <* LEM :LEX>,
      <& GD INTROD LEX :EXPL> = '+,
      advance(cl.sub);
  #! WORD.CAT;)
```

3.5.6 Analysen av infinitivfraser

Infinitivfrasen räknas här, jämte huvud- och bisatsen, som en av satsens tre huvudtyper. Liksom huvudsatserna bildar infinitivfraserna en i formellt avseende relativt homogen kategori. Grammatiken kan i sin föreliggande form endast ta hand om infinitivfraser som fungerar som argument till verb, alltså som infinitivobjekt. (Se beskrivning av reglerna `obj.inf`, `obj.inf.mark` och `obj.inf.no.mark`!) De har dock i verkligheten fler uppgifter än så att fylla i språket. Normalt (när de fungerar som infinitivobjekt) propageras ett led i den överordnade satsen till den underordnade infinitivfrasen, i vilken detta led eventuellt får en annan funktion. Det är antingen ett subjekt eller ett direkt objekt i infinitivfrasen som på detta sätt kommer utifrån. Infinitivfraser analyseras av regeln `cl.inf`:

```
(define sve.gram-entry cl.inf
  #u <& GD PHR.CAT> ::= 'CL,
      <& GD TYPE> ::= 'INF,
      (if <& GD OBJ.DIR>
        then <& REG OBJ.DIR.PROP> ::= '+)
```

```

else continue),

(inf.mark,
 adverbials1
//continue),
v.inf,
do(<& REG V1.LEM :VERBCOMP>),
do(<& GD PRED LEX :VERBACTION>),
(if <& REG OBJ.DIR.PROP>
 then <& REG OBJ.DIR.VER>
 else continue),
adverbials2,
minorstore*;
#! WORD.CAT or PHR.CAT or SEP.CAT;)

```

Vid regelns ansättning är ett subjekt eller direkt objekt redan definierat. I det senare fallet ges en flagga <& REG OBJ.DIR.PROP> värdet +. Detta görs för att det senare skall vara möjligt att kontrollera att en verbaktionsregel har verifierat närvaron av detta direkta objekt. (Vi skall strax se hur.) I det föregående fallet, när ett subjekt propagerats, behöver inga speciella tester göras, eftersom varje verb kan ta ett subjekt och eftersom ett subjekt aldrig utsätts explicit i en infinitivfras. Infinitivfraser inleds av ett fakultativt infinitivmärke, alltid *att*. (Från- eller närvaron av detta styrs, eller lämnas oavgjord, av de olika infinitivobjektsreglerna.) Därefter kommer den första adverbialsgruppen. Denna får dock förekomma endast under förutsättning att infinitivmärket är utsatt (vilket får som konsekvens att infinitivmärkeslösa infinitivfraser inte kan negeras⁸). Därefter (eller först) i infinitivfrasen kommer verbet i infinitivform, som känns igen av regeln *v.inf*. Verbkompositions- och verbaktionsreglerna fungerar som vid analysen av huvud- och bisatser. Om infinitiv verbet utgörs av *ha* tilldelar regeln *vc.aux.ha* (s. 24) infinitivfrasens tempus värdet *PERF.INF*, som står för perfekt infinitiv. I alla andra fall räknas infinitivfrasens tempus som odefinierat. Ovannämnda verifikation av närvaron av ett direkt objekt görs av argumentidentifieringsregeln *obj.dir*. Som vi ser i definitionen av denna regel (s. 25 f) kommer flaggan <& OBJ.DIR.VER> att tilldelas värdet + om ett direkt objekt redan finns definierat när anropet till regeln sker. Villkoret att om ett direkt objekt propagerats så skall dess närvaro också på detta sätt ha verifierats, uttrycks i regelkroppen till *cl.inf* alldeles efter anropet till verbaktionsregeln. Sist i infinitivfrasen får, liksom i huvud- och bisatsen, en andra adverbialsgrupp förekomma. Den hjälpregel till *cl.inf* som känner igen ett infinitivmärke, *inf.mark*, har följande utseende:

```

(define sve.gram-entry inf.mark
  #u <* LEM>='ATT.SN,
    <& GD INF.MARK>:=:<* LEM :LEX>,
  advance;)

```

⁸ Alla håller inte med om detta.

Regeln `inf.mark` torde kunna förstås utan några ytterligare kommentarer. Regeln `v.inf` känner igen infinitiv verbet vid analysen av infinitivfraser:

```
(define sve.gram-entry v.inf
  #u <* WORD.CAT>='VERB,
  <* INF>='+,
  <& REG V1.LEM>:=:<* LEM>,
  <& REG V1.INF>:=:'+,
  unif-atom(<& REG V1.PASS>,<* PASS>),
  advance;)
```

Verknings sättet hos `v.inf` är helt parallellt med det hos regeln `v.fin`, som känner igen det finita verbet i huvud- och bisatser. Kommentarer till denna regel (s. 20 f) förklarar också hur `v.inf` arbetar.

3.5.7 Analysen av adverbial i de olika satstyperna

I huvudsatsen kan, förutom i fundamentet, ett adverbial placeras på två ställen, dels i positionen närmast efter subjektets normalposition, dels sist i satsen. (Detta syns i definitionen av huvudsatsregeln.) Antalet adverbial i dessa två positioner är i princip inte begränsat. Vi kan alltså tala om två möjliga adverbialsgrupper. Till sin sammansättning är de delvis olika: Vissa slag av adverbial placeras företrädesvis i den första gruppen, medan en placering i den andra gruppen föredras för andra typer av adverbial. I bisatsen och infinitivfrasen finns likaledes två möjliga adverbialsgrupper. I dessa båda satstyper placeras den första direkt före det finita verbet och den andra sist i satsen. Med avseende på vilka slag av adverbial som normalt kan ingå är de första adverbialsgrupperna i huvudsats, bisats och infinitivfras likadana. Detsamma gäller de finala adverbialsgrupperna. De båda regler som känner igen den första och den andra adverbialsgruppen i huvudsatsen kan således användas också i bisats- och infinitivfrasregeln. De båda reglerna heter `adverbials1` och `adverbials2`. Formuleringen av dem förutsätter att ett antal adverbialstyper finns definierade. Det är alltså lämpligt att först gå igenom dessas definitioner.

Grammatiken känner igen ett begränsat antal adverbialstyper — den har i detta avseende stora lakuner —, inalles sex stycken: sanningsvärdes- och talarattitydsadverbial, konjunktionella adverbial, omständighets-, tids- och rumsadverbial. Var och en av dessa typer karakteriseras av ett predikat som definierar vilka fraser som kan utgöra ett adverbial av den aktuella typen.

Sanningsvärdesadverbial är, semantiskt sett, sådana adverbial som anger (talarens bedömning av) satsens (verbhandlingens) sanningshalt. Hit räknas negationer (som *inte*, *knappast* etc.) och adverbial som *kanske*, *troligen* och *säkert*. Klassen definieras av predikatet `legal.adv.truth`:

```
(define sve.gram-entry legal.adv.truth
  #u <?1 PHR.CAT>='ADVP,
  <?1 HEAD LEX :TRUTH>='+;)
```

Sanningsvärdesadverbial utgörs alltså av adverbfraser vars huvudord i lexembasen markerats som ett sanningsvärdesadverb, d.v.s. har särdraget `TRUTH`. Ett talar-

attitydsadverbial anger talarens inställning till det faktum som uttrycks av satsen. Typiska exempel är *tyvärr* och *lyckligtvis*. Predikatet `legal.adv.attitude` definierar vilka de kan vara:

```
(define sve.gram-entry legal.adv.attitude
  #u  <?1 PHR.CAT>='ADVP,
      <?1 HEAD LEX :ATTITUDE>='+;)
```

Adverbfraser vars huvudord är av talarattitydstyp, d.v.s. i lexembasen är markerade med särdraget `ATTITUDE` kan således fungera som talarattitydsadverbial. Konjunktionella adverbial uttrycker en sats (semantiska) förhållande till förkontexten. Det är sålunda fråga om adverbial som *sålunda*, *nämligen*, och *också*. Dessa adverbial definieras av predikatet `legal.adv.conj`:

```
(define sve.gram-entry legal.adv.conj
  #u  <?1 PHR.CAT>='ADVP,
      <?1 HEAD LEX :CONJ>='+;)
```

Konjunktionella adverbial utgörs alltså av adverbfraser vars huvudord i lexembasen markerats som ett konjunktionellt adverb, d.v.s. har särdraget `CONJ`. Omständighetsadverbial benämnes här sådana adverbial som består av en prepositionsfras vars rektion utgörs av en explikativ bisats med *att* som bisatsinledare. Endast ett fåtal prepositioner är tillåtna i detta sammanhang, exempelvis *utan*, *genom*, och *för*. Dessa är i lexembasen markerade med särdraget `CIRC`. Predikatet som definierar klassen av omständighetsadverbial får sålunda följande utseende:

```
(define sve.gram-entry legal.adv.circ
  #u  <?1 PHR.CAT>='PP,
      <?1 RECT PHR.CAT>='CL,
      <?1 RECT INTROD LEX>='ATT.SN.1,
      <?1 PREP LEX :CIRC>='+;)
```

Tidsadverbialen, vars funktion är att specificera tidpunkten för det som uttrycks i en sats, är i formellt avseende en mer varierad klass. Den föreliggande grammatiken godkänner två frastyper som tidsadverbial: adverbfraser och prepositionsfraser. På en adverbfras ställs kravet att dess huvudord skall vara markerat som ett temporalt adverb (`TEMP=+` i lexembasen). *Nu*, *ofta* och *ibland* är exempel på sådana adverbial. En prepositionsfras tillåts som tidsadverbial endast om dess rektion har en temporal referens. Detta krav är uppfyllt om huvudordet i lexembasen givits särdraget `TEMP` (som innebär att det kan referera till en tidpunkt), vilket exempelvis substantivet *elvatid* har. Predikatet `temp.expr`, som definierar klassen av möjliga tidsadverbial, får således följande definition:

```
(define sve.gram-entry temp.expr
  #u  (<?1 PHR.CAT>='ADVP,
      <?1 HEAD LEX :TEMP>='+
      /<?1 PHR.CAT>='PP,
      <?1 RECT>,
      <?1 RECT PHR.CAT>='NP,
      <?1 RECT HEAD LEX :TEMP>='+);)
```

Detta predikat används även i temporalobjektsregeln: En fras som kan fungera som tidsadverbial kan också fungera som temporalobjekt. Samma förhållande gäller mellan rumsadverbial och lokationsobjekt. Vilka fraser som kan fungera som sådana definieras av predikatet `loc.expr`:

