



UPPSALA
UNIVERSITET

En fonotaktisk modell för svensk fonemigenkänning

Sebastian Berlin

Institutionen för lingvistik och filologi
Språkteknologiprogrammet
Examensarbete i datorlingvistik

8 november 2010

Handledare:
Alexander Seward, Veridict AB
Mats Dahllöf, Uppsala Universitet

Sammandrag

I denna uppsats undersöks om fonotaktiska regler kan användas för att förbättra taligenkänning. En fonotaktisk modell för svenska skapas utifrån dokumenterade regler och inkorporeras i ett redan befintligt system. Resultat för fonemigenkänning med och utan den fonotaktiska modellen jämförs och redovisas. Resultatet är en blygsam förbättring, 1,33%, men visar ändå på att en fonotaktisk modell kan ge ett förbättrat resultat.

En utvärdering av själva modellen mot en guldstandard visar på en hög täckning, 97,65%. Samtidigt ger modellen en ganska låg täckning, 11,90%, mot helt slumpmässiga fonsekvenser, vilket talar för att modellen inte är för tillåtande.

Ett antal förslag för vidare utveckling och förbättring av modellen ges också.

Abstract

This thesis investigates whether phonotactical rules can be used to improve speech recognition. A phonotactical model for Swedish is created from documented rules and is incorporated in an already existing system. Results for phoneme recognition with and without the phonotactical model are compared and presented. The result is a modest improvement, 1.33%, but still shows that a phonotactical model can give an improved result.

An evaluation of the model in itself against a gold standard shows a high recall, 97.65%. At the same time the model gives a quite low recall, 11.90%, against completely random phone sequences, which suggests that the model isn't too permissive.

A number of ways to further develop and improve the model are also proposed.

Innehåll

Sammandrag	2
Abstract	2
Innehåll	3
Förord	5
1 Inledning	6
1.1 Syfte	6
1.2 Disposition	6
2 Bakgrund	7
2.1 Svensk fonologi	7
2.1.1 ASTA	8
2.2 Svensk fonotax	8
2.2.1 Genererande regler	9
2.2.2 Begränsande regler	10
2.2.3 Modifierande regler	10
2.3 Finita transduktorer	11
2.3.1 Specialtecken	11
2.3.2 Operationer	12
2.4 Taligenkänning	13
2.5 <i>Speech Recognition Grammar Specification</i>	14
3 Utförande	17
3.1 Resurser	17
3.1.1 SpeechCAD	17
3.1.2 Data	18
3.2 Implementation av regler	18
3.2.1 Genererande regler	18
3.2.2 Begränsande regler	18
3.2.3 Modifierande regler	19
3.2.4 Kombination av regler	24
4 Utvärdering	25
4.1 Fonemigenkänning	26
5 Diskussion	27

Litteraturförteckning	28
A Enheter	29
A.1 I	29
A.2 F_1	30
A.3 F_2	30
A.4 F_3	30
A.5 F_4	31
A.6 M_a	31
A.7 M_{p1}	32
A.8 M_{p2}	33
A.9 M_{p3}	33
A.10 V_0	33
A.11 V_a	33
A.12 V_{p1}	33
A.13 V_{p2}	33
A.14 V_{p3}	33
B Begränsande regler	34
B.1 $I + V$	35
B.2 $V_0 + C$	36
B.3 $V_0 + V_{p1}$	36
B.4 $V_{p1} + C$	36
B.5 $V_{p1} + V_{p2}$	37

Förord

Jag vill tacka alla på Veridict för att ha gett mig idén till, och möjligheten att utföra, denna uppsats. Särskilt vill jag tacka Alexander Seward, som handlett mig, med allt vad det innebär i form av tips, uppsatsgranskning och ifrågasättning av mina alltför knasiga infall.

Jag vill också tacka Mats Dahllöf för handledning under uppsatsskrivningen och feedback.

Sist, men inte minst, vill jag tacka Johannes Berlin, för att vara en bra bror i allmänhet och för att tillhandahålla material till data i synnerhet.

1 Inledning

1.1 Syfte

Syftet med den här uppsatsen är att ta reda på om det går att utnyttja fonotaktiska regler för att förbättra svensk taligenkänning. Hypotesen är att dessa begränsningar medför ett bättre resultat för fonemigenkänning, jämfört med system som inte tar hänsyn till dessa regler. Detta eftersom det inte finns några begränsningar i det vanliga förfarandet vid fonemigenkänning, vad det gäller fonotax. Resultatet kan således bli något som en svensk talare inte brukar, eller i vissa fall knappt kan, uttala.

För att kunna undersöka resultatet av en sådan utveckling av taligenkänning skapades en modell utifrån redan befintliga iakttagelser av svensk fonotax. Denna inkorporerades sedan som en del i ett redan existerande system för taligenkänning.

1.2 Disposition

Jag inleder uppsatsen med att beskriva fonotaktiska regler för svenska i kapitel 2. Här definieras en uppsättning foner, som tillsammans med ASTA-alfabetet används i uppsatsen. Tre typer av fonotaktiska regler, som till stor del bygger på Sigurd (1965), definieras i detta kapitel. Grunderna för finita transduktorer, taligenkänning och SRGS förklaras även kortfattat.

I kapitel 3 beskrivs utförandet av arbetet. Detta kapitel inleds med en genomgång av systemet som arbetet utfördes i och data som användes. Sedan följer beskrivningar av hur reglerna definierade i föregående kapitel realiserar på ett sätt som kan inkorporeras i systemet. Här beskrivs även hur dessa regler kombineras.

Kapitel 4 beskriver några sätt att utvärdera den fonotaktiska modellen. Resultatet av dessa utvärderingar redovisas samt lite tankar om vad detta resultat säger.

Uppsatsen avslutas i kapitel 5 med en diskussion om hur man skulle kunna utveckla modellen ytterligare och vad detta skulle kunna leda till.

I bilagorna finns alla enheter ordnade efter vilken position de står på och möjliga kombinationer mellan olika enheter.

2 Bakgrund

2.1 Svensk fonologi

I denna uppsats används fonerna i tabell 2.1 för att representera svenska vokaler. Som synes innefattar dessa allofonerna [æ:], [æ], [ø:] och [ø], vilka används vid vokalsänkning, se avsnitt 2.2.3 samt de två diftongerna [au] och [eu]. Det görs ingen skillnad på den korta motsvarigheten till [ɛ:] och [ɛ], vilket man inte gör i "centralt mellansvensk standarduttal" (Engstrand, 2004, s. 113).

Denna representation av svenska vokaler följer inte strikt IPA-specifikationer i det att korta vokaler har samma symbol som långa, med den skillnad att de långa vokalerna följs av :. Denna vokaluppsättning är samma som används i Sigurd (1965), utökade med sänkta vokaler från Engstrand (2004). Den ger också en tydligare översikt över fonempar för korta och långa vokaler. Eftersom det är av föga intresse för denna uppsats att i detalj definiera svenska vokaler fann jag det tillräckligt att använda mig av denna något förenklade uppsättning.

Tabell 2.1: Vokaler i svenskan och hur de representeras i den här uppsatsen.

Fon	ASTA	Exempel	Fon	ASTA	Exempel
[i:]	i:	pingv <u>i</u> n	[i]	i	pingv <u>i</u> n
[e:]	e:	ren	[ɛ]	e	h <u>ä</u> st
[ɛ:]	ä:	r <u>ä</u> v			
[y:]	y:	myrslo <u>k</u>	[y]	y	nyckelp <u>i</u> ga
[u:]	u:	lem <u>u</u> r	[u]	u	klumpf <u>i</u> sk
[ø:]	ö:	ö <u>d</u> la	[ø]	ö	mjölkh <u>a</u> j
[ɑ:]	a:	ap <u>a</u>	[ɑ]	a	ap <u>a</u>
[o:]	å:	å <u>l</u>	[o]	å	r <u>å</u> tt
[u]	o:	noshör <u>n</u> ing	[u]	o	o <u>r</u> m
[æ:]	ä2:	tär <u>n</u> a	[æ]	ä2	ter <u>m</u> it
[ø:]	ö2:	noshör <u>n</u> ing	[ø]	ö2	mö <u>r</u> t
[au]	AU	aurorahuss <u>n</u> ok	[eu]	EU	europ <u>e</u> isk maskorm

De konsonanter som använts står att finna i tabell 2.2. Även här har jag utgått från Sigurd (1965). De flesta konsonanter är ganska oproblematiske, men det finns några som värda att förklara ytterligare. [r] innefattar alla varianter av svenska r-ljud, vilka kan variera mycket mellan olika dialekter. Ett annat uttal som varierar i svenskan är sje-ljudet. För dessa ljud har jag använt mig av IPA-symbolen [ʃ] och gör ingen skillnad på de olika uttalen. Ytterligare en grupp av konsonanter som är värda att nämna är retroflexerna: [ɭ], [ɻ], [ɗ], [ɬ]

och [ʂ]. Dessa uppstår vid assimilation av [r] och [l], [n], [d], [t] respektive [s] (Engstrand, 2004, s. 170). Regler för retroflexivering behandlas i avsnitt 2.2.3.

Tabell 2.2: Konsonanter i svenskan och hur de representeras i den här uppsatsen.

Fon	ASTA	Exempel	Fon	ASTA	Exempel
[r]	r	rå <u>tt</u> a	[k]	k	kro <u>ko</u> dil
[l]	l	l <u>e</u> mur	[t]	t	ta <u>x</u>
[m]	m	myr <u>s</u> lok	[s]	s	s <u>äl</u>
[ŋ]	ng	pi <u>ng</u> vin	[ç]	tj	tj <u>ä</u> der
[n]	n	n <u>ä</u> bbmus	[h]	h	h <u>aj</u>
[v]	v	r <u>ä</u> v	[ʃ]	s	sch <u>i</u> mpan
[j]	j	g <u>ä</u> dda	[ʀ]	r	rostp <u>ä</u> rluggla
[b]	b	bl <u>ä</u> sfisk	[r̥]	rn	ö <u>r</u> n
[g]	g	gro <u>d</u> a	[d̥]	rd	m <u>å</u> rd
[d]	d	di <u>ng</u> o	[t̥]	rt	h <u>jo</u> rt
[f]	f	toff <u>l</u> djur	[ʂ]	rs	hars <u>kr</u> ank
[p]	p	ap <u>a</u>			

2.1.1 ASTA

ASTA (*Augmented Swedish Transcription Alphabet*) är ett alfabet under utveckling som tagits fram på KTH. ASTA är anpassat för transkription av svenskt tal, och innehåller, utöver symboler för svenska fonem, xenofoner, d.v.s. främmande foner (Eklund och Lindström, 1998), och extralingvistiska ljud.