```
(define sve.gram-entry loc.expr
  #u (<?1 PHR.CAT>='ADVP,
      <?1 HEAD LEX :LOC>='+
      /<?1 PHR.CAT>='PP,
      <?1 RECT>,
      <?1 RECT PHR.CAT>='NP,
      <?1 RECT HEAD LEX :LOC>='+);)
```

Grammatiken tillåter, som synes, adverbfraser och prepositionsfraser som rumsadverbial. En adverbfras godkänns endast om dess huvudord är ett lokativt adverb, d.v.s. märkt med särdraget LOC i lexembasen. Exempel på sådana adverb är *här*, *överallt* och *därborta*. En prepositionsfras tillåts som rumsadverbial endast om den som rektion har en nominalfras med lokativ referens. Detta krav anses vara uppfyllt om nominalfrasens huvudord har särdraget LOC. Exempel på substantiv med detta särdrag är *bibliotek*, *sko* och *strand*.

Den normala placeringen av ett icke topikaliserat adverbial är, om detta är av den typ som brukar kallas satsadverbial, till vilken sanningsvärdes- och talarattitydsadverbial samt konjunktionella adverbial räknas, i den första adverbialsgruppen. När det gäller övriga typer av adverbial föredras en placering i den andra (finala) adverbialsgruppen. I en mening som *Männen plundrade tyvärr inte staden igår* är adverbialen placerade enligt detta mönster. Om vi i grammatiken utesluter andra sätt att distribuera adverbialen i en mening, kan den subregel som känner igen den första adverbialsgruppen ges följande formulering:

```
(define sve.gram-entry adverbials1
  #u (legal.adv.truth(<*>),
      (& GD ADV.TRUTH11>,
       assign(& GD ADV.TRUTH12>,<*>)
       /assign(& GD ADV.TRUTH11>,<*>))
      /legal.adv.attitude(<*>),
      (& GD ADV.ATTITUDE11>,
       assign(& GD ADV.ATTITUDE12>,<*>)
       /assign(& GD ADV.ATTITUDE11>,<*>))
      /legal.adv.conj(<*>),
      (& GD ADV.CONJ11>,
       assign(& GD ADV.CONJ12>,<*>)
       /assign(& GD ADV.CONJ11>,<*>)),
      advance,
      adverbials1
      //continue;
  #! PHR.CAT or WORD.CAT or SEP.CAT;)
```

Regeln godkänner att den första adverbialsgruppen är tom eller innehåller ett eller flera adverbial. Namnen på de attribut som håller adverbialen är fyrledade. Första ledet **ADV.** anger att det är fråga om ett adverbial. Andra ledet anger vilket slag av adverbial det är fråga om. **TRUTH** står för sanningsvärdesadverbial, **ATTITUDE** för talarattitydsadverbial, **CONJ** för konjunktionellt adverbial, **CIRC** för omständighetsadverbial, **TEMP** för tidsadverbial, och **LOC** för rumsadverbial. Det tredje ledet utgörs antingen av siffran 1 eller siffran 2 och anger vilken av de båda adverbialsgrupperna adverbialet förekommer i. Det sista ledet i namnet utgörs av ett löpnummer, vars funktion är att ge unika namn åt adverbialen, i de fall två adverbial av samma typ är placerade i samma adverbialsgrupp⁹. Exempelvis får de två adverbialen i *Han har troligen inte dött.* beteckningarna **ADV.TRUTH11** och **ADV.TRUTH12**.

Logiskt sett är regeln, **adverbials1**, av en independent disjunktion. I den första disjunkten testas först om den i-tur-stående konstituenten är ett sanningsvärdesadverbial. Är så fallet tilldelas **ADV.TRUTH11**, eller, om denna redan är definierad, **ADV.TRUTH12**, värdet av den i-tur-stående konstituentens beskrivning.

Ger testet negativt resultat för sanningsvärdesadverbial, upprepas proceduren för talarattitydsadverbial och konjunktionellt adverbial. Misslyckas alla tre testerna avbryts analysen. I det motsatta fallet, då ett adverbial sålunda infogats i beskrivningen av den mening som är under analys, sker ett avancemang och, därefter, ett rekursivt anrop till **adverbials1**. Detta för att eventuella återstående led i den första adverbialsgruppen skall kunna omhändertas. (Varje anrop till **adverbials1** konsumerar maximalt ett led.) Den andra disjunkten i regeln består bara av ett **continue** och tar hand om fallet att den första adverbialsgruppen är tom eller, i de fall **adverbials1** anropats av sig själv, inga adverbial återstår i den första adverbialsgruppen.

Regeln som analyserar den andra (finala) adverbialsgruppen, **adverbials2**, är konstruerad enligt samma mönster som **adverbials1**, men, i enlighet med det som ovan sagts, är det andra typer av adverbial som tillåts i denna:

```
(define sve.gram-entry adverbials2
  #u (legal.adv.circ(<*>),
      (& GD ADV.CIRC21>,
       assign(& GD ADV.CIRC22>,<*>)
       /assign(& GD ADV.CIRC21>,<*>))
    /temp.expr(<*>),
      (& GD ADV.TEMP21>,
       assign(& GD ADV.TEMP22>,<*>)
       /assign(& GD ADV.TEMP21>,<*>))
    /loc.expr(<*>),
      (& GD ADV.LOC21>,
       assign(& GD ADV.LOC22>,<*>)
       /assign(& GD ADV.LOC21>,<*>))),
  advance,
```

⁹ Fler än två adverbial av samma typ i samma adverbialsgrupp klarar grammatiken inte av, men enbart triviala ändringar behövs för att öka detta antal.

```

    adverbials2
  //continue;
  #! PHR.CAT or WORD.CAT or SEP.CAT;)

```

I fundamentalsposition kan nästan varje typ av adverbial stå. Den regel, `adv.in.pva`, som tar hand om topikaliserande adverbial tillåter var och en av de sex typer av adverbial som definierats i grammatiken att stå i fundamentalsposition:

```

(define sve.gram-entry adv.in.pva
  #u (pva.free,
    (legal.adv.truth(<& REG PVA>),
      assign.pva.to(<& GD ADV.TRUTH>,<& REG PVA>)
    /legal.adv.attitude(<& REG PVA>),
      assign.pva.to(<& GD ADV.ATTITUDE>,<& REG PVA>)
    /legal.adv.conj(<& REG PVA>),
      assign.pva.to(<& GD ADV.CONJ>,<& REG PVA>)
    /legal.adv.circ(<& REG PVA>),
      assign.pva.to(<& GD ADV.CIRC>,<& REG PVA>)
    /temp.expr(<& REG PVA>),
      assign.pva.to(<& GD ADV.TEMP>,<& REG PVA>)
    /loc.expr(<& REG PVA>),
      assign.pva.to(<& GD ADV.LOC>,<& REG PVA>))
  //continue);)

```

När ett adverbial är identiskt med fundamentet läggs inga siffror till dess namn, eftersom dess position entydigt bestäms genom angivelsen att det är identiskt med fundamentet.

3.6 Innehållet i LPS-0's lexikala komponent

Den lexikala komponenten i en språkbeskrivning till UCP består av tre delar: ett antal morfleksikon (stam- och affixleksikon), en lemma- och en lexembas. Morfleksikonen innehåller morfer (stammar, böjningsmorfer eller oböjliga ord) till vilka en UCP-sats knutits. När en teckensträng återfunnits i ett morfleksikon evalueras denna sats. Morfleksikonen representerar därmed den aktiva delen av lexikonet. Lemma- och lexembasen utgör den deklarativt formulerade delen av lexikonet. Som ett exempel på de olika lexikonkomponenternas användning kan vi se på den lexikala information som finns knuten till verbet *förklara*. Detta verb har en regelbunden böjning och tillhör den första konjugationen. I LPS-0's språkbeskrivning motsvaras denna av böjningsmönstret `pattern.älska`. Denna information knyts till verbets stam, som i detta fall är *förklara* och läggs i stamlexikonet `sve.dic`:

```

(define sve.dic förklara #u pattern.älska;)

```

Till lemmat `FÖRKLARA.VB` knyts information om vilka lexem som ingår i lemmat. Detta anges sålunda i lemmabasen:

```

(define lemma FÖRKLARA.VB (LEX FÖRKLARA.VB.1) (LEX2 FÖRKLARA.VB.2))

```

Den teknik som utnyttjas för att knyta flera lexem till ett lemma är att det

första lexemet knyts till attributet `LEX`, det andra till `LEX2` o.s.v. För verblexemen anges vilka verbaktionsregler som är förknippade med dem:

```
(define lexeme FÖRKLARA.VB.1 (VERBACTION va.plundra))
(define lexeme FÖRKLARA.VB.1 (VERBACTION va.ntyda))
```

I LPS-0 används lemmabasen endast för att lagra information om vilka lexem som tillhör ett visst lemma, utom ifråga om verblexem, till vilka även särdragen `PASS` och `DEPON` samt attributet `VERBCOMP` knyts. (Ett särdrag antar något av värdena `+` och `NIL`.) `PASS` anger att verbet kan förekomma i passiv form och `DEPON` anger att det är fråga om ett deponensverb. Särdragen `PASS` och `DEPON` är aktuella endast vid den morfologiska analysen. Inga detaljer kommer därför att ges här om deras användning. `VERBCOMP`-attributet anger vilken verbkompositionsregel som knyts till lemmat.

I lexembasen finns fler slag av information än i lemmabasen. Till substantiv och proprier knyts särdragen `TEMP`, `LOC`, och `STUFF`. `TEMP` anger att ordet kan vara huvudord i en nominalfras som refererar till en tidpunkt. `LOC` anger att ordet kan vara huvudord i en nominalfras som refererar till en plats. Särdraget `STUFF` ges åt substantiv som tillhör klassen *dividua*, d.v.s. ämnesnamn och en del andra ord. Till adverb knyts särdragen `TRUTH`, `ATTITUDE`, `CONJ`, `TEMP`, `LOC`. Deras innebörd och användning beskrivs närmre i avsnitt 3.5.7. Till verblexem knyts, vilket visats ovan, information om dess valens iform av en verbaktionsregel (under attributet `VERBACTION`). Till subjunktioner knyts särdraget `EXPL`, som anger att de kan fungera som bisatsinledare i en explikativ bisats.

För vissa attribut i lemma- och lexembasen har defaultvärden givits. Om ett attribut inte definierats vare sig för det aktuella lemmat eller lexemet eller bland defaultvärdena, har det värdet `NIL`, som ifråga om särdrag tolkas som ett negativt värde. Defaultvärdena har i språkbeskrivningen definierats med satsen:

```
(define default-values
  (DLEX (apply-function gen-lex))
  (PASS +)
  (VERBCOMP vc.ta))
```

För ord som tillhör vissa ordklasser, för vilka det inte varit meningsfullt, åtminstone i språkbeskrivningens nuvarande utvecklingsskede, att urskilja olika betydelser, har i den föreliggande språkbeskrivningen en en-till-entydig korrespondens mellan lemma och lexem upprätthållits. De aktuella ordklasserna är proprier, pronomen, relativpronomen, adverb, konjunktioner, subjunktioner, och prepositioner. Eftersom lexemet i dessa fall lätt kan genereras ur lemmanamnet har detta definierats som ett defaultvärde till attributet `DLEX` (Default LEXem) som (`apply-function gen-lex`). Detta innebär att när man accessar attributet `DLEX` hos ett lemma så returneras det värde som erhålls då lispfunktionen `gen-lex` appliceras på lemmanamnet. Detta värde utgörs av den atom som bildas om man appenderar `.1` till lemmanamnet och bildar således namnet på det första (och i de här fallen enda) lexemet under lemmat ifråga. Om lemmat exempelvis är `ATT.SN` så är motsvarande lexem (det värde som `gen-lex` returnerar) `ATT.SN.1`. De övriga två defaultdefinierade attributen är särdraget `PASS`, som ges positivt värde (eftersom de flesta verb kan passiveras) och `VERBCOMP`, vars defaultvärde är `vc.ta`.

Innehållet i det minimala lexikon som ingår LPS-0's språkbeskrivning är för alla ordklasser utom verben tämligen trivialt.

Vad gäller verben har den lemma- och lexemindelning som görs i Svensk ordbok (1986) behållits. I många fall har dock inte alla lexem tagits upp i LPS-0's lexikon. De verbaktionsregler som knutits till verblexemen svarar mot en typisk konstruktion eller ett antal typiska konstruktioner i vilket lexemet brukar förekomma, men inte nödvändigtvis alla. Eftersom ett omfattande och systematiskt arbete med framtagandet av lexikon pågår inom LPS-projektet och eftersom det föreliggande arbetets syfte främst varit att utveckla en grammatisk beskrivning och lexikonet endast tjänat som ett underlag för testning av denna, har ingen större möda ägnats formuleringen av detta.

3.7 Ett antal exempel på analyser genererade av LPS-0

Som illustration till LPS-0 ges här trettiofyra exempel på analyser som genererats av denna parser.

1) KALLE SPRINGER.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
             GENDER = UTR
             NUMB = SING
             PROPR = +
             HEAD = (LEX = KALLE.PM.1
                    WORD.CAT = NOUN)
             REF = +)
      SUBJ = <* FUND>
      PRED = (LEX = SPRINGA.VB.1)
      TENSE = PRES
      SEP = .))
```

Detta exempel visar den enklaste typen av deklarativ huvudsats, vilken består av ett subjekt och ett predikat, som utgörs av ett intransitivt verb. I analysen anges fraskategori (PHR.CAT) som sats (CL; clause), satstyp (TYPE) som huvudsats (MAIN) och modus (MODE) som deklarativt (DECL). Fundamentet (FUND) utgörs av nominalfrasen *Kalle*, som också är subjekt (SUBJ) i satsen. Fundamentet och subjektet är alltså identiska. För predikatet (PRED) anges lexemtillhörighet (LEX). Tempus (TENSE) är presens (PRES). Det avslutande skiljetecknet (SEP; separator) utgörs av en punkt (.).

2) KALLE HAR SPRUNGIT.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
```

```

FUND = (PHR.CAT = NP
        GENDER = UTR
        NUMB = SING
        PROPR = +
        HEAD = (LEX = KALLE.PM.1
                WORD.CAT = NOUN)
        REF = +)
SUBJ = <* FUND>
PRED = (LEX = SPRINGA.VB.1)
TENSE = PERF
SEP = .))

```

Detta exempel visar att sammansatta tempus hanteras på samma sätt som enkla: Det enda som skiljer denna analys från exempel 1), är att tempus här angivits som perfekt (PERF).

3) LISA HÅLLER AV KALLE.

```

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = LISA.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      SUBJ = <* FUND>
      PRED = (LEX = HALLA.VB+AV.PL.1)
      TENSE = PRES
      OBJ.DIR = (PHR.CAT = NP
                 GENDER = UTR
                 NUMB = SING
                 PROPR = +
                 HEAD = (LEX = KALLE.PM.1
                         WORD.CAT = NOUN)
                 REF = +)
      SEP = .))

```

Detta exempel är avsett att demonstrera hanteringen av partikelverb. Analysen av en sats där predikatet utgörs av ett sådant får samma struktur som i de fall predikatet är ett enkelt verb. Partikelverbslemmat (HÅLLA.AV+AV.PL) genereras genom att verblemmat (HÅLLA.VB) och verbpartikkellemmat (AV.PL) och ett mellanliggande plustecken konkateneras. Predikatet tillhör det första (och enda) lexemet (HÅLLA.VB+AV.PL.1) under detta lemma. Detta predikat tar förutom subjekt också ett direkt objekt (OBJ.DIR) som argument.

4) SOLDATERNA PLUNDRADE STADEN.

```

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              FORM = DEF
              GENDER = UTR
              NUMB = PLUR
              HEAD = (WORD.CAT = NOUN
                      LEX = SOLDAT.NN.1)
              SPEC = <* FUND FORM>
              REF = +)
      SUBJ = (PHR.CAT = NP
              FORM = DEF
              GENDER = UTR
              NUMB = SING
              HEAD = (WORD.CAT = NOUN
                      LEX = STAD.NN.1)
              SPEC = <* SUBJ FORM>
              REF = +)
      PRED = (LEX = PLUNDRA.VB.1)
      TENSE = PRET
      OBJ.DIR = <* FUND>
      SEP = .))

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              FORM = DEF
              GENDER = UTR
              NUMB = PLUR
              HEAD = (WORD.CAT = NOUN
                      LEX = SOLDAT.NN.1)
              SPEC = <* FUND FORM>
              REF = +)
      SUBJ = <* FUND>
      PRED = (LEX = PLUNDRA.VB.1)
      TENSE = PRET
      OBJ.DIR = (PHR.CAT = NP
                 FORM = DEF
                 GENDER = UTR
                 NUMB = SING
                 HEAD = (WORD.CAT = NOUN
                         LEX = STAD.NN.1)
                 SPEC = <* OBJ.DIR FORM>
                 REF = +)

```

SEP = .))

När ett enkelt verb tar både subjekt och direkt objekt uppstår en tvetydighet: Den första läsningen svarar mot fallet att det direkta objektet är topikaliserat och den andra, som är den ordinära läsningen, att subjektet står i fundamentalsposition. I den första analysen är således *staden* subjekt och *soldaterna* direkt objekt. Eftersom plundra normalt kräver ett animat subjekt kunde ett utnyttjande av selektionsrestriktioner vid analysen utesluta denna läsning. I den andra analysen är *soldaterna* subjekt och *staden* direkt objekt, vilket naturligtvis motsvarar den semantiskt sett mest sannolika läsningen.

5) LISA PRENUMERERAR PÅ GÖTEBORGS-POSTEN.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = LISA.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      SUBJ = < * FUND >
      PRED = (LEX = PRENUMERERA.VB.1)
      TENSE = PRES
      OBJ.PREP = (PHR.CAT = PP
                  PREP = (LEX = PA.PP.1)
                  RECT = (PHR.CAT = NP
                          GENDER = UTR
                          NUMB = SING
                          PROPR = +
                          HEAD = (LEX = GÖTEBORGS-POSTEN.PM.1
                                  WORD.CAT = NOUN)
                          REF = +))
      SEP = .))
```

Exemplet visar en analys av en mening där prediktet tagit ett prepositionsobjekt (OBJ.PREP) som argument. Detta utgörs av en prepositionsfras: *på Göteborgs-Posten*.

6) PÅ GÖTEBORGS-POSTEN PRENUMERERAR LISA.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = PP
              PREP = (LEX = PA.PP.1)
              RECT = (PHR.CAT = NP
```

```

                                GENDER = UTR
                                NUMB = SING
                                PROPR = +
                                HEAD = (LEX = GÖTEBORGS-POSTEN.PM.1
                                        WORD.CAT = NOUN)
                                REF = +))
SUBJ = (PHR.CAT = NP
        GENDER = UTR
        NUMB = SING
        PROPR = +
        HEAD = (LEX = LISA.PM.1
                WORD.CAT = NOUN)
        REF = +)
PRED = (LEX = PRENUMERERA.VB.1)
TENSE = PRES
OBJ.PREP = < * FUND >
SEP = .))

```

Ett prepositionsobjekt kan topikaliseras.

7) GÖTEBORGS-POSTEN PRENUMERERAR LISA PÅ.

```

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = GÖTEBORGS-POSTEN.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      SUBJ = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = LISA.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      PRED = (LEX = PRENUMERERA.VB.1)
      TENSE = PRES
      OBJ.PREP = (PHR.CAT = PP
                  PREP = (LEX = PA.PP.1)
                  RECT = < * FUND >)
      SEP = .))

```

Exemplet visar analysen av en mening där prepositionsobjektet splittrats på så sätt att prepositionen står på prepositionsobjektets normala plats, medan dess rektion

topikaliserats.

8) MANNEN TAR PÅ HATTEN.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              FORM = DEF
              GENDER = UTR
              NUMB = SING
              HEAD = (WORD.CAT = NOUN
                      LEX = MAN.NN.1)
              SPEC = < * FUND FORM >
              REF = +)
      SUBJ = < * FUND >
      PRED = (LEX = TA.VB.4)
      TENSE = PRES
      OBJ.PREP = (PHR.CAT = PP
                  PREP = (LEX = PA.PP.1)
                  RECT = (PHR.CAT = NP
                          FORM = DEF
                          GENDER = UTR
                          NUMB = SING
                          HEAD = (WORD.CAT = NOUN
                                  LEX = HATT.NN.1)
                          SPEC = < * OBJ.PREP RECT FORM >
                          REF = +)
                  SEP = .))
```

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              FORM = DEF
              GENDER = UTR
              NUMB = SING
              HEAD = (WORD.CAT = NOUN
                      LEX = MAN.NN.1)
              SPEC = < * FUND FORM >
              REF = +)
      SUBJ = < * FUND >
      PRED = (LEX = TA.VB+PA.PL.1)
      TENSE = PRES
      OBJ.DIR = (PHR.CAT = NP
                 FORM = DEF
                 GENDER = UTR
```

```

NUMB = SING
HEAD = (WORD.CAT = NOUN
        LEX = HATT.NN.1)
SPEC = <* OBJ.DIR FORM>
REF = +)
SEP = .))

```

Denna mening exemplifierar en typ av ambiguitet som kan uppstå då ett ord kan tolkas både som preposition och verbpartikel. I första analysen är det det enkla verbet (TA.VB.4) som är predikat. Det tar ett prepositionsobjekt, här *på hatten*, som argument och *på* fungerar sålunda som preposition. Den andra analysen motsvarar fallet att *på* fungerar som verbpartikel. Predikatet utgörs då av partikel verbet (TA.VB+PÅ.PL.1), som tar ett direkt objekt *hatten*. Denna ambiguitet är egentligen möjlig bara i skreven svenska. I talat språk skiljs de båda läsningarna åt genom prosodin: En preposition är normalt obetonad medan en verbpartikel betonas ganska kraftigt.

9) KALLE GAV LISA BOKEN.