De ASTA-symboler som används i denna uppsats finns i tabell 2.1 och 2.2. Dessa används vid implementationen av modellen eftersom de används i systemet för att representera språkljud.

2.2 Svensk fonotax

Sigurd (1965) låg till grund vid konstruktionen av den fonotaktiska modellen. Sigurd definierar regler för konstruktion av **enkla ord** (eng: *simple words*). enkla ord Motsatsen till dessa är sammansättningar (eng: *compounds*), vilka är kombinationer av två rotmorfem, ett prefix och ett rotmorfem eller ett rotmorfem och ett suffix.

Definitionen av ord är aldrig särskilt trivial och i svenskan brukar även sammansatta ord räknas som ord. Eftersom denna uppsats behandlar språket på en fonotaktisk nivå används begreppet "ord" för enkla ord. Ett sammansatt ord som *jätteapa* behandlas således som de två orden *jätte* och *apa*.

Enkla ord beskrivs fonotaktiskt som ett av följande:

1. Ord som består av endast en betonad stavelse, ex: *å*, *ål*, *små*.
2. Ord som består av en betonad stavelse, följd av en eller flera obetonade stavelser, ex: *apa*, *apor*, *adliga*.
3. Ord som består av en betonad stavelse, föregången av en eller flera obetonade stavelser, ex: *hangar*, *apelsin*.

4. Ord som består av en betonad stavelse, följd av en eller flera obetonade stavelser och föregången av en eller flera betonade stavelser, ex: *hangarerna, apelsinens*.

Ursprungligen svenska ord (eng: *genuine Swedish words*) finns bara i grupp 1 och 2, medan låneord frekvent förekommer i grupp 3 och 4.

Sigurd skriver ofta om marginella fall (eng: *marginal cases*) och sekundära fall (eng: *secondary cases*). Marginella fall är konsonantkluster, som förekommer i ett fåtal låneord eller arkaiska ord. Sekundära fall är konsonantkluster, som bildas genom böjningar av ord. Jag har valt att godkänna alla fall av båda dessa typer.

2.2.1 Genererande regler

Ett ord består av ett antal **enheter**, vilka kan vara antingen konsonantkluster eller vokaler. Dessa är tagna från Sigurd (1965). Enheterna ser olika ut beroende på deras position i förhållande till den betonade vokalen, och kan i vissa fall vara "tomma", d.v.s. inte bestå av någon fon. Alla enheter finns i bilaga A. Hur dessa enheter benämns visas i figur 2.1.

$$I + (V_a + M_a)^* + V_0 + \left\{ \begin{array}{l} F_1 \\ M_{p1} + V_{p1} + \left\{ \begin{array}{l} F_2 \\ M_{p2} + V_{p2} + \left\{ \begin{array}{l} F_3 \\ M_{p3} + V_{p3} + F_4 \end{array} \right\} \end{array} \right\} \end{array} \right\} ?$$

Figur 2.1: Kombinationen av enheter för att bilda enkla ord. Viss notation är här lånad från reguljära uttryck: ? markerar att en del av uttrycket inte är obligatorisk och * markerat att en del av uttrycket kan förekomma noll eller flera gånger.

V står för vokal, alla andra enheter är någon form av konsonantkluster. V₀ Den betonade vokalen benämns V₀, där 0 står för position noll, eftersom alla andra enheter får sina benämningar beroende på förhållandet till denna. I står I för initialt konsonantkluster och är det kluster som inleder ett ord. F står för F för finalt konsonantkluster och är det kluster som förekommer som sista enheten i ett ord. M är mediala konsonantkluster, d.v.s. kluster som förekommer på en annan plats i ett ord än initialt eller finalt. I V_a och M_a står a för "ante" (lat: *före*) och innebär att enheten förekommer före V₀. På liknande sätt står p för "post" (lat: *efter*), och visar att enheten förekommer efter V₀. Siffran i F, M_p och V_p står för vilken position efter V₀ de förekommer på.

Antalet möjligheter begränsas mer och mer ju längre från V₀ man kommer; i själva V₀ kan alla vokaler förekomma. Det är också bara i V₀ som långa vokaler kan förekomma, även om det finns viss ovisshet vad det gäller vokallängd, se Sigurd (1965, s. 25).

Vad det gäller konsonantkluster tas ingen hänsyn till positionen hos M_a. Detta eftersom ord med enheter före V₀ oftast är låneord, varför dessa enheter inte nödvändigtvis följer svensk fonotax.

2.2.2 Begränsande regler

Begränsande regler ser till två enheter och hur dessa kan kombineras. Även dessa regler kommer från Sigurd (1965) och alla regler som används i uppsatsen finns i bilaga B. I denna uppsats tillämpas endast regler som begränsar två på varandra omedelbart följande enheter, om man inte räknar ”tomma” enheter; det kan finnas begränsningar mellan två vokaler om det inte förekommer någon konsonant mellan dessa. De enhetskombinationer som begränsas är: $I+V$, V_0+C , V_0+V_{p1} , $V_{p1}+C$ och $V_{p1}+V_{p2}$.

För kombinationen $I+V$, där V kan vara både V_0 och V_a , är regeln ganska tillåtande; bara 17 av de totalt 55 möjliga konsonantklustren i position I har överhuvudtaget begränsningar på den följande vokalen. De flesta av dessa kluster, 11 stycken, slutar på antingen $[j]$ eller $[v]$. Dessa 11 kluster har även flest otillåtna kombinationer; av totalt 69 finns 53 hos något av dem. Reglerna för $I+V$ är alltså ganska homogena; 2 av konsonanterna står för en majoritet av begränsningarna. Ju längre klustrena är desto större är chansen för begränsningar. Ett kluster begränsar aldrig mer än samma kluster med en extra konsonant, t.ex. har $[skv]$ samma begränsningar som $[kv]$ plus några till. Det finns bara ett enkonsonantkluster som har några begränsningar, nämligen $[g]$.

För V_0+C finns det två regler: vokalen är alltid kort före $[ŋ]$ och $[j]$ samt vokalen är alltid lång före tom kluster, d.v.s. i slutet av ett ord eller då den följs av en annan vokal.

Utan att gå i detalj på de resterande kombinationerna kan nämnas att dessa är ganska få. Ju senare i ordet de kommer desto oftare beror de på böjningssuffix.

2.2.3 Modifierande regler

De två regeltyperna ovan begränsar hur enheter kan förekomma, beroende på position och kontext. Modifierande regler, å andra sidan, ändrar på foner. En annan sak som skiljer dessa regler från de övriga två är att de inte tar hänsyn till morfem- eller ordgränser.

Retroflexivering¹ (Linell m.fl., 1971) kan infalla då $[r]$, eller andra retroflexer, följs av $[t]$, $[d]$, $[n]$, $[s]$ eller $[l]$. Då försvinner $[r]$ och den efterföljande konsonanten omvandlas till $[t]$, $[d]$, $[ŋ]$, $[s]$ respektive $[l]$, d.v.s. motsvarande retroflex. Retroflexivering kan ske upprepade gånger efter varandra om det förekommer flera konsonanter i rad som kan retroflexiveras. Även om retroflexivering ofta uppstår i vardagligt tal förekommer det inte i vissa dialekter, vid superartikulering eller i vissa fall då det ursprungliga $[r]$ är långt; *barrträ* uttalas $[bar:trɛ:d]$, inte $[barʀrɛ:d]$ eller $[baʀrɛ:d]$.

Retroflexivering

Vokalsänkning (Engstrand, 2004) uppstår i svenskan då $[ɛ:]$, $[ɛ]$, $[ø:]$ eller $[ø]$ följs av $[r]$ eller annan retroflex. Vokalerna sänks då och realiseras då som $[æ:]$, $[æ]$, $[ɤ:]$ respektive $[ɤ]$. Implementationen av denna regel begränsar även förekomsten av vanliga och sänkta vokaler, då dessa står i komplementär distribution.

Vokalsänkning

¹Jag väljer att använda termen ”retroflexivering” för att ge upphov till så lite förvirring som möjligt. Samma fenomen benämns även som ”supradentalisering” i Engstrand (2004) och Linell m.fl. (1971), men detta är också ett steg i denna process.

2.3 Finita transduktorer

Finita transduktorer (eng: *Finite-State Transducers*) (**FST**) är en typ av finita FST maskiner (eng: *Finite-State Machines*) (**FSM**)², som mappar mellan två reguljära språk (Jurafsky och Martin, 2009). En FST kan definieras som en 7-tupel:

$$\begin{aligned} T &= \{Q, \Sigma, \Delta, q_0, F, \delta, \sigma\} \\ q_0 &\in Q \\ F &\subseteq Q \\ \delta &(q, w) \\ \sigma &(q, w) \end{aligned}$$

I denna definition är Q en finit mängd tillstånd. Σ och Δ är alfabet för input respektive output. q_0 är starttillståndet och F är en mängd finala tillstånd. δ är en funktion för transitioner mellan tillstånd, vilken returnerar en mängd tillstånd, $Q' \subseteq Q$, givet tillståndet $q \in Q$ och strängen $w \in \Sigma^*$. Till skillnad från FSA:er kan det alltså finnas flera transitioner från ett tillstånd för samma input, eftersom dessa kan ge olika output. Funktionen σ ger en mängd av strängar, $o \in \Delta^*$, givet tillståndet $q \in Q$ och $w \in \Sigma^*$, d.v.s. output. Transitioner kan också tilldelas vikter, vilket ger en viktad finit transduktor (eng: *Weighted Finite-State Transducer*) (**WFST**).

I denna uppsats används notationen *input:output* för transitioner i FST:z

2.3.1 Specialtecken

Det finns ett antal specialtecken som kan användas i FSM:er. Dessa kan sättas på bågar som input, och i fallet med FST:er output, likt vanliga symboler. Det speciella med dem är att de inte tolkas ordagrant, som symboler i alfabetet.

Symbolen ϵ (epsilon) tolkas som ingenting. Detta gör att man kan skapa ϵ bågar som inte kräver input eller inte ger output. Med operationen *epsilon removal* kan man modifiera en FSM så att man får en motsvarande FSM utan ϵ . ϵ användes vid implementationen av retroflexivering, se 3.2.3. I denna uppsats skrivs epsilon antingen ϵ eller *eps*.

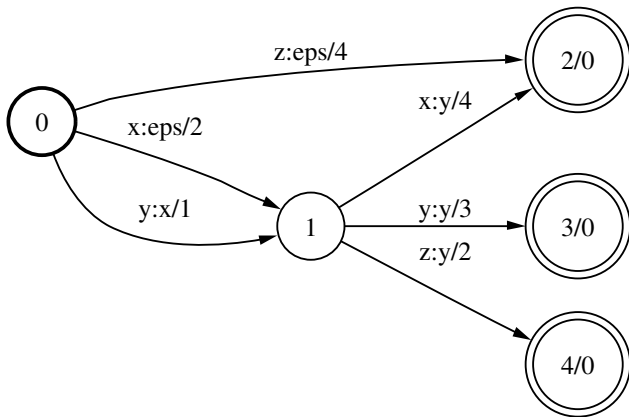
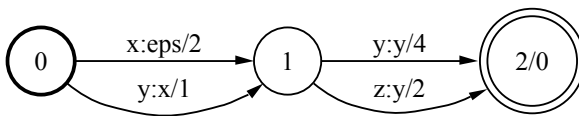
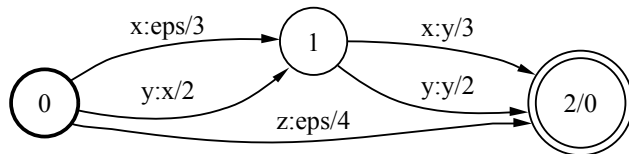
Symbolen ρ (rho) motsvarar alla symboler i alfabetet som inte finns på ρ explicita utbågar från tillståndet och är således ett förenklat sätt att skriva, som inte kräver att man definierar alla dessa bågar. Har man t.ex. en FSM med $\Sigma = \{a, b, c\}$ och en transition $q_1 = (q_0, a)$ skulle $q_2 = (q_0, \rho)$ motsvara $q_2 = (q_0, b)$ och $q_2 = (q_0, c)$. ρ var mycket användbar vid implementationen av både begränsande och modifierande regler då dessa behandlar en eller ett fåtal enheter. I denna uppsats skrivs rho antingen ρ eller *rho*.

²Definitionen går isär något när det gäller förhållanden mellan FSM:er, FSA:er och FST:er. Jag har valt att innefatta både FSA och FST i begreppet FSM, på samma sätt som bl.a. Eisner (2002). Ibland definieras istället FST som en typ av FSA, bl.a. i Jurafsky och Martin (2009). Jag har valt den förstnämnda definitionen eftersom den stämmer överens med terminologin i systemet modellen implementeras i.

2.3.2 Operationer

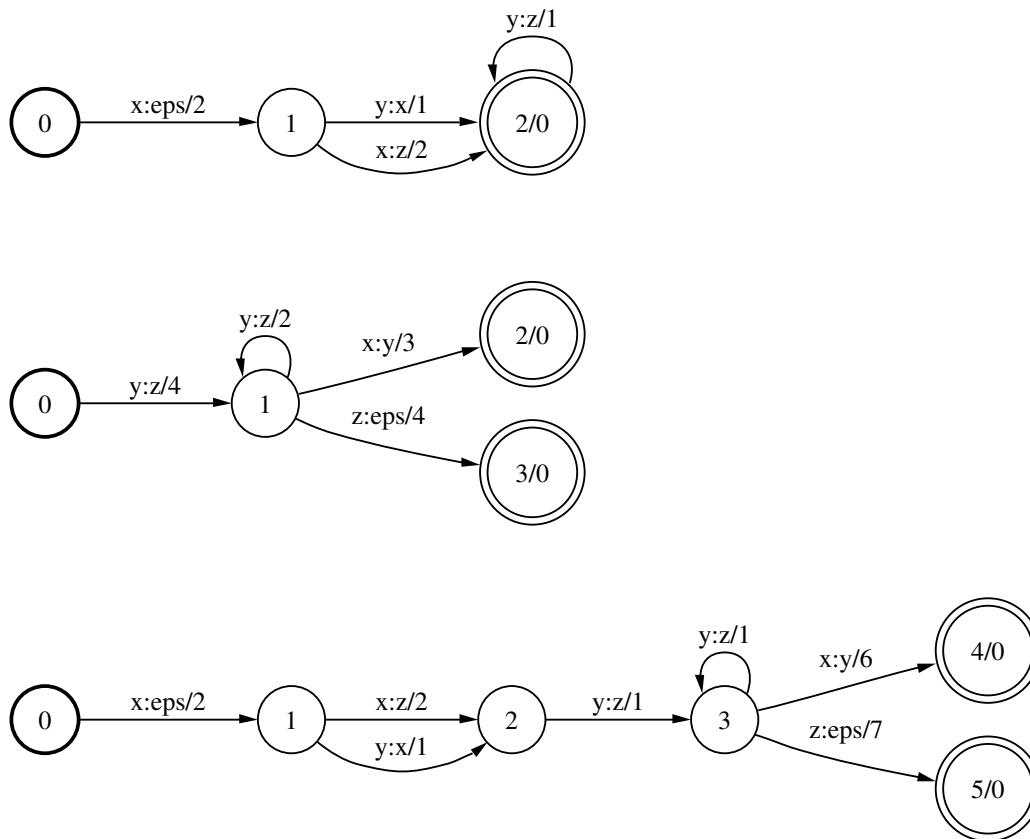
Det finns åtskilliga operationer som kan användas i samband med FSM:er. Jag kommer att ta upp några av dessa, som användes vid skapandet av modellen. En mer utförlig lista av operationer finns i Seward (2003).

Figur 2.2 visar **unionen** mellan två FST:er, $A \cup B = C$. En union mellan två FSM:er ger en automat som accepterar alla strängar som båda dessa automater accepterar. Union används av genererande regler, se avsnitt 3.2.1, då det kan förekomma F eller M_p .



Figur 2.2: Exempel av operationen union. Den nedre FST:n är en union av de två övre. Bilderna är lånade från Seward (2003). Siffran efter snedstreck visar vilken vikt en transition har.

Konkatenering kedjar ihop två FSM:er. Detta resulterar i en FSM som är en sekvens av de båda FSM:erna, där den andras starttillstånd sammanfogas med de finala tillstånden i den första. Figur 2.3 visar konkateneringen $A + B = C$. Med konkatenering går det att kedja flera enheter efter varandra för att skapa ord, se avsnitt 3.2.1.



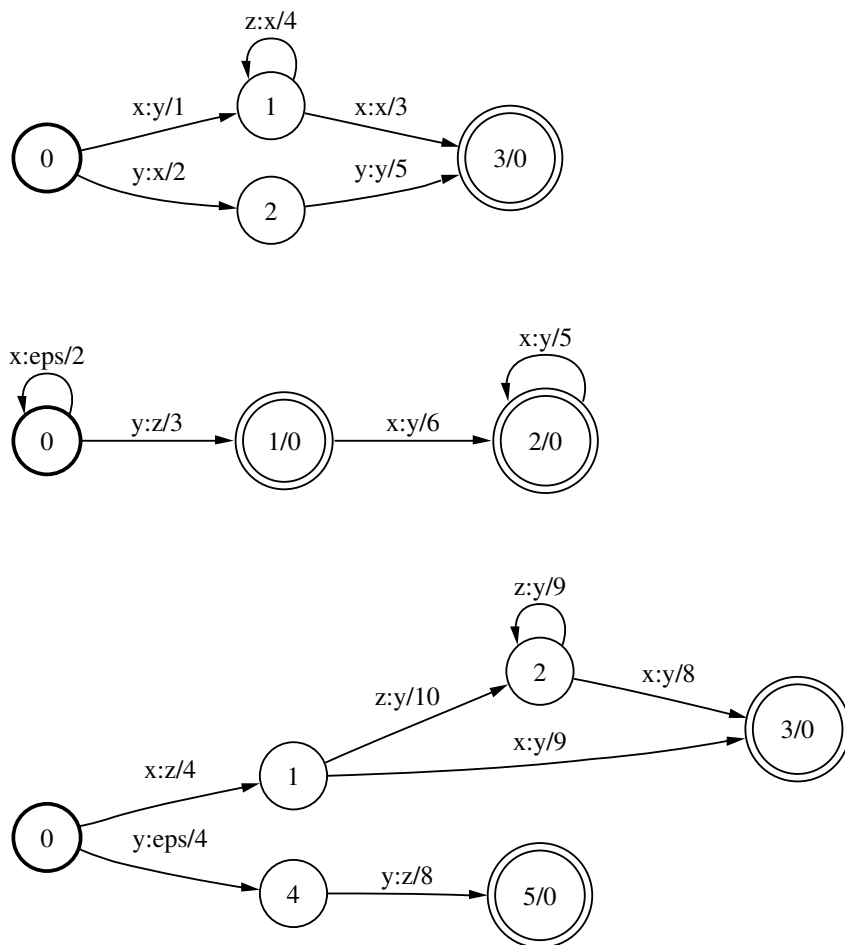
Figur 2.3: Exempel av operationen konkatenering. Den nedre FST:n är en konkatenering av de två övre. Bilderna är lånade från Seward (2003). Siffran efter snedstreck visar vilken vikt en transition har.