```

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = KALLE.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      SUBJ = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = LISA.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      PRED = (LEX = GE.VB.1)
      TENSE = PRET
      OBJ.INDIR = (PHR.CAT = NP
                   FORM = DEF
                   GENDER = UTR
                   NUMB = SING
                   HEAD = (WORD.CAT = NOUN
                           LEX = BOK.NN.1)
                   SPEC = <* OBJ.INDIR FORM>
                   REF = +)
      OBJ.DIR = <* FUND>

```

SEP = .))

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = KALLE.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      SUBJ = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = LISA.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      PRED = (LEX = GE.VB.1)
      TENSE = PRET
      OBJ.INDIR = < * FUND >
      OBJ.DIR = (PHR.CAT = NP
                 FORM = DEF
                 GENDER = UTR
                 NUMB = SING
                 HEAD = (WORD.CAT = NOUN
                          LEX = BOK.NN.1)
                 SPEC = < * OBJ.DIR FORM >
                 REF = +)
      SEP = .))
```

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = KALLE.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      SUBJ = < * FUND >
      PRED = (LEX = GE.VB.1)
      TENSE = PRET
      OBJ.INDIR = (PHR.CAT = NP
```

```

GENDER = UTR
NUMB = SING
PROPR = +
HEAD = (LEX = LISA.PM.1
        WORD.CAT = NOUN)
REF = +)
OBJ.DIR = (PHR.CAT = NP
           FORM = DEF
           GENDER = UTR
           NUMB = SING
           HEAD = (WORD.CAT = NOUN
                  LEX = BOK.NN.1)
           SPEC = <* OBJ.DIR FORM>
           REF = +)
SEP = .))

```

Detta är ett exempel på en trefaldig ambiguitet. Denna uppstår eftersom verbet tar tre argument (subjekt, direkt objekt och indirekt objekt) och eftersom grammatiken tillåter att vart och ett av dessa topikaliseras. En ytterligare förutsättning är att nominalfraserna som står i de aktuella funktionerna inte är markerade för kasus, i vilket fall trefaldigheten reduceras till tve- eller entydighet. I första analysen är *Lisa* subjekt, *boken* indirekt objekt (OBJ.INDIR), och *Kalle* direkt objekt. Denna läsning skulle givetvis kunna uteslutas (åtminstone så gott som) på semantiska grunder. Den andra analysen svarar mot en läsning där *Lisa* är subjekt, *Kalle* är indirekt objekt, och *boken* är direkt objekt. Denna läsning är naturligtvis helt acceptabel ur semantisk synvinkel, ehuru den fördens skull knappast är särskilt sannolik. Den läsning som man i de allra flesta fall skulle ge åt meningen representeras av den tredje och sista analysen, där *Kalle* är subjekt, *Lisa* indirekt objekt och *boken* direkt objekt.

10) KALLE GAV BOKEN TILL LISA.

```

(* = (PHR.CAT = CL
     TYPE = MAIN
     MODE = DECL
     FUND = (PHR.CAT = NP
            GENDER = UTR
            NUMB = SING
            PROPR = +
            HEAD = (LEX = KALLE.PM.1
                   WORD.CAT = NOUN)
            REF = +)
     SUBJ = (PHR.CAT = NP
            FORM = DEF
            GENDER = UTR
            NUMB = SING
            HEAD = (WORD.CAT = NOUN

```

```

                LEX = BOK.NN.1)
            SPEC = <* SUBJ FORM>
            REF = +)
    PRED = (LEX = GE.VB.1)
    TENSE = PRET
    OBJ.DIR = <* FUND>
    OBJ.PREP = (PHR.CAT = PP
                PREP = (LEX = TILL.PP.1)
                RECT = (PHR.CAT = NP
                        GENDER = UTR
                        NUMB = SING
                        PROPR = +
                        HEAD = (LEX = LISA.PM.1
                                WORD.CAT = NOUN)
                        REF = +))
    SEP = .))

```

```

(* = (PHR.CAT = CL
     TYPE = MAIN
     MODE = DECL
     FUND = (PHR.CAT = NP
             GENDER = UTR
             NUMB = SING
             PROPR = +
             HEAD = (LEX = KALLE.PM.1
                     WORD.CAT = NOUN)
             REF = +)
     SUBJ = <* FUND>
     PRED = (LEX = GE.VB.1)
     TENSE = PRET
     OBJ.DIR = (PHR.CAT = NP
               FORM = DEF
               GENDER = UTR
               NUMB = SING
               HEAD = (WORD.CAT = NOUN
                       LEX = BOK.NN.1)
               SPEC = <* OBJ.DIR FORM>
               REF = +)
     OBJ.PREP = (PHR.CAT = PP
                 PREP = (LEX = TILL.PP.1)
                 RECT = (PHR.CAT = NP
                         GENDER = UTR
                         NUMB = SING
                         PROPR = +
                         HEAD = (LEX = LISA.PM.1
                                 WORD.CAT = NOUN)

```

REF = +))

SEP = .))

Med avseende på de i normala fall föredragna läsningarna är denna mening och den föregående synonyma. Eftersom prepositionsfrasen här har samma relation, semantiskt sett, som det indirekta objektet har till verbet, klassificerar många grammatiker en sådan prepositionsfras som ett indirekt objekt, så t.ex. Thorell (1973). I den här presenterade grammatiken har dock en mer ytinriktat synsätt valts, enligt vilket prepositionsfrasen analyseras som ett prepositionsobjekt.

11) DET BÖRJAR NU.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              CASE = BASIC
              GENDER = NEUTR
              NUMB = SING
              REF = +
              HEAD = (LEX = DEN.PN.1
                      WORD.CAT = PRON))
      SUBJ = <* FUND>
      PRED = (LEX = BORJA.VB.1)
      TENSE = PRES
      OBJ.TEMP = (PHR.CAT = ADVP
                  HEAD = (LEX = NU.AB.1))
      SEP = .))
```

Meningen ger exempel på ett temporalobjekt OBJ.TEMP, som utgörs av ett *nu*.

12) NU BÖRJAR DET.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = ADVP
              HEAD = (LEX = NU.AB.1))
      SUBJ = (PHR.CAT = NP
              CASE = BASIC
              GENDER = NEUTR
              NUMB = SING
              REF = +
              HEAD = (LEX = DEN.PN.1
                      WORD.CAT = PRON))
      PRED = (LEX = BORJA.VB.1)
      TENSE = PRES
      OBJ.TEMP = <* FUND>
      SEP = .))
```

Ett temporalobjekt kan, som de flesta typer av verbargument, placeras i fundamentsposition.

13) ELVATIDEN BÖRJAR DET VID.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              FORM = DEF
              GENDER = UTR
              NUMB = SING
              HEAD = (WORD.CAT = NOUN
                      LEX = ELVATID.NN.1)
              SPEC = < * FUND FORM >
              REF = +)
      SUBJ = (PHR.CAT = NP
              CASE = BASIC
              GENDER = NEUTR
              NUMB = SING
              REF = +
              HEAD = (LEX = DEN.PN.1
                      WORD.CAT = PRON))
      PRED = (LEX = BORJA.VB.1)
      TENSE = PRES
      OBJ.TEMP = (PHR.CAT = PP
                  PREP = (LEX = VID.PP.1)
                  RECT = < * FUND >)
      SEP = .))
```

Ett temporalobjekt kan realiseras som en prepositionsfras. Denna kan, som i denna mening, splittras på samma sätt som ett prepositionsobjekt.

14) EN MAN BOR I HUSET.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              FORM = INDEF
              GENDER = UTR
              NUMB = SING
              SPEC = INDEF
              DETER = (LEX = EN.AL.1)
              HEAD = (WORD.CAT = NOUN
                      LEX = MAN.NN.1)
              REF = +)
      SUBJ = < * FUND >
```

```

PRED = (LEX = BO.VB.1)
TENSE = PRES
OBJ.LOC = (PHR.CAT = PP
            PREP = (LEX = I.PP.1)
            RECT = (PHR.CAT = NP
                    FORM = DEF
                    GENDER = NEUTR
                    NUMB = SING
                    HEAD = (WORD.CAT = NOUN
                            LEX = HUS.NN.1)
                    SPEC = <* OBJ.LOC RECT FORM>
                    REF = +))
SEP = .))

```

Meningen ger exempel på ett lokationsobjekt (OBJ.LOC). Detta utgörs här av en sammanhållen prepositionsfras *i huset*.

15) HÄR BOR KALLE.

```

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = ADVP
              HEAD = (LEX = HAR.AB.1))
      SUBJ = (PHR.CAT = NP
              GENDER = UTR
              NUMB = SING
              PROPR = +
              HEAD = (LEX = KALLE.PM.1
                      WORD.CAT = NOUN)
              REF = +)
      PRED = (LEX = BO.VB.1)
      TENSE = PRES
      OBJ.LOC = <* FUND>
      SEP = .))

```

Lokationsobjekt kan även utgöras av adverbfraser och topikaliseras.

16) DET BOR EN MAN I HUSET.

```

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              HEAD = (LEX = DET.PN.1))
      SUBJ.FORM = <* FUND>
      PRED = (LEX = BO.VB.1)
      TENSE = PRES
      SUBJ.PROPER = (PHR.CAT = NP

```

```

FORM = INDEF
GENDER = UTR
NUMB = SING
SPEC = INDEF
DETER = (LEX = EN.AL.1)
HEAD = (WORD.CAT = NOUN
        LEX = MAN.NN.1)
REF = +)
OBJ.LOC = (PHR.CAT = PP
PREP = (LEX = I.PP.1)
RECT = (PHR.CAT = NP
        FORM = DEF
        GENDER = NEUTR
        NUMB = SING
        HEAD = (WORD.CAT = NOUN
                LEX = HUS.NN.1)
        SPEC = <* OBJ.LOC RECT FORM>
        REF = +))
SEP = .))

```

Vissa verb, exempelvis sådana som betecknar något slag av befintlighet, kan, istället för att förekomma med vanligt subjekt, ingå i en konstruktion med ett formellt subjekt (SUBJ.FORM) plus ett egentligt subjekt (SUBJ.PROPER). Ett formellt subjekt utgörs alltid av ett platshållande *det*. Det egentliga subjektet är alltid en nominalfras i obestämd form. Ett formellt subjekt förekommer alltid tillsammans med ett egentligt subjekt, och vice versa.

17) LISA ANTYDER ATT MANNEN HAR TAGIT PENGARNA.