Komposition tar output från en FST och använder det som input i en annan. Detta innebär att FST:n $C = A \circ B$ i figur 2.4 tar input från A och ger output från B. Detta kan bl.a. användas för att skapa kaskader av FST:er, se avsnitt 2.4. I denna uppsats användes komposition även vid implementation av de modifierande reglerna, se avsnitt 2.2.3.

2.4 Taligenkänning

Taligenkänning kan ske i en kaskad (Seward, 2003), figur 2.5, från förprocessering av ljudet till grammatik, och möjligt ännu "högre" steg. Varje steg i denna kaskad består av ett godtyckligt antal FST:er. Stegen komponeras sedan till en enda FST, vilken tar samma input som den första FST:n och ger samma output som den sista.

Det första steg, F , tar hand om akustisk förprocessering. Detta steg omvandlar akustiska signaler till särdragsvektorer (eng: *feature vectors*). I steg två, S , mappas särdragsvektorer till distributionsettiketter genom PDF:er (*probability density functions*). H är en transduktor, som omvandlar en sekvens av PDF-etiketter till en sekvens av HMM-etiketter. I D mappas HMM-etiketterna



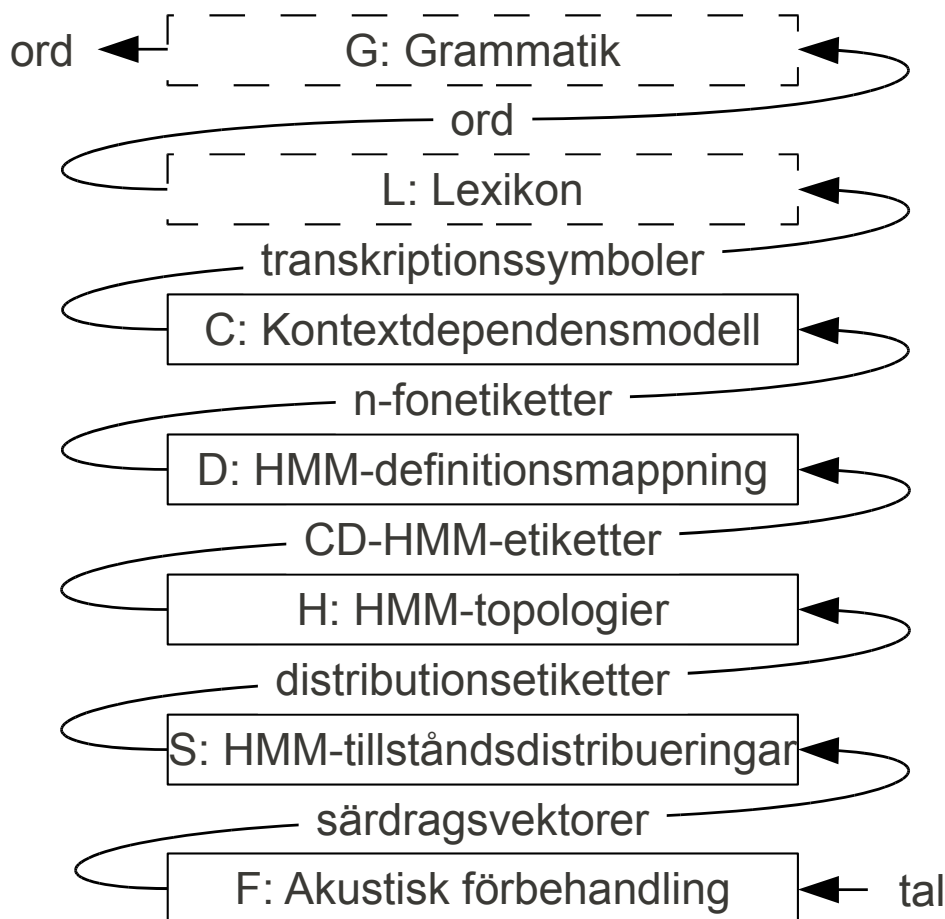
Figur 2.4: Exempel av operationen komposition. Den nedre FST:n är en komposition av de två övre. Bilderna är lånade från Seward (2003). Siffran efter snedstreck visar vilken vikt en transition har.

om till n-fonetiketter. Steg C introducerar fonetisk kontext, d.v.s. hur en fon påverkas av närliggande foner. L är ett lexikon som mappar en sekvens av transkriptionssymboler till ord. Grammatiken, G , bestämmer hur dessa ord kan kombineras till längre enheter.

I figur 2.5 finns stegen G och L med, vilka inte har använts i denna uppsats eftersom modellen bara använts vid fonemigenkänning. Istället har den fonotaktiska modellen lagts till sist i kaskaden, efter C .

2.5 Speech Recognition Grammar Specification

Speech Recognition Grammar Specification (SRGS) är en standard för representation av taligenkänningsgrammatik, framtagen av W3C (Hunt och McGlashan, 2004). SRGS går att skrivas med två olika syntaxer: *Augmented BNF syntax* (ABNF) och XML. SRGS struktur gör det möjligt att översätta SRGS-grammatiker till FSM:er eller reguljära uttryck. I den här uppsatsen har bara



Figur 2.5: Ett exempel på hur en taligenkänningskaskad kan se ut. De två översta stegen används inte vid fonemkänning.

XML använts, varför ABNF-formatet inte tas upp i detalj. Exempel ges av samma anledning i XML-kod.

I figur 2.6 visas en bit SRGS-kod med dess beståndsdelar uppmärkta.

1. Regler är icketerminaler. Dessa kan innehålla en mängd *items* och tokens.
2. Ett *item* innehåller ett token eller en regelreferens.
3. Ett token är en terminal, representerad som en textsträng.
4. En regelreferens refererar till en regel genom dess id och ger möjligheten att gå in i denna regel.
5. Med attributet *repeat* går det att definiera hur många gånger *item* ska förekomma. Detta ger möjligheten att återskapa effekten av +, * och ? i för reguljära uttryck.
6. Elementet *one-of* ger möjligheten att välja ett *item* från en lista *items*. Detta kan återskapa effekten av reguljära uttryck av typen (a|b|...).

```

<rule id="Mp2" scope="private"> (1)
  <item> (2)
    <one-of> (6)
      <item>
        <ruleref uri="#Mp2_r" /> (4)
      </item>
    [...]
      <item>
        d (3)
      </item>
    [...]
  </one-of>
</item>
%Mp2
</rule>
<rule id="Mp3" scope="private">
  <one-of>
  [...]
    <item>
      r
      <item repeat="0-1"> (5)
        n
      </item>
    </item>
    <item>
      s t
    </item>
  </one-of>
%Mp3
</rule>

```

Figur 2.6: Exempel på hur en SRGS-grammatik kan se ut i XML. I figuren visas hur de genererande reglerna för M_{p2} och M_{p3} ser ut i SRGS-XML.

3 Utförande

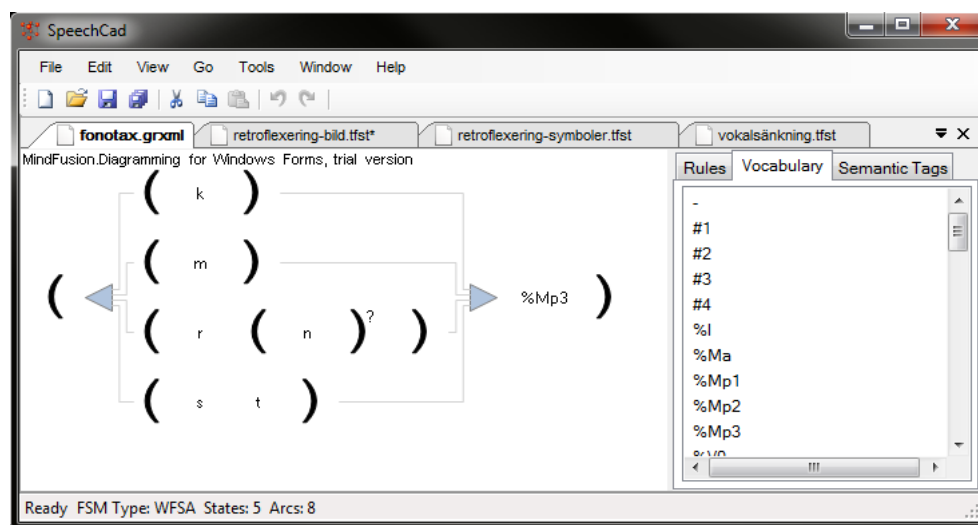
3.1 Resurser

3.1.1 SpeechCAD

SpeechCAD är en arbetsmiljö utvecklad av Veridict, som är anpassad för att utveckla system för taligenkänning.

Detta system innefattar V#, ett funktionellt högnivåspråk, som är anpassat för att enkelt kunna skapa just sådana system. I V# finns en mängd funktioner för att bygga och utvärdera taligenkänningsmodeller. I den här uppsatsen användes en rad funktioner för att modifiera FST:er enligt operationerna i avsnitt 2.3.2.

För att skapa de genererande reglerna användes Grammar Builder, en komponent av SpeechCAD, med vilken man kan bygga SRGS-grammatiker. Grammar Builder har ett grafiskt gränssnitt i vilket man kan komponera noder med hjälp av *drag-and-drop*, se figur 3.1.



Figur 3.1: En skärmbild från Grammar Builder. Denna bild visar hur den genererande regeln M_{p3} , en av de regler som visas i figur 2.6, visualiseras i det grafiska gränssnittet.

3.1.2 Data

För projektet skapades en mindre mängd data i form av inspelade meningar. Dessa märktes fonetiskt upp i programmet WaveSurfer¹ för att kunna användas i SpeechCAD.

För att göra det mer intressant valde jag att basera meningarna på en domän, sjukhusjournaler, som skulle kunna vara aktuell för taligenkänning. Jag letade därför rätt på ett antal fall, som används inom läkarutbildningen, för att använda som grund datan. Dessa är utformade som riktiga sjukhusjournaler.

Utöver meningarna som fanns i fallen skrevs ytterligare några meningar av vilka vissa innehöll fonsekvenser som strider mot modellen. Detta för att kunna undersöka om modellen var för tillåtande.

Datan består av 54 meningar med sammanlagt 1847 foner. Antalet unika foner är 44, antal difoner 429 och antal trifoner 867.

3.2 Implementation av regler

För att kunna använda mig av de fonotaktiska reglerna beskrivna i 2.2 skapade jag FSM:er, som motsvarar dessa. Dessa är antingen FSA:er eller FST:er, beroende på typen av regel.

3.2.1 Genererande regler

De genererande reglerna implementerades som en FSA, vilken skapades via SRGS. Enheterna sattes ihop på rad med konkatenering och där finalt eller medialt kluster förekommer kombinerades dessa med union. I fallet V_a och M_a , där det kan förekomma en, i teorin, obegränsat lång kedja av dessa användes kleene-stjärna.

I samband med konkatenering av enheter introducerades en uppmärkning av enhetsgränser. Detta för att kunna implementera regler som bara går att applicera på vissa platser i ett ord och för att lättare kunna se hur ord byggs upp av systemet. Denna uppmärkning visas och förklaras i tabell 3.1. Även symbolen $-$ introducerades som representation av en tom enhet, eftersom vissa enheter som kan vara tomma. All uppmärkning av enhetsgränser kommer efter den enhet den avser. Uppmärkningen tas bort i ett senare steg för att ge korrekta utsträngar. Med denna uppmärkning ser ordet *pingvinerna* (p i ng v i : n e r n a)² ut på följande vis:

p %I i % V_a ng v % M_a i : % V_0 n % $Mp1$ e % $Vp1$ r n % $Mp2$ a % $Vp2$ $-$ #3.

3.2.2 Begränsande regler

Jag fann två typer av restriktioner: de som kräver att en enhet följs av en annan och de som kräver att en enhet föregås av en annan. Två typer av regler skapades för att ta hand om dessa två fall. I figur 3.2 och 3.5 finns exempel för regeltyp 1 respektive 2. V , M och H står för vänster-, mitt- respektive högerled. FSA:er för exempelreglerna i figur 3.2 och 3.5 finns i figur 3.3 respektive 3.6.

¹<http://www.speech.kth.se/wavesurfer/>

²I detta skede har inte reglerna för retroflexivering applicerats, varför det bara finns ett möjligt uttal.

Tabell 3.1: Uppmärkning av enheter med förklaring och motsvarande enhet.

Uppmärkning	Förklaring	Motsvarande enhet
%I	Initialt konsonantkluster.	I
%Ma	Medialt konsonantkluster före V_0 .	M_a
%Mp1	Medialt konsonantkluster på position 1 efter V_0 .	M_{p1}
%Mp2	Medialt konsonantkluster på position 2 efter V_0 .	M_{p2}
%Mp3	Medialt konsonantkluster på position 3 efter V_0 .	M_{p3}
%Va	Vokal före V_0 .	V_a
%V0	Betonad vokal.	V_0
%Vp1	Vokal på position 1 efter V_0 .	V_{p1}
%Vp2	Vokal på position 2 efter V_0 .	V_{p2}
%Vp3	Vokal på position 3 efter V_0 .	V_{p3}
#1	Finalt konsonantkluster på position 1 efter V_0 .	F_1
#2	Finalt konsonantkluster på position 2 efter V_0 .	F_2
#3	Finalt konsonantkluster på position 3 efter V_0 .	F_3
#4	Finalt konsonantkluster på position 4 efter V_0 .	F_4

Figur 3.4 och 3.7 visar pseudo-FSA:er för dessa regler. Nedsänkt n innebär att det kan förekomma flera tillstånd efter varandra. De streckade bågarna och tillstånden kan förekomma, om ledet innehåller fler än en symbol

$$\begin{aligned}
 V &= \{\text{sp1}\} \\
 M &= \{\%I\} \\
 H &= \{i:, i, ä:, e\}
 \end{aligned}$$

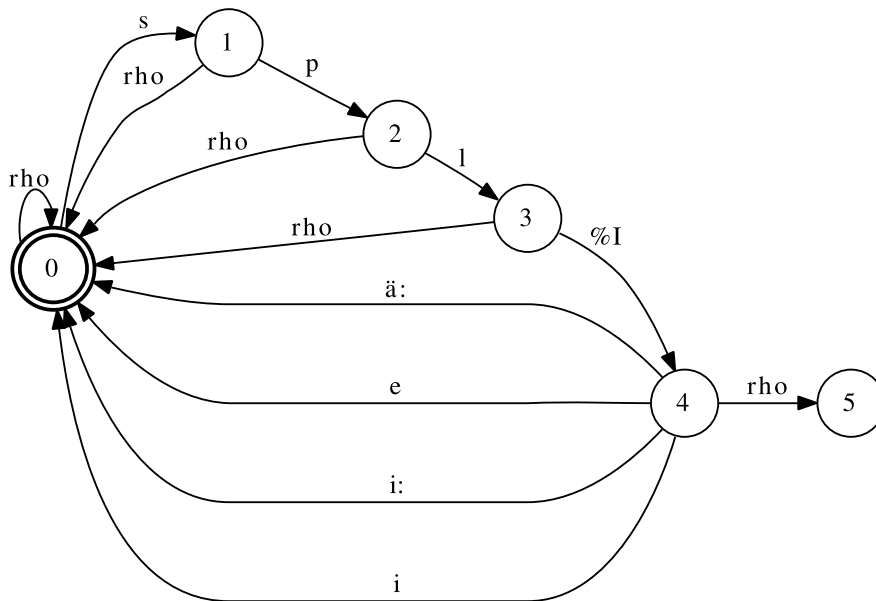
Figur 3.2: Exempel på hur en regel av typ 1 kan se ut. Detta är för vad som kan följa initialklustret sp1.

Reglerna för I-V-kombinationer implementerades som regeltyp 1, men då bara för instanser av I som hade några restriktioner, d.v.s. vr, spr, sp1, sv, tv, dv, kv, skv, fj, mj, nj, bj, pj, spj, fn, sk samt g. Man skulle kunna skapa en variant av regeltyp 1 som istället för att tillåta de högerled som definieras i regeln nekade de högerled som inte definieras, och använda denna för regler där det är fler högerled som tillåts än nekas. Jag valde dock att bara använda den ursprungliga regeltypen eftersom H aldrig blir särskilt stor och för att vara konsekvent.

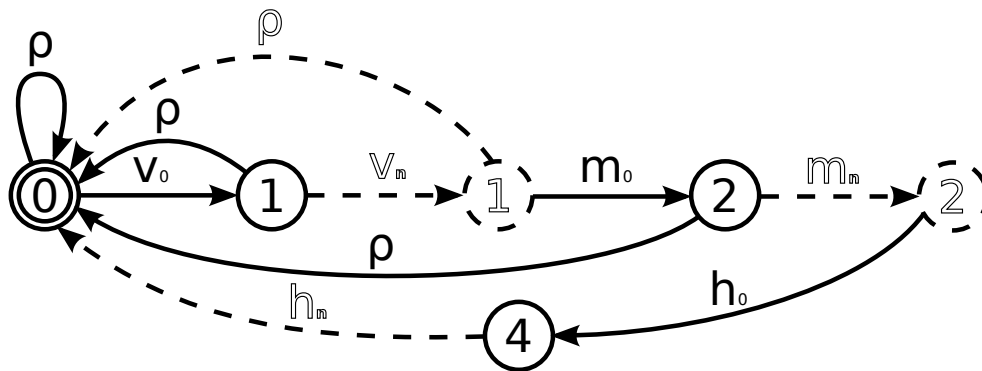
De två reglerna för kombinationer av V_0 -C implementerades som typ 2 regler.

3.2.3 Modifierande regler

Till skillnad från de övriga reglerna implementeras de modifierande reglerna som FST:er, då de ändrar på symboler. En annan sak som är annorlunda är att dessa regler inte tar hänsyn till var i ett ord fonen förekommer. Därför är uppmärkningen som finns i de andra reglerna inte med i beräkningen.



Figur 3.3: En FSA som visar hur exempelregeln i figur 3.2 realiseras.



Figur 3.4: Regeltyp 1 som en pseudo-FSA.

$V = \{i, e, y, ö, a, å, o, u\}$

$M = \{\%V0\}$

$H = \{ng, j\}$

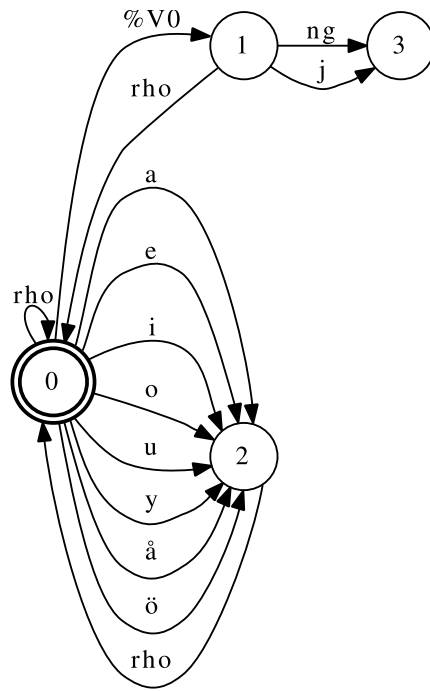
Figur 3.5: Exempel på hur en regel av typ 2 kan se ut. Detta är regeln som bara tillåter kort vokal före ng och j.