```

(* = (PHR.CAT = CL
TYPE = MAIN
MODE = DECL
FUND = (PHR.CAT = NP
        GENDER = UTR
        NUMB = SING
        PROPR = +
        HEAD = (LEX = LISA.PM.1
                WORD.CAT = NOUN)
        REF = +)
SUBJ = <* FUND>
PRED = (LEX = ANTYDA.VB.2)
TENSE = PRES
OBJ.EXPL = (PHR.CAT = CL
            TYPE = SUBORD
            SUBTYPE = EXPL
            INTROD = (LEX = ATT.SN.1)
            SUBJ = (PHR.CAT = NP

```

```

FORM = DEF
GENDER = UTR
NUMB = SING
HEAD = (WORD.CAT = NOUN
        LEX = MAN.NN.1)
SPEC = <* OBJ.EXPL SUBJ FORM>
REF = +)
PRED = (LEX = TA.VB.1)
TENSE = PERF
OBJ.DIR = (PHR.CAT = NP
           FORM = DEF
           GENDER = UTR
           NUMB = PLUR
           HEAD = (WORD.CAT = NOUN
                  LEX = PENG.NN.1)
           SPEC = <* OBJ.EXPL OBJ.DIR FORM>
           REF = +))

```

SEP = .))

Verblexemet ANTYDA.VB.2 konstrueras med ett explikativobjekt (OBJ.EXPL). Detta består av en explikativ (SUBTYPE=EXPL) bisats (PHR.CAT=CL och TYPE=SUBORD; subordinate). Dess bisatsinledare (INTROD; introducer) består av ett *att*.

18) HAN BÖRJAR SJUNGA.

```

(* = (PHR.CAT = CL
     TYPE = MAIN
     MODE = DECL
     FUND = (PHR.CAT = NP
            CASE = NOM
            GENDER = UTR
            NUMB = SING
            REF = +
            HEAD = (LEX = HAN.PN.1
                   WORD.CAT = PRON))
     SUBJ = <* FUND>
     PRED = (LEX = BORJA.VB.2)
     TENSE = PRES
     OBJ.INF = (SUBJ = <* FUND>
               PHR.CAT = CL
               TYPE = INF
               PRED = (LEX = SJUNGA.VB.1))
     SEP = .))

```

Verbet tar i detta exempel ett infinitivobjekt som argument. Detta utgörs av en infinitivfras (PHR.CAT=CL och TYPE=INF). Subjektet i den överordnade satsen blir också subjekt i den underordnade infinitivfrasen.

19) HON TVINGADE HONOM ATT SJUNGA.

```

(*) = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              CASE = NOM
              GENDER = UTR
              NUMB = SING
              REF = +
              HEAD = (LEX = HON.PN.1
                      WORD.CAT = PRON))
      SUBJ = <* FUND>
      PRED = (LEX = TVINGA.VB.1)
      TENSE = PRET
      OBJ.DIR = (PHR.CAT = NP
                 CASE = OBL
                 GENDER = UTR
                 NUMB = SING
                 REF = +
                 HEAD = (LEX = HAN.PN.1
                         WORD.CAT = PRON))
      OBJ.INF = (SUBJ = <* OBJ.DIR>
                 PHR.CAT = CL
                 TYPE = INF
                 INF.MARK = ATT.SN.1
                 PRED = (LEX = SJUNGA.VB.1))
      SEP = .))

```

I detta fall, där verbet *tvinga* är predikat i den överordnade satsen, förmedlas det direkta objektet i denna som subjekt i den underordnade infinitivfrasen. I denna är infinitivmärket (INF.MARK) utsatt.

20) BOKEN FINNS ATT LÄSA I BIBLIOTEKET.

```

(*) = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              FORM = DEF
              GENDER = UTR
              NUMB = SING
              HEAD = (WORD.CAT = NOUN
                      LEX = BOK.NN.1)
              SPEC = <* FUND FORM>
              REF = +)
      SUBJ = <* FUND>
      PRED = (LEX = FINNAS.VB.1)
      TENSE = PRES

```

```

OBJ.INF = (OBJ.DIR = <* FUND>
           PHR.CAT = CL
           TYPE = INF
           INF.MARK = ATT.SN.1
           PRED = (LEX = LASA.VB.1))
OBJ.LOC = (PHR.CAT = PP
           PREP = (LEX = I.PP.1)
           RECT = (PHR.CAT = NP
                  FORM = DEF
                  GENDER = NEUTR
                  NUMB = SING
                  HEAD = (WORD.CAT = NOUN
                         LEX = BIBLIOTEK.NN.1)
                  SPEC = <* OBJ.LOC RECT FORM>
                  REF = +))
SEP = .))

```

Meningen ger exempel på ett fall där ett subjekt i en överordnad sats förmedlas som direkt objekt i en infinitivfras.

21) STADEN PLUNDRADES AV SOLDATERNA.

```

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
             FORM = DEF
             GENDER = UTR
             NUMB = SING
             HEAD = (WORD.CAT = NOUN
                    LEX = STAD.NN.1)
             SPEC = <* FUND FORM>
             REF = +)
      PASS = +
      SUBJ = <* FUND>
      PRED = (LEX = PLUNDRA.VB.1)
      TENSE = PRET
      AGENT = (PHR.CAT = PP
             PREP = (LEX = AV.PP.1)
             RECT = (PHR.CAT = NP
                    FORM = DEF
                    GENDER = UTR
                    NUMB = PLUR
                    HEAD = (WORD.CAT = NOUN
                           LEX = SOLDAT.NN.1)
                    SPEC = <* AGENT RECT FORM>
                    REF = +))
      SEP = .))

```

Detta är den passiverade varianten av exempel 4) (i dess normala läsning). Passiv diates anges i analysen med särdraget PASS. I en normal passiv sats motsvarar agenten (AGENT) subjektet vid aktiv diates och subjektet motsvarar det direkta objektet vid aktiv diates.

22) MANNEN ANTYDS HA TAGIT PENGARNA.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              FORM = DEF
              GENDER = UTR
              NUMB = SING
              HEAD = (WORD.CAT = NOUN
                      LEX = MAN.NN.1)
              SPEC = < * FUND FORM >
              REF = +)
      PASS = +
      SUBJ = < * FUND >
      PRED = (LEX = ANTYDA.VB.2)
      TENSE = PRES
      OBJ.INF = (SUBJ = < * FUND >
                 PHR.CAT = CL
                 TYPE = INF
                 PRED = (LEX = TA.VB.1)
                 TENSE = PERF.INF
                 OBJ.DIR = (PHR.CAT = NP
                            FORM = DEF
                            GENDER = UTR
                            NUMB = PLUR
                            HEAD = (WORD.CAT = NOUN
                                    LEX = PENG.NN.1)
                            SPEC = < * OBJ.INF OBJ.DIR FORM >
                            REF = +))
      SEP = .))
```

Om vi uppfattar denna mening som en passiv variant av 17), så har vi här ett exempel på en avvikande typ av passivering, där ett explikativt objekt vid aktiv diates splittrats vid passiveringen i ett subjekt och ett infinitivobjekt, vars subjekt är identiskt med subjektet i den överordnade satsen.

23) KALLE AVSATTES FRÅN STATSRÅDSPOSTEN.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
```

```

        GENDER = UTR
        NUMB = SING
        PROPR = +
        HEAD = (LEX = KALLE.PM.1
                WORD.CAT = NOUN)
        REF = +)
    PASS = +
    SUBJ = <* FUND>
    PRED = (LEX = AVSATTA.VB.1)
    TENSE = PRET
    OBJ.PREP = (PHR.CAT = PP
                PREP = (LEX = FRAN.PP.1)
                RECT = (PHR.CAT = NP
                        FORM = DEF
                        GENDER = UTR
                        NUMB = SING
                        HEAD = (WORD.CAT = NOUN
                                LEX = STATSRADSPOST.NN.1)
                        SPEC = <* OBJ.PREP RECT FORM>
                        REF = +))
    SEP = .))

```

Ett prepositionsobjekt påverkas normalt inte vid passivering.

24) KALLE TVINGADES ATT SPRINGA.

```

    (* = (PHR.CAT = CL
        TYPE = MAIN
        MODE = DECL
        FUND = (PHR.CAT = NP
                GENDER = UTR
                NUMB = SING
                PROPR = +
                HEAD = (LEX = KALLE.PM.1
                        WORD.CAT = NOUN)
                REF = +)
        PASS = +
        SUBJ = <* FUND>
        PRED = (LEX = TVINGA.VB.1)
        TENSE = PRET
        OBJ.INF = (SUBJ = <* FUND>
                  PHR.CAT = CL
                  TYPE = INF
                  INF.MARK = ATT.SN.1
                  PRED = (LEX = SPRINGA.VB.1))
        SEP = .))

```

När verbet *twinga* konstrueras med direkt objekt och infinitivobjekt i aktiv diates,

som i exempel 19), förmedlas det direkta objektet som subjekt i infinitivfrasen. Vid passiv diates är, följdriktigt, subjekten i den överordnade satsen och infinitivfrasen identiska.

25) HAN KUNDE TYVÄRR INTE SJUNGA IGÅR.

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              CASE = NOM
              GENDER = UTR
              NUMB = SING
              REF = +
              HEAD = (LEX = HAN.PN.1
                      WORD.CAT = PRON))
      SUBJ = <* FUND>
      ADV.ATTITUDE11 = (PHR.CAT = ADVP
                        HEAD = (LEX = TYVARR.AB.1))
      ADV.TRUTH11 = (PHR.CAT = ADVP
                     HEAD = (LEX = INTE.AB.1))
      PRED = (LEX = KUNNA.VB.1)
      TENSE = PRET
      OBJ.INF = (SUBJ = <* FUND>
                 PHR.CAT = CL
                 TYPE = INF
                 PRED = (LEX = SJUNGA.VB.1)
                 ADV.TEMP21 = (PHR.CAT = ADVP
                               HEAD = (LEX = IGAR.AB.1)))
      SEP = .))
```

```
(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = DECL
      FUND = (PHR.CAT = NP
              CASE = NOM
              GENDER = UTR
              NUMB = SING
              REF = +
              HEAD = (LEX = HAN.PN.1
                      WORD.CAT = PRON))
      SUBJ = <* FUND>
      ADV.ATTITUDE11 = (PHR.CAT = ADVP
                        HEAD = (LEX = TYVARR.AB.1))
      ADV.TRUTH11 = (PHR.CAT = ADVP
                     HEAD = (LEX = INTE.AB.1))
      PRED = (LEX = KUNNA.VB.1)
```

```

TENSE = PRET
OBJ.INF = (SUBJ = <* FUND>
           PHR.CAT = CL
           TYPE = INF
           PRED = (LEX = SJUNGA.VB.1))
ADV.TEMP21 = (PHR.CAT = ADVP
              HEAD = (LEX = IGAR.AB.1))
SEP = .))