Retroflexivering

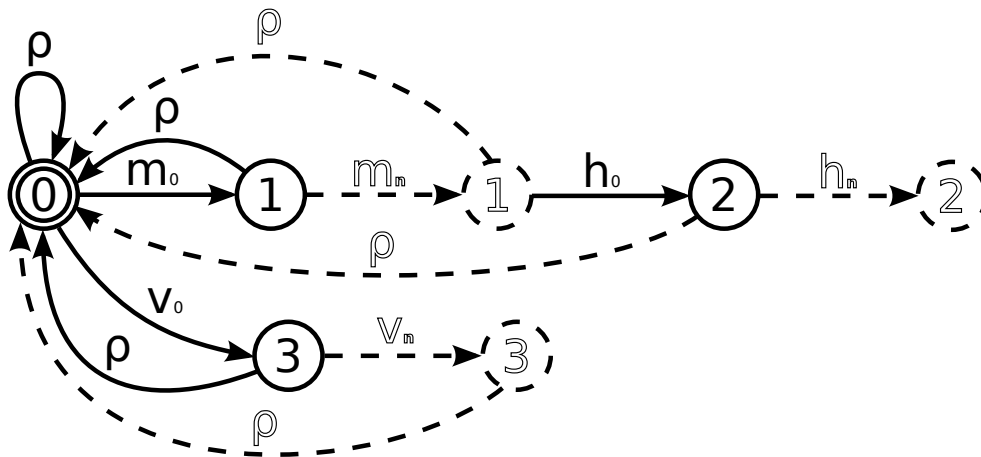
$r \in \{d, l, n, s, t\}$

$r' \in \{r, rd, rl, rn, rs, rt\}$

$q_r \in \{q_2, q_3, q_4, q_5, q_6\}$



Figur 3.6: En FSA som visar hur exempelregeln i figur 3.5 realiseras.

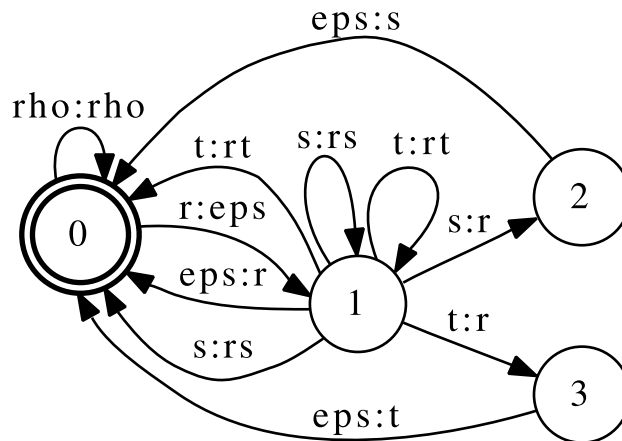


Figur 3.7: Regeltyp 2 som en pseudo-FSA.

FST:n för retroflexivering, figur 3.8, tillåter allt förutom r med bågen (rho:rho). Så fort den läser ett r går FST:n till q_1 och skriver eps. I detta tillstånd kan den gå tillbaka för alla symboler som inte är någon av de konsonanter som kan retroflexiveras, r. Detta ger stigen (r:eps, eps:r), vilket ju är ekvivalent med (r:r).

Läser FST:n däremot r ges flera möjliga stigar. Den kan gå tillbaka till q_0 och skriver då den retroflex som motsvarar. Tar man s som exempel blir stigen: (r:eps, s:rs). Input (r, s) kan således ge output (rs).

Eftersom det inte är säkert att retroflexivering infaller bara för att möjligheten ges finns det en alternativ väg från q_1 . Denna går till q_r när r läses och skriver r. Från q_r läses eps och r skrivs. Detta ger stigen: (r:eps, s:r,



Figur 3.8: Retroflexiveringsreglen som en FST. För att få en någorlunda tydlig bild är bara s och t med. I hela FST:n är stigar för alla r med.

eps:s). Alltså kan input (r, s) ge output (r, s).

Den sista bågen från q_1 , som är en loop, gör det möjligt för flera retroflexiveringar att följa på varandra.

Tabellerna 3.2, 3.3 och 3.4 visar de tre olika output retroflexiveringsreglen kan ge för ordet *torsdag*³.

Tabell 3.2:

Intillstånd	Uttillstånd	Insymbol	Utsymbol	Stig
0	0	t	t	t
0	0	o:	o:	t o:
0	1	r	ε	t o: ε
1	2	s	r	t o: ε r
2	0	ε	s	t o: ε r s
0	0	d	d	t o: ε r s d
0	0	a:	a:	t o: ε r s d a:
0	0	g	g	t o: ε r s d a: g

Vokalsänkning

$$r \in \{r, rd, rl, rn, rs, rt\}$$

$$v \in \{\ddot{a}:, e, \ddot{o}:, \ddot{ö}\}$$

$$v' \in \{\ddot{a}2:, \ddot{ä}2, \ddot{o}2:, \ddot{ö}2\}$$

Även regeln för vokalsänkning implementeras som en FST, figur 3.9. På samma sätt som med retroflexiveringen finns det här loop för q_0 för alla symboler som inte är en av de vokaler som påverkas av vokalsänkning.

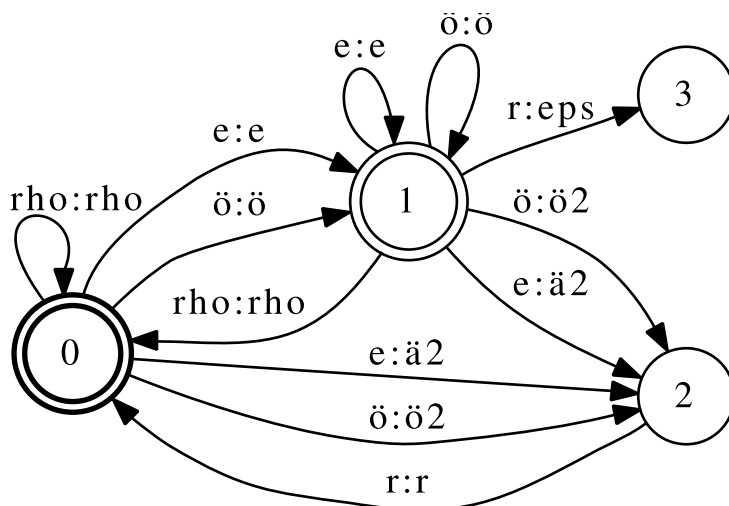
³Jag har inte brytt mig om hur resten av ordet ser ut i dessa exempel. Egentligen är det nog troligare att iallafall exempel 3 uttalas [toʂd̥a] snarare än [toʂd̥u:g], som det gör här, eftersom både retroflexivering och bortfallet av [g] samt modifieringen av [a:] är uttalslättnader som uppkommer i vardagligt tal.

Tabell 3.3:

Intillstånd	Uttillstånd	Insymbol	Utsymbol	Stig
0	0	t	t	t
0	0	o:	o:	t o:
0	1	r	ϵ	t o: ϵ
1	0	s	rs	t o: ϵ rs
0	0	d	d	t o: ϵ rs d
0	0	a:	a:	t o: ϵ rs d a:
0	0	g	g	t o: ϵ rs d a: g

Tabell 3.4:

Intillstånd	Uttillstånd	Insymbol	Utsymbol	Stig
0	0	t	t	t
0	0	o:	o:	t o:
0	1	r	ϵ	t o: ϵ
1	1	s	rs	t o: ϵ rs
1	0	d	rd	t o: ϵ rs rd
0	0	a:	a:	t o: ϵ rs rd a:
0	0	g	g	t o: ϵ rs rd a: g



Figur 3.9: Vokalsänkingsregeln som en FST. För att få en någorlunda tydlig bild är bara e och ö, samt r. I hela FST:n är stiggar för alla v och r med.

När v läses finns det två möjliga bågar. Den ena leder till q_2 och skriver motsvarande sänkt vokal: v' . Från q_2 kan FST:n bara ta r som input. Detta innebär att en sänkt vokal måste följas av en retroflex. Detta ger, för t.ex. vokalen \ddot{o} , stigen ($\ddot{o}:\ddot{o}2$, $r:r$). Input (\ddot{o} , r) kan således ge output ($\ddot{o}2$, r).

En annan möjlighet vid läsning av v i q_0 är att gå till q_1 och skriva v . I q_1 finns en loop för att tillåta flera v på rad. Skulle sedan v följas av v' finns det också en båge från q_1 till q_2 . Eftersom v inte får följas av r finns det en båge från q_1 till q_3 , som är en återvändsgränd; det går inte att komma till ett finalt

tillstånd från denna.

Den sista bågen från q_1 går till q_0 och både läser och skriver ρ . Denna återgår alltså till den ursprungliga loopen.

Tabell 3.5 visar hur order *dörr* behandlas av regeln för vokalsänkning.

Tabell 3.5:

Intillstånd	Uttillstånd	Insymbol	Utsymbol	Stig
0	0	d	d	d
0	2	ö	ö2	d ö2
2	0	r	r	d ö2 r

3.2.4 Kombination av regler

När alla reglerna var färdiga kombinerades de till en enda FSA. Denna motsvarar svensk fonotax, som den definieras i avsnitt 2.2. Denna FSA kan sedan implementeras i ett taligenkänningsystem genom att komponera in den i kaskaden.

Man kan se varje regel av en typ som en regelgrupp, då regler av samma typ implementeras på samma sätt. De genererande reglerna, g , är utgångspunkten. Sedan tas snittet $g \cap b$, där b är de begränsande reglerna, för att bara tillåta de ord som uppfyller kraven för både g och b . Innan de modifierande reglerna, m , appliceras måste uppmärkningen tas bort. Detta sker genom att en FST, u , komponeras. u skriver om alla instanser av uppmärkning till ϵ . Sedan komponeras m vilket ger en FST, $((g \cap b) \circ u) \circ m$. Outputen från denna används sedan som en FSA, vilken bara accepterar ord enligt fonotaxen.

Eftersom reglerna inte skapats för att vara så effektiva som möjligt, sett till antalet tillstånd, utan för att vara enkla att skriva och överskådliga, finns det en hel del redundans i denna FSA. Detta går dock enkelt att åtgärda med hjälp av FSM-operationer. De operationer som användes för detta var *epsilon removal*, *determinization* och *minimization*.

Även den minimerade FSA:n är på tok för stor för att kunna visas grafiskt, men för att få en uppfattning av den kan nämnas att den har 7281 transitioner och 206 tillstånd, av vilka 130 är finala.

4 Utvärdering

Det är inte helt självklart hur man kan utvärdera en fonotaktisk modell, då detta skulle kräva ett facit över uttalet för ett visst språk, i fallet för denna uppsats svenska. Även om man lyckades få fram en entydig fonotax, som alla höll med om, skulle man behöva skapa en lista över alla enkla ord som följer denna.

För att ändå få någon form av bedömning av den fonotaktiska modellen utvärderades denna mot orden i ett manuellt annoterat ytlexikon. Detta gjordes genom att undersöka hur många av orden i denna ordlista som godkändes av modellen. Värden för denna undersökning och ordlistan finns i tabell 4.1. Som synes ges en hög täckning av denna ordlista, vilket tyder på att fonotaxen klara av den väl. Efter att ha undersökt de ord som inte täcktes av fonotaxen kunde jag konstatera att många av dessa innehöll xenofoner, d.v.s. foner som inte räknas till svensk fonologi, vilka inte togs med vid skapandet av modellen.

Tabell 4.1: Utvärdering av modellen mot en ordlistan från ytlexikon.

Antal ord	Ordlängd	Foner	Bigram	Trigram	Täckning
9459	1–19	52	1051	5586	97,65%

Det mycket svårt att undersöka huruvida modellen är mer tillåtande än den borde vara. Trots detta ville jag kontrollera att den inte tillåter allt möjligt. För detta syfte skapades en lista med slumpmässigt genererade strängar. Dessa skapades som helt godtyckliga sekvenser av en till tjugo av fonerna som ingår i modellens alfabet. Statistik för denna lista och modellens täckning av dessa sekvenser finns i tabell 4.2.

Relevanta skillnader mellan den slumpgenererade ordlistan och ytlexikonet är dels antalet ord och antalet foner. Att antalet ord är såpass mycket färre beror på tekniska begränsningar vid behandlingen av den slumpgenererade ordlistan. Den mindre mängden foner beror på att bara de foner som finns i modellen användes vid genereringen av strängarna. Den helt godtyckliga kombinationen av foner medför också att antalet bi- och trigram blir fler, ungefär dubbelt så många i båda fallen, för denna lista. Det låga värdet på täckningen av denna lista tyder på att modellen ändå är ganska restriktiv. Nämnas bör dock att det i denna lista kan förekomma strängar som kan anses som mer eller mindre omöjliga att uttala.

Jämför man täckningen för dessa två ordlistor för modellen med täckningen för helt fri fonotax märker man att den inte uppnår samma täckning av orden i ytlexikonet. En fri fonotax skulle ju täcka alla dessa. Modellen har dock en mycket hög täckning. Eftersom en minoritet av de slumpmässiga strängarna i

Tabell 4.2: Utvärdering av modellen mot en slumpgenererad ordlista.

Antal ord	Ordlängd	Foner	Bigram	Trigram	Täckning
1000	1–20	47	2097	8172	11,90%

den slumpgenererade ordlistan bör vara sådana som täcks av en svensk fonotax talar den låga täckningen för modellen. En fri fonotax godkänner alla sekvenser även i denna lista.

4.1 Fonemigenkänning

En annan typ av utvärdering som gjordes av modellen var att inkorporera den i en fonemigenkännare. Detta för att undersöka om modellen kunde förbättra denna igenkänning. För denna utvärdering användes datan beskriven i avsnitt 3.1.2. Först kördes vanlig fonemigenkänning på denna och resultatet jämfördes med fonemigenkänning med modellen. Resultatet består av Phoneme Error Rate (**PER**), vilket fungerar som Word Error Rate (**WER**) (Jurafsky och Martin, 2009, s. 362), fast på fonemnivå. Resultatet av dessa körningar står att finna i tabell 4.3.

För att få fram dessa siffror gjordes upprepade körningar, med varierade parametrar, för båda modellerna för att få fram de bästa resultaten. Anmärkas kan det faktum att det finns en viss obalans mellan *insertion* och *deletion* i båda fallen. Detta beror på att om man försökte få dessa jämnare, vilket skulle kunna ses som ett sätt att minimera antalet fel, ökade det totala antalet fel p.g.a. ökning av *substitution*, vilken också är den överlägset största felkategorin.

Tabell 4.3: Utvärdering av fonemigenkänning med och utan den fonotaktiska modellen.

Körning	PER	Antal fel	Ins.	Del.	Sub.
Vanlig fonemigenkänning	38,58%	784	93	185	506
Fonotax	37,25%	757	72	189	496

Som synes ger den fonotaktiska modellen än förbättring, om än en blygsam sådan.

5 Diskussion

Resultatet av denna uppsats visar på att det med ganska enkla medel går att skapa en fonotaktisk modell, som går att implementera i ett taligenkänningsystem. Modellen som skapades för denna uppsats fick en hög (97,65%) täckning mot guldstandard samtidigt som den fick en låg (11,90%) täckning mot slumpmässiga fonsekvenser.

Resultatet visar också på att en sådan modell kan förbättra fonemigenkänning. Modellen i denna uppsats gav en förbättring på 1,33% då den inkorporerades i ett system för fonemigenkänning.

Ett första steg för vidare utveckling av modellen bör vara att se över den teoretiska grunden för fonotaxen som används. Jag valde att utgå nästan uteslutande från en och samma källa, Sigurd (1965), eftersom jag ville vara säker på att hinna skapa en modell som representerade denna väl och testa den i ett system. Ytterligare data för modellen skulle kunna inhämtas antingen från annan litteratur eller göra empiriska undersökningar av uttal.

Särskilt i det senare fallet skulle man också kunna få fram statistik över hur vanliga olika kombinationer av enheter är. Detta skulle ge möjligheten att ta fram en viktad modell, snarare än som den är nu, då den bara tillåter eller inte tillåter enheter. Detta skulle kunna göras med n-gram på fonemnivå, men gissningsvis skulle resultatet bli bättre om man istället gjorde detta på enhetsnivå. Annoteringen av enheterna borde också kunna användas för detta. Det finns också i Sigurd (1965) statistik över hur olika enheter förekommer på olika positioner vilket borde kunna användas vid viktning.

Vid vidare utveckling av modellen vore det bra att ha mer data i form av inspelat, uppmärkt tal. Detta vore särskilt användbart om man väljer att gå vidare med maskininlärning, då man behöver stora mängder av träningsdata samt testdata.

Något som jag märkte under skapandet av modellen var att den kan tillåta ett ord som inte följer fonotaxen genom att dela upp det i flera mindre ord. Det är dock svårt att hitta ordgränser i tal, utom vid diktering. Däremot borde man kunna få modellen att premiera längre ord för att undvika att den delar upp allt i mycket små delar.

Litteraturförteckning

- Eisner, Jason. Parameter Estimation for Probabilistic Finite-State Transducers. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2002.
- Eklund, Robert och Lindström, Anders. How To Handle “Foreign” Sounds in Swedish Text-to-Speech Conversion: Approaching the ‘Xenophone’ Problem. *Proceedings of ICSLP 98*, 1998.
- Engstrand, Olle. *Fonetikens grunder*. Studentlitteratur, 2004.
- Hunt, Andrew och McGlashan, Scott. Speech Recognition Grammar Specification Version 1.0, Mars 2004. <http://www.w3.org/TR/speech-grammar/>.
- Jurafsky, Danial och Martin, James H. *Speech and Language Processing*. Prentice Hall, 2009. ISBN 0-13-504196-1, 978-0-13-504196-3.
- Linell, Per, Svensson, Bengt, och Öhman, Sven. *Ljudstruktur*. Gleerups, 1971.
- Seward, Alexander. *Efficient Methods for Automatic Speech Recognition*. Univeritetservice US AB, 2003. ISBN 91-7283-657-1.
- Sigurd, Bengt. *Phonotactic Structures in Swedish*. Berlingska boktryckeriet, 1965.

A Enheter

A.1 I

-	f	gr	l	pr	sm	str
b	fj	h	m	r	sn	t
bj	fl	j	mj	s	sp	tj
bl	fn	k	n	sj	spj	tr
br	fr	kl	nj	sk	spl	tv
d	g	kn	p	skr	spr	v
dr	gl	kr	pj	skv	sr	vr
dv	gn	kv	pl	sl	st	

A.2 F_1

-	js	lms	mnds	ng	rft	rts
b	jsk	lmsk	mns	ngd	rfts	rv
bs	jskt	ln	mnt	ngds	rg	rvd
bsk	jst	lns	mp	ngk	rgs	rvs
bt	jsts	lp	mps	ngks	rj	rvt
bts	jt	lps	mpt	ngkt	rjs	s
d	jts	lpt	ms	ngkts	rjt	sk
ds	k	ls	msk	ngn	rk	sks
dsk	ks	lsk	mskt	ngns	rks	skt
f	kst	lskt	mst	ngnt	rkt	sp
fs	ksts	lst	msts	ngs	rl	sps
fst	kt	lsts	mt	ngsk	rls	st
fsts	kts	lt	mts	ngst	rm	sts
ft	l	lts	n	ngsts	rm d	sj
fts	lb	lv	nd	ngt	rms	sjs
g	lbs	lvd	nds	ngts	rmst	sjt
gd	ld	lvs	ndsk	p	rm t	sjts
gds	lds	lvt	nj	ps	rn	t
gs	ldst	m	njd	psk	rns	tm
gst	lf	mb	njs	pst	rnt	ts
gt	lfs	mbs	njt	psts	rp	tsk
gts	lft	md	ns	pt	rps	tsj
j	lfts	m ds	nsk	pts	rpt	tsjs
jd	lj	mf	nskt	r	rs	v
jds	ljd	mfs	nst	rb	rsk	vd
jf	ljds	mj	nsts	rbs	rskt	vds
jfs	ljs	mjd	nsj	rbt	rst	vs
jk	ljt	mjs	nsjs	rd	rsts	vst
jks	lk	mjt	nt	rds	rsj	vt
jp	lks	mn	nts	rf	rsjs	vts
jps	lm	mnd	nts j	rfs	rt	