```

Denna mening innehåller tre adverbial. Ett talarattitydsadverbial (ADV.ATTITUDE11) *tyvärr* och ett sanningsvärdesadverbial (ADV.TRUTH11) *inte* står i huvudsatsens första adverbialsgrupp. Det tredje adverbialet, ett tidsadverbial *igår*, ger upphov till en ambiguitet eftersom det kan tolkas som hörande till antingen infinitivfrasen eller huvudsatsen. I båda fallen står det i den andra (finala) adverbialsgruppen och för således namnet ADV.TEMP21.

26) UTAN ATT HON MÄRKTE DET BÖRjade HAN SJUNGA.

```

(* = (PHR.CAT = CL
     TYPE = MAIN
     MODE = DECL
     FUND = (PHR.CAT = PP
            PREP = (LEX = UTAN.PP.1)
            RECT = (PHR.CAT = CL
                   TYPE = SUBORD
                   SUBTYPE = EXPL
                   INTROD = (LEX = ATT.SN.1)
                   SUBJ = (PHR.CAT = NP
                          CASE = NOM
                          GENDER = UTR
                          NUMB = SING
                          REF = +
                          HEAD = (LEX = HON.PN.1
                                 WORD.CAT = PRON))
                   PRED = (LEX = MARKA.VB.3)
                   TENSE = PRET
                   OBJ.DIR = (PHR.CAT = NP
                              CASE = BASIC
                              GENDER = NEUTR
                              NUMB = SING
                              REF = +
                              HEAD = (LEX = DEN.PN.1
                                     WORD.CAT = PRON))))
     SUBJ = (PHR.CAT = NP
            CASE = NOM
            GENDER = UTR
            NUMB = SING

```

```

REF = +
HEAD = (LEX = HAN.PN.1
        WORD.CAT = PRON))
PRED = (LEX = BORJA.VB.2)
TENSE = PRET
OBJ.INF = (SUBJ = <* SUBJ>
           PHR.CAT = CL
           TYPE = INF
           PRED = (LEX = SJUNGA.VB.1))
ADV.CIRC = <* FUND>
SEP = .))

```

I ovanstående mening analyseras *Utan att hon märkte det* som en prepositionsfras där prepositionen styr en explikativ bisats. (Thorell (1973) tolkar *utan att* som en sammansatt subjunktion. Han skulle sålunda kalla den fras som här analyserats som en prepositionsfras för en bisats.) Den tilldelas funktionen omständighetsadverbial (ADV.CIRC; circumstance).

27) SPRING!

```

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = IMP
      PRED = (LEX = SPRINGA.VB.1)))

```

En imperativ (MODE=IMP) huvudsats. Analysen visar vilka lättlästa och lättöverskådliga analyser LPS-0 genererar.

28) TA INTE PÅ HATTEN!

```

(* = (PHR.CAT = CL
      TYPE = MAIN
      MODE = IMP
      ADV.TRUTH11 = (PHR.CAT = ADVP
                    HEAD = (LEX = INTE.AB.1))
      PRED = (LEX = TA.VB.4)
      OBJ.PREP = (PHR.CAT = PP
                 PREP = (LEX = PA.PP.1)
                 RECT = (PHR.CAT = NP
                        FORM = DEF
                        GENDER = UTR
                        NUMB = SING
                        HEAD = (WORD.CAT = NOUN
                               LEX = HATT.NN.1)
                        SPEC = <* OBJ.PREP RECT FORM>
                        REF = +))))

```

```

(* = (PHR.CAT = CL
      TYPE = MAIN

```

```

MODE = IMP
ADV.TRUTH11 = (PHR.CAT = ADVP
                HEAD = (LEX = INTE.AB.1))
PRED = (LEX = TA.VB+PA.PL.1)
OBJ.DIR = (PHR.CAT = NP
           FORM = DEF
           GENDER = UTR
           NUMB = SING
           HEAD = (WORD.CAT = NOUN
                  LEX = HATT.NN.1)
           SPEC = <* OBJ.DIR FORM>
           REF = +)))

```

Samma ambiguitet som i exempel 8), fast i imperativ version.

29) BOR DET EN MAN HÄR?

```

(* = (PHR.CAT = CL
     TYPE = MAIN
     MODE = INTERR
     QUESTIONTYPE = YES-NO
     SUBJ.FORM = (PHR.CAT = NP
                 HEAD = (LEX = DET.PN.1))
     PRED = (LEX = BO.VB.1)
     TENSE = PRES
     SUBJ.PROPER = (PHR.CAT = NP
                   FORM = INDEF
                   GENDER = UTR
                   NUMB = SING
                   SPEC = INDEF
                   DETER = (LEX = EN.AL.1)
                   HEAD = (WORD.CAT = NOUN
                          LEX = MAN.NN.1)
                   REF = +)
     OBJ.LOC = (PHR.CAT = ADVP
               HEAD = (LEX = HAR.AB.1))))

```

Ett exempel på en ja-nej-fråga (MODE=INTERR och QUESTIONTYPE=YES-NO).

30) FÖRBJÖD HON HONOM ATT LÄSA BOKEN?

```

(* = (PHR.CAT = CL
     TYPE = MAIN
     MODE = INTERR
     QUESTIONTYPE = YES-NO
     SUBJ = (PHR.CAT = NP
            CASE = NOM
            GENDER = UTR
            NUMB = SING

```

```

REF = +
HEAD = (LEX = HON.PN.1
        WORD.CAT = PRON))
PRED = (LEX = FORBJUDA.VB.2)
TENSE = PRET
OBJ.DIR = (PHR.CAT = NP
           CASE = OBL
           GENDER = UTR
           NUMB = SING
           REF = +
           HEAD = (LEX = HAN.PN.1
                  WORD.CAT = PRON))
OBJ.INF = (SUBJ = <* OBJ.DIR>
           PHR.CAT = CL
           TYPE = INF
           INF.MARK = ATT.SN.1
           PRED = (LEX = LASA.VB.1)
           OBJ.DIR = (PHR.CAT = NP
                    FORM = DEF
                    GENDER = UTR
                    NUMB = SING
                    HEAD = (WORD.CAT = NOUN
                           LEX = BOK.NN.1)
                    SPEC = <* OBJ.INF OBJ.DIR FORM>
                    REF = +))))

```

Ytterligare en ja-nej-fråga.

31) MANNEN SOM FÖRSER HENNE MED BÖCKER BOR I FINLAND.

```

(* = (PHR.CAT = CL
     TYPE = MAIN
     MODE = DECL
     FUND = (PHR.CAT = NP
            FORM = DEF
            GENDER = UTR
            NUMB = SING
            HEAD = (WORD.CAT = NOUN
                   LEX = MAN.NN.1)
            SPEC = <* FUND FORM>
            REF = +
            ATTR.REL = (CORR = (PHR.CAT = <* FUND PHR.CAT>
                              HEAD = <* FUND HEAD>
                              FORM = <* FUND FORM>
                              GENDER = <* FUND GENDER>
                              NUMB = <* FUND NUMB>
                              REF = <* FUND REF>)
            PHR.CAT = CL
            TYPE = SUBORD
            SUBTYPE = REL
            INTROD = SOM.RP.1

```

```

SUBJ = < * FUND ATTR.REL CORR >
PRED = (LEX = FORSE.VB.1)
TENSE = PRES
OBJ.DIR = (PHR.CAT = NP
           CASE = OBL
           GENDER = UTR
           NUMB = SING
           REF = +
           HEAD = (LEX = HON.PN.1
                  WORD.CAT = PRON))
OBJ.PREP =
(PHR.CAT = PP
 PREP = (LEX = MED.PP.1)
 RECT = (PHR.CAT = NP
        FORM = INDEF
        GENDER = UTR
        NUMB = PLUR
        HEAD = (WORD.CAT = NOUN
                LEX = BOK.NN.1)
        SPEC = < * FUND ATTR.REL
                OBJ.PREP RECT FORM >
        REF = +)))))

SUBJ = < * FUND >
PRED = (LEX = BO.VB.1)
TENSE = PRES
OBJ.LOC = (PHR.CAT = PP
          PREP = (LEX = I.PP.1)
          RECT = (PHR.CAT = NP
                GENDER = NEUTR
                NUMB = SING
                PROPR = +
                HEAD = (LEX = FINLAND.PM.1
                       WORD.CAT = NOUN)
                REF = +))

SEP = .))

```

I denna analys visas hur relativa (SUBTYPE=REL) bisatser hanteras. Relativsatsen fungerar här som ett attribut (REL.ATTR) till den nominalfras som står i fundamentalspositionen. Korrelatet (CORR) till relativsatsen utgör ett utsnitt av den nominalfras den modifierar. Detta utsnitt ärver de grammatiska dragen samt huvudordet, och bildar i sig en nominalfras. Detta korrelat tilldelas en funktion i relativsatsen. I detta fall är det subjekt. Enligt den traditionella synen bland grammatiker, som exempelvis representeras av Torell (1973), är det dock relativpronomenet som i detta fall skulle tolkas som subjekt. Här har alltså en annan analys valts.

32) Huset som han bor i plundrares igår.

```

(* = (PHR.CAT = CL
     TYPE = MAIN
     MODE = DECL
     FUND = (PHR.CAT = NP
           FORM = DEF
           GENDER = NEUTR
           NUMB = SING
           HEAD = (WORD.CAT = NOUN

```

```

          LEX = HUS.NN.1)
SPEC = < * FUND FORM>
REF = +
ATTR.REL = (CORR = (PHR.CAT = < * FUND PHR.CAT>
                    HEAD = < * FUND HEAD>
                    FORM = < * FUND FORM>
                    GENDER = < * FUND GENDER>
                    NUMB = < * FUND NUMB>
                    REF = < * FUND REF>))
          PHR.CAT = CL
          TYPE = SUBORD
          SUBTYPE = REL
          INTROD = SOM.RP.1
          SUBJ = (PHR.CAT = NP
                 CASE = NOM
                 GENDER = UTR
                 NUMB = SING
                 REF = +
                 HEAD = (LEX = HAN.PN.1
                        WORD.CAT = PRON))
          PRED = (LEX = BO.VB.1)
          TENSE = PRES
          OBJ.LOC = (PHR.CAT = PP
                   PREP = (LEX = I.PP.1)
                   RECT = < * FUND ATTR.REL CORR>)))

PASS = +
SUBJ = < * FUND>
PRED = (LEX = PLUNDRA.VB.1)
TENSE = PRET
ADV.TEMP21 = (PHR.CAT = ADVP
              HEAD = (LEX = IGAR.AB.1))
SEP = .))

```

I ovanstående mening har en prepositionsfras som till funktionen är ett lokationsobjekt splittrats på så vis att prepositionens rektion utgörs av korrelatet.

33) HON HAR EN HUND SOM KAN SJUNGA.

```

(* = (PHR.CAT = CL
     TYPE = MAIN
     MODE = DECL
     FUND = (PHR.CAT = NP
            CASE = NOM
            GENDER = UTR
            NUMB = SING
            REF = +
            HEAD = (LEX = HON.PN.1
                   WORD.CAT = PRON))
     SUBJ = < * FUND>
     PRED = (LEX = HA.VB.1)
     TENSE = PRES
     OBJ.DIR = (PHR.CAT = NP
               FORM = INDEF
               GENDER = UTR
               NUMB = SING
               SPEC = INDEF

```

```

DETER = (LEX = EN.AL.1)
HEAD = (WORD.CAT = NOUN
        LEX = HUND.NN.1)
REF = +
ATTR.REL = (CORR = (PHR.CAT = <* OBJ.DIR PHR.CAT>
                    HEAD = <* OBJ.DIR HEAD>
                    FORM = <* OBJ.DIR FORM>
                    GENDER = <* OBJ.DIR GENDER>
                    NUMB = <* OBJ.DIR NUMB>
                    REF = <* OBJ.DIR REF>)
            PHR.CAT = CL
            TYPE = SUBORD
            SUBTYPE = REL
            INTROD = SOM.RP.1
            SUBJ = <* OBJ.DIR ATTR.REL CORR>
            PRED = (LEX = KUNNA.VB.1)
            TENSE = PRES
            OBJ.INF = (SUBJ = <* OBJ.DIR ATTR.REL CORR>
                     PHR.CAT = CL
                     TYPE = INF
                     PRED = (LEX = SJUNGA.VB.1))))
SEP = .))

```

I denna mening agerar korrelerat subjekt både i relativsatsen och infinitivfrasen.

3.8 Grammatikens kompetens — sammanfattning

En av de mest intressanta egenskaperna vid bedömningen av en parser är givetvis dess täckningsgrad, d.v.s. hur stor del av språket (av en viss typ, stil eller inom en given domän) den potentiellt har förmåga att analysera. Vid utvecklingen av LPS-0 har ambitionen varit att endast utveckla grammatiska regler för analysen av svenskans satstyper. Vid utvärderingen av denna parser skall därför här bortses från de fall då parsern inte klarar av att analysera en sats p.g.a. den ofullständiga behandlingen av övriga frastyper eller p.g.a. brister i lexikonet. Den intressanta frågan är alltså: Hur långt skulle satsreglerna i LPS-0 räcka, om lexikonet var fullständigt och grammatiken utan undantag kunde analysera övriga frastyper?

LPS-0's språkbeskrivning täcker in följande satstyper: deklarativa huvudsatser, imperativa satser, ja-nej-frågor, två typer av bisatser och infinitivfraser. Satser av dessa typer analyseras i mer eller mindre fullständig omfattning av LPS-0. För att en deklarativ huvudsats skall klaras av måste den överensstämma med följande

satsschema (som är av diderichsenliknande typ):

FUND ^a	FIN ^b	SUBJ ^c	ADV1 ^d	INFIN ^e	PART ^f	ARG ^g	ADV2 ^h
<i>Kalle</i>	<i>sjöng</i>						<i>igår</i>
<i>Igår</i>	<i>hade</i>	<i>hon</i>	<i>inte</i>	<i>tagit</i>		<i>den</i>	
<i>Han</i>	<i>håller</i>				<i>av</i>	<i>henne</i>	

^a I fundamentet står ett led, som utgörs av ett argument till predikatet eller ett adverbial.

^b Det finita verbet

^c Subjektet eller det formella subjektet placeras här om det inte är identiskt med fundamentet

^d Den första adverbialsgruppen. Kan vara tom.

^e Eventuell infinit verbform

^f Eventuell verbpartikel

^g Predikatets argument, förutom subjekt och med möjligt undantag av ett som står i fundamentet.

^h Den andra (finala) adverbialsgruppen. Kan vara tom.

En majoritet av de huvudsatser som förekommer i normal sakprosa torde överensstämma med detta schema. Undantag utgör exempelvis vissa meningar med det besvärliga ordet *kanske*, som *Han kanske sjunger.*, liksom fall där ett argument till verbet, annat än subjektet, hamnar före verbet, som i *Lantbruksstyrelsen har av tullen begärt att få hjälp med namnen på djurägarna.* Imperativa satser kan analyseras av LPS-0 om de överensstämmer med detta schema:

IMP ^a	ADV1	PART ^b	ARG ^c	ADV2
<i>Spring</i>				
<i>Ta</i>	<i>inte</i>	<i>på^d</i>	<i>hatten</i>	<i>nu</i>
<i>Ge</i>			<i>boken till henne</i>	

^a Det imperativa verbet

^b Eventuell verbpartikel

^c Predikatets argument, utom subjekt

^d Representerar endast en av två möjliga läsningar.

Undantag från detta schema torde vara sällsyntare än undantag från LPS-0's huvudsatsschema. Även schemat för ja-nej-frågor torde vara mycket vältäckande och stämma överens med de allra flesta ja-nej-frågor:

FIN	SUBJ	ADV1	INFIN	PART	ARG	ADV2
<i>Har</i>	<i>du</i>		<i>sjungit</i>			

Ifråga om bisatser är grammatikens potentiella teckningsgrad betydligt sämre. Regler har implementerats för endast två typer. Den första av dessa är relativsatser,

som skall överensstämma med följande schema för att kunna analyseras av LPS-0:

REL ^a	SUBJ ^b	ADV1	FIN	INFIN	PART	ARG ^c	ADV2
<i>som</i>	<i>han</i>	<i>inte</i>	<i>har</i>	<i>bott</i>		<i>i</i>	
	<i>han</i>		<i>höll</i>		<i>av</i>		

^a Relativpronomenet, som är fakultativt om korrelatet inte är subjekt i relativsatsen.

^b Subjektet sätts inte ut om det är identiskt med korrelatet.

^c Predikatets argument, förutom subjekt och med möjligt undantag av ett som är identiskt med korrelatet.

Även om det kan förmodas att de flesta relativsatsen är formade enligt detta schema är undantag från det ganska vanliga. Exempelvis har ingenting gjorts i grammatiken för att hantera fall där en preposition styr relativpronomenet, som i *Sjukdomen i vilken hunden insjuknade var rabies..* Explikativa bisatser, den andra kategorin av bisatser som tagits med i LPS-0's grammatik, är inte lika problematiska och kan analyseras om de överensstämmer med detta schema:

INL ^a	SUBJ	ADV1	FIN	INFIN	PART	ARG	ADV2
<i>att</i>	<i>hon</i>		<i>hade</i>	<i>tagit</i>		<i>boken</i>	

^a Bisatsinledaren

Undantag från detta torde vara av samma slag som undantagen från LPS-0's huvudsatsschema. Även dess infinitivfrasschema torde vara tämligen vältäckande:

INFM ^a	ADV1 ^b	INF ^c	INFIN	PART	ARG	ADV2
<i>att</i>		<i>springa</i>				<i>idag</i>

^a Infinitivmärke. (Fakultativt)

^b Infinitivmärket måste vara utsatt för att adverbial skall få förekomma före infinitivverbet.

^c Verb i infinitivform

Grammatiken kan hantera partikelverb och sammansatta tempus på ett tillfredsställande sätt. Ingenting har dock gjorts för att ta hand om reflexiva verb. Både aktiv och passiv diates är möjlig. Ifråga om de typer av verbargument som kan hanteras är grammatiken ofullständig. Arbetet inom LPS-projektet med ett verbvalenslexikon ger vid handen att ett trettiotal typer kan behövas. I den föreliggande grammatiken har elva definierats: subjekt, formellt subjekt, egentligt subjekt, indirekt objekt, direkt objekt, prepositionsobjekt, lokationsobjekt, temporalobjekt, explikationsobjekt, infinitivobjekt, och agent. Endast en begränsad uppsättning verbvalensregler (verbaktionsregler) har definierats. Lexikonet kan dock betraktas som deras logiska hemvist. LPS-0's kompetens vad gäller adverbial är endast fragmentarisk.

Möjligheten att samordna satser har helt ignorerats i LPS-0's grammatik. De ovan nämnda fenomen som språkbeskrivningen inte klarar av kan relativt lätt inkorporeras i den. I samband med samordningar uppstår dock problem vars lösning inte ges någon antydning i LPS-0's grammatik. Jag tänker då på det fenomen som brukar kallas dubbelsyftning, och som består i att ett led som är utsatt i endast en

av två eller flera satser i en samordning i själva verket har en funktion i flera dessa, som i exempel som: *Han spelar och sjunger.* och *Kalle gav boken åt Lisa och hon pengarna åt honom..*

4 LPS-0 — Utvärdering av resultat

En parsers egenskaper bestäms i hög grad av det underliggande parsingmaskineriet. Många drag hos LPS-0 är sålunda en direkt konsekvens av UCP's konstruktion. Även om UCP är ett kraftfullt och ofta väl fungerande redskap för analys av naturligt språk, är den dock behäftad med ett antal svagheter, som huvudsakligen beror UCP-formalismens utpräglat procedurella karaktär. Dessa bör nämnas innan de specifika egenskaperna hos LPS-0 diskuteras.

Utmärkande för UCP är att hela kontrollen av vad som sker i charten under parsingprocessens förlopp ligger i grammatikskrivarens händer. Sålunda måste avancemang av aktiva bågar, ansättning av regler, lagring av resultat etc. begäras explicit i de regler som utgör språkbeskrivningen. Motiven till att på detta sätt överlämna alla parsingstrategiska beslut till grammatikskrivaren, har främst varit att att denne därmed skall ges möjlighet att formulera grammatiken på ett sätt som svarar mot ett från ekonomisk synpunkt optimalt beteende hos parsern. Det finns dock ett antal nackdelar med språkbeskrivningar som på detta sätt är procedurella. Detta har lett till att man inom datalingsvistik mer och mer övergivit procedurellt formulerade språkbeskrivningar till förmån för deklarativa grammatiker. Problemen med procedurella språkbeskrivningar kommer sig till största delen av att den innehåller två vitt skilda slag av information: dels fakta om språkliga uttrycks uppbyggnad, dels parsingtekniska instruktioner. Därmed får den som skriver språkbeskrivningen en svårare uppgift: Han har både att specificera lingvistiska fakta och parsingteknisk information. För att kunna göra detta måste han vara väl insatt i det bakomliggande parsingmaskineriets verkningssätt och hela tiden beakta dess beteende. För att kunna utnyttja de fördelar det procedurella skrivsättet ger måste han också, gång på gång, göra parsingstrategiska överväganden. Hans uppgift är därmed betydligt mer krävande än dens som skriver en parsergrammatik i en deklarativ formalism, där man kan bortse från allt parsingtekniskt.

En annan nackdel, av mer teknisk art, med procedurella språkbeskrivningar, är att de är specialdesignade för ett visst slag av parsingmaskineri, som inte tillåter några större modifikationer. En deklarativt formulerad parsinggrammatik kan därmed användas ihop med varje tänkbar tillräckligt kraftfull parsingteknik. Arbetar man med deklarativa grammatiker är man alltså fri att övergå till en bättre parsingmetod, om en sådan blir tillgänglig, vilket man med största sannolikhet inte kan göra om man arbetar med en procedurell språkbeskrivning (annat än med stora svårigheter).

Deklarativa språkbeskrivningar har också den fördelen framför procedurella att de är meningsfulla i sig själva — i någon mening — och förståeliga utan referens till ett visst slag av parsingmaskineri. En grammatik till en parser representerar ett vetenskapligt resultat och ett sådant bör vara presenterat på ett så rent och så allmänt tillgängligt sätt som möjligt och utan att vara låst till en viss tillämpning.

Procedurella språkbeskrivningar motsvarar i mindre grad dessa krav än deklarativa.

UCP har, förutom de ovan nämnda nackdelarna med en procedurell formalism, ett antal mer specifika egenskaper som försvårar grammatikskrivarens uppgift. Den kanske mest framträdande av dessa är den begränsning av möjligheterna att organisera språkbeskrivningen i ett antal modulära enheter som blir följden om man vill ha en så ekonomiskt effektiv språkbeskrivning som möjligt. Detta är en konsekvens av det sätt varpå nya aktiva bågar initieras i UCP: I LPS-0's språkbeskrivning ansätts exempelvis prepositionsfrasregeln sedan en preposition känts igen (d.v.s. parsingprocessen försiggår i detta fall bottom-up), och instruktionen om detta måste finnas i den regel som lagrar analysen av en preposition. Detta innebär att det inte varit möjligt att fördela arbetet så att en person tog hand om analysen av ord och en annan ansvarade för de syntaktiska komponenterna, utan att den senare var tvungen att lägga sig i vad morfologen gjorde. Detta försvårar givetvis det praktiska arbetet med UCP.

Även bristen på bra tracerutiner gör UCP svår att arbeta med. De tracerutiner som finns är alltför primitiva för att ge grammatikskrivaren något mer användbart stöd. Anledningen till detta är att man under trace får information om allt som händer i parsern, morfologi och syntax omvartannat. Man kan välja vilka slag av operationer som skall anges i trace-utskriften och man kan välja att hoppa över evalueringen av ett antal bearbetningssteg, men det finns ingen möjlighet att ange vad man är intresserad av i lingvistiska termer. Analysen av en normallång mening kan omfatta några tusen operationer. Det är således långt ifrån lätt att i en trace hitta det som för tillfället är relevant. Arbetet med UCP skulle underlättas avsevärt om den utrustades med tracerutiner där man exempelvis kunde begära att få se evalueringen av en viss regel, analysen av fraser av en viss kategori, eller alla försök att finna subjektet till en sats. De nämnda bristerna hos de nuvarande tracerutinerna är knappast en svaghet hos UCP som sådan, utan snarare hos den nuvarande UCP-implementationen.

Av det som ovan sagts bör man inte dra slutsatsen att UCP skulle vara ett särskilt dåligt parsingsystem, tvärtom bör det betonas att många syntaktiska konstruktioner kan analyseras på ett mycket övertygande sätt i UCP. Man bör också komma ihåg att många föregivet deklarativa lingvistiska formalismer i realiteten har många procedurella drag, varför ovanstående argument inte talar med full styrka till deras fördel. UCP's har den goda egenskapen att vara speciellt lämpad för implementering av grammatiska principer som är formulerade i enighet med traditionell (pre-chomskyansk) grammatikteori. Ett betydelsefullt exempel på detta är den naturlighet varmed diderichsenska satsscheman låter sig formuleras i UCP-formalismen. Ett system som är otvivelaktigt överlägset UCP i att utföra traditionell satslösning och generera den typ av analyser som LPS-0 arbetar med är förmodligen inte lätt att hitta. Icke desto mindre vill jag hävda att många av de ovan nämnda dragen hos UCP är otillfredsställande, och att vissa genomgripande förändringar behövs för att systemet ska kunna fungera väl.

För en parser är givetvis egenskaperna hos dess språkbeskrivning minst lika avgörande som de hos dess parsingmaskineri. Låt mig därför sammanfatta och diskutera de specifika dragen hos grammatiken till LPS-0. Det viktigaste av dessa är givetvis det slag av grammatisk analys som produceras. Denna anger i detta

fall den grammatiska funktionsstrukturen i de analyserade satserna (och övriga frastyper). En sådan analys är mer semantiskt orienterad än en rent frasstrukturell analys. Typiskt för LPS-0 är att frasstrukturen analyseras parallellt med att den funktionella strukturen byggs upp. Ett alternativ till detta arbetssätt är att dela upp analysprocessen i två steg, varav det första utför en analys av frasstrukturen och det andra ur det genererade frasstrukturträdet beräknar den grammatiska funktionsstrukturen. I det senare fallet möjliggörs en separering av frasstrukturella fakta från grammatiskt funktionell information i språkbeskrivningen. I LPS-0's språkbeskrivning förekommer dessa två slag av fakta sida vid sida i reglerna. Detta överensstämmer väl med valet av Diderichsens (1966) modell som utgångspunkt för analysen av olika satstyper. Denna har i LPS-0 kombinerats med en valensteoretiskt grundat tillvägagångssätt för analysen av de argument som knyts till verb. Härvid används s.k. verbaktionsregler, i vilka det anges vilka argument förutom subjekt eller formellt subjekt verbet konstrueras med. Vid formuleringen av dessa regler utnyttjas inga procedurella drag hos UCP-formalismen. De är alltså, i denna mening, deklarativa till sitt väsen och lätta att formulera. Detta är en viktig fördel, eftersom det kan förutses att grammatiken måste kompletteras med ett betydande antal verbaktionsregler när den byggs ut. I LPS-0 knyts endast en verbaktionsregel till varje lexem. Detta leder i de fall där ett verb kan förekomma i flera olika konstruktioner till antingen att verbaktionsregeln blir komplicerad (eftersom den innehåller många alternativ) eller att verbets valens blir ofullständigt beskriven. Detta problem kan enkelt lösas genom att fler verbaktionsregler tillåts för varje lexem. Argumenten till ett verb anges i verbaktionsreglerna explicit både för aktiv och passiv diates. Detta leder till en viss redundans i språkbeskrivningen, eftersom ett verbs konstruktion vid passiv diates ofta lätt kan härledas ur dess konstruktion vid aktiv diates. Detta är dock svårt att undvika emedan verbaktionsreglerna är procedurella såtillvida att de består av direkta anrop till argumentidentifieringsregler. Vid formuleringen av dessa har däremot en betydande generalisering kunnat göras genom det uniforma hanterandet av fundamentet i en deklarativ huvudsats och korrelatet till en deklarativ bisats.

Semantisk information utnyttjas i ringa omfattning i språkbeskrivningen. I verbens valensangivelser (d.v.s. i verbaktionsreglerna) finns sålunda inga selektionsrestriktioner angivna. Detta leder till att LPS-0 för vissa meningar genererar analyser som från semantisk synpunkt är absurda. Många av de ambiguiteter som parsern finner, är således att betrakta som artefakter. Verbaktionsreglerna är dock formulerade på ett sådant sätt att de utan svårigheter kan kompletteras med selektionsrestriktioner.

Den parser som här presenterats, LPS-0, lämnar inte tillräckligt underlag för en slutgiltig och mer precis bedömning av de principer varpå den vilar; Därtill är dess språkbeskrivning alltför begränsad till sin omfattning. Det kan dock förmodas att många av de tekniker som utnyttjats vid dess konstruktion även i framtiden kommer att spela en stor roll i parsingsammanhang. Det underliggande parsingmaskineriet, chartparsingsystemet UCP, representerar en effektiv och kraftfull algoritm för analys av naturligt språk. Bruket av valensorienterade språkbeskrivningar — i en eller annan form — torde vara en förutsättning för varje mer framgångsrikt parsingssystem. De kanske två största problemen med valens i parsingsammanhang, om

man bortser från bristen på valenslexikon, är dels att avgöra hur detaljerad, hur semantiskt orienterad valensbeskrivningen skall vara, dels att finna en lämplig form för representation av valensinformationen. LPS-0 är inte mer än ett trevande försök att använda valensinformation, och dessa problem har här endast givits primitiva lösningar. LPS-0 är således bara en första approximation till en på valensteoretiska principer grundad parser för svenska.

REFERENSER

Carlsson, Mats (1981). Uppsala Chart Parser version 2 (UCP-2) System Documentation. Centrum för datorlingvistik, Uppsala universitet.

Diderichsen, Paul (1966). *Elementær dansk Grammatik*. Köpenhamn: Gyldendal.

Schieber, Stuart M. (1986). *Introduction to Unification-Based Approaches to Grammar*. Stanford: CSLI.

Somers, H. L. (1987). *Valency and Case in Computational Linguistics*. Edinburgh: Edinburgh University Press.

Svensk Ordbok (1986). Stockholm: Esselte studium.

Sågvall Hein, Anna (1987). Parsing by means of Uppsala Chart Processor (UCP). i L. Bolc (ed), *Natural Language Parsing Systems*. Berlin & Heidelberg: Springer Verlag.

Sågvall Hein, Anna (1987a). Forskningsprogram för projektet EN LEXIKONORIENTERAD PARSER FÖR SVENSKA. Språkdata, Göteborgs Universitet (dupl.)

Sågvall Hein, Anna (1988). The LPS Inflectional Grammar. A listing of the rules. Språkdata, Göteborgs Universitet (dupl.)

Sågvall Hein, Anna & Ahrenberg, Lars (1985). A Parser for Swedish. Status Report for SVE.UCP June 1985. Centrum för datorlingvistik, Uppsala universitet.

Thorell, Olof (1973). *Svensk grammatik*. Stockholm: Esselte studium.

Toporowska-Gronostaj, Maria (1987). Om verben i LPS-projektets lexikon. Språkdata, Göteborgs universitet.