A.3 F_2

-	gs	ln	ns	r	sk
b	gt	ls	nt	rd	st
d	k	lt	ng	rn	sj
ds	ks	m	ngs	rs	t
f	kt	ms	p	rt	ts
g	l	n	ps	s	v

A.4 F_3

-	k	n	s
g	m	r	t

A.5 F_4

- n t

A.6 M_a

-	gn	lj	ms	ngdr	rd	sh
b	gr	lk	mst	ngg	rf	sj
bd	j	lkr	mt	nggg	rg	sk
bf	jk	lm	mv	nggl	rj	skr
bj	jm	ln	n	nggv	rk	skv
bk	js	lp	nd	ngh	rk m	sl
bl	jt	lpl	ndr	ngk	rks	sm
bm	k	lpt	ndsj	ngkl	rl	sp
bn	kd	ls	nf	ngkr	rm	spl
br	kl	lsj	nfl	ngksj	rn	spr
bskr	kn	lt	nh	ngkt	rp	sr
bstr	kr	ltr	nj	ngkv	rpr	st
bv	ks	ltj	ns	ngl	rs	stj
d	kskl	lv	nsf	ngn	rsl	stm
dh	kskr	lvr	nsj	ngpl	rsm	str
dj	kspl	m	nsk	ngs	r sn	sjm
dm	kspr	mb	ns l	ngsn	r sp	t
dr	kstrl	mb l	ns m	ngsj	r sj	tl
ds	kstrr	mbr	nsp	ngt	rt	tm
dstr	ksj	md	nst	ngtr	rtf	tr
dv	kt	mf	ns v	ngv	rtl	ts
f	kv	mfl	nsj	p	rtr	tsj
fg	l	ml	nt	pl	rts	ttj
fl	lb	mn	ntg	pn	rtsj	v
fr	ld	mp	ntl	pr	rtj	vl
ft	ldn	mpl	ntr	ps	rv	vr
g	lf	mpr	ntj	psj	rvj	
gd	lfr	mps	nv	pt	s	
gl	lg	mpt	ng	r	sf	
gm	lh	mr	ngd	rb	sgr	

A.7 M_{p1}

-	j n t r	l k n	m p s	ng d	r j l	s k
b	j p	l m	m p t	ng g	r j n	s k l
bl	j p n	l m n	m r	ng k	r j s	s k n
bn	j r	l m s k	m s	ng k l	r j s n	s k r
br	j s	l n	m s k	ng k n	r k	s l
bsk	j s k	l p	m s k n	ng k r	r k l	s m
d	j s l	l p l	m s l	ng k s	r k n	s n
dg	j s m	l p n	m s n	ng k t	r k r	s p
dgn	j s n	l p t	m s t	ng l	r k t	s p l
dj	j s t	l r	m s t r	ng n	r l	s p n
dk	j s t r	l s	m t	ng r	r l n	s p r
dl	j t	l s k	m t l	ng s	r m	s r
dm	j t l	l s k l	m t n	ng s l	r m d	s t
dn	j t n	l s k n	n	ng s n	r m l	s t l
dr	k	l s l	n d	ng s t	r m n	s t m
ds	k d	l s n	n d l	ng s t r	r m s	s t n
dsk	k j	l s t	n d n	ng s j	r m s n	s t r
dsl	k l	l s t r	n d r	ng t	r m s t	s j
dsn	k m	l t	n d s	ng t n	r n	t
f	k n	l t j	n d s k	p	r n s l	t g
fl	k r	l t n	n d s l	p f	r p	t j
fn	k s	l t r	n f	p j	r p l	t j n
fr	k s l	l t s	n j	p l	r p n	t l
fs	k s n	l v	n j d	p n	r p r	t m
fsn	k s t	l v d	n l	p r	r p t	t n
fst	k s t l	l v l	n r	p s	r s	t r
ft	k s t r	l v n	n s	p s k	r s k	t s
ftl	k s j	l v r	n s k	p s l	r s k l	t s k
ftn	k t	m	n s k l	p s n	r s k n	t s k n
fts	k t l	m b	n s k n	p s t	r s l	t s l
g	k t n	m b n	n s l	p t	r s n	t s n
gd	k v	m b r	n s n	r	r s t	t s j
gl	l	m d	n s r	r b	r s t n	t s j n
gm	l b	m f	n s t	r b r	r s j	v
gn	l d	m f l	n s t l	r d	r t	v d
gr	l d l	m f n	n s t n	r d l	r t l	v d n
gs	l d n	m j	n s t r	r d n	r t n	v g m
gsl	l f	m j d	n s j	r d r	r t r	v j
gsn	l f r	m j n	n s j n	r d s l	r t s	v j n
gst	l f t	m k	n s j r	r f	r t s l	v k
j	l g	m k n	n t	r f n	r t s n	v l
jd	l g j	m l	n t g n	r f t	r v	v n
jdl	l j	m n	n t l	r f t l	r v d	v r
jdn	l j d	m n d	n t n	r g	r v l	v s
jf	l j n	m p	n t r	r g l	r v n	v s t
jk	l j r	m p l	n t s	r g n	s	v t
jkn	l k	m p n	n t s k	r j	s b	
jn	l k l	m p r	ng	r j d	s f	

A.8 M_{p2}

-	k	l	s	k	n	t	r	r	s	t	s	t
d	k	s	m	ng	r	d	r	t	s	j		
f	l	n	ng	s	k	r	n	s	t			
g	l	s	n	d	p	r	s	k	s	k	t	s

A.9 M_{p3}

k	m	r	r	n	s	t
---	---	---	---	---	---	---

A.10 V_0

a	e:	o	u:	ä:	ö
a:	i	o:	y	å	ö:
e	i:	u	y:	å:	AU
EU					

A.11 V_a

a	i	u	å	e
o	y	ö	AU	EU

A.12 V_{p1}

a	e	i	o	u	y	å
---	---	---	---	---	---	---

A.13 V_{p2}

a	e	i	o	u	å
---	---	---	---	---	---

A.14 V_{p3}

a	e
---	---

B Begränsande regler

I tabellerna nedan står enheterna för den första positionen överst och i den efterföljande till vänster. Giltiga kombinationer markeras med "o" och ogiltiga med "-". I B.1 och B.3 gäller kombinationsmöjligheterna för både långa och korta vokaler då dessa står i position V_0 .

B.1 I + V

	-	b	b	b	b	d	d	d	f	f	f	f	f	g	g	g	g	h	j
		j	l	r		r	v		j	l	n	r		l	n	r			
i	o	o	o	o	o	o	o	-	o	-	o	o	o	o	o	o	o	o	o
e	o	o	-	o	o	o	o	-	o	-	o	-	o	o	o	o	o	o	o
y	o	o	-	o	o	o	o	-	o	-	o	o	o	-	o	o	o	o	o
ö	o	o	o	o	o	o	o	-	o	o	o	o	o	-	o	o	o	o	o
ä	o	o	o	o	o	o	o	o	o	o	o	-	o	o	o	o	o	o	o
a	o	o	-	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
å	o	o	-	o	o	o	o	-	o	o	o	o	o	o	o	o	o	o	o
o	o	o	-	o	o	o	o	-	o	o	o	-	o	o	o	o	o	o	o
u	o	o	u	o	o	o	o	-	o	o	o	o	o	o	o	o	o	o	o
	k	k	k	k	k	l	m	m	n	n	p	p	p	p	r	s	s	s	s
	l	n	r	v			j		j		j	l	r			j	k	k	r
i	o	o	o	o	o	o	o	o	o	-	o	-	o	o	o	o	o	o	o
e	o	o	o	o	o	o	o	-	o	-	o	-	o	o	o	o	o	o	o
y	o	o	o	o	-	o	o	-	o	-	o	-	o	o	o	o	o	o	o
ö	o	o	o	o	-	o	o	o	o	o	o	-	o	o	o	o	o	o	-
ä	o	o	o	o	o	o	o	o	o	-	o	o	o	o	o	o	o	o	o
a	o	o	o	o	o	o	o	-	o	-	o	o	o	o	o	o	o	o	o
å	o	o	o	o	-	o	o	-	o	-	o	o	o	o	o	o	o	o	o
o	o	o	o	o	o	o	o	-	o	o	o	-	o	o	o	o	o	o	o
u	o	o	o	o	-	o	o	u	o	o	o	o	o	o	o	o	o	o	o
	s	s	s	s	s	s	s	s	s	s	s	t	t	t	t	v	v		
	k	l	m	n	p	p	p	p	t	t	v		j	r	v		r		
	v				j	l	r		r										
i	o	o	o	o	o	-	o	o	o	o	o	o	o	o	o	o	o	o	o
e	-	o	o	o	o	-	-	o	o	o	o	o	o	o	o	o	o	o	o
y	-	o	o	o	o	-	-	o	o	o	-	o	o	o	-	o	-		
ö	-	o	o	o	o	-	-	o	o	o	-	o	o	o	-	o	o		
ä	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
a	o	o	o	o	o	-	-	o	o	o	o	o	o	o	o	o	o	o	o
å	-	o	o	o	o	-	-	o	o	o	o	o	o	o	o	o	o	o	o
o	-	o	o	o	o	-	-	-	o	o	o	o	o	o	o	-	o	o	
u	-	o	o	o	o	o	-	o	o	o	o	o	o	o	o	o	o	-	

B.2 $V_0 + C$

	a	e	i	o	u	y	å	ö	a:	e:	i:	o:	u:	y:	å:	ä:	ö:
-	-	-	-	-	-	-	-	-	o	o	o	o	o	o	o	o	o
d	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
f	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
g	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
j	o	o	o	o	o	o	o	o	-	-	-	-	-	-	-	-	-
k	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
l	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
m	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
n	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
ng	o	o	o	o	o	o	o	o	-	-	-	-	-	-	-	-	-
p	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
r	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
s	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
sj	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
t	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o

B.3 $V_0 + V_{p1}$

	a	e	i	o	u	y	å	ä	ö
a	-	o	o	o	o	o	o	o	o
e	o	o	o	o	o	o	o	o	o
i	o	o	o	o	o	o	o	o	o
å	o	o	o	-	o	o	o	-	-
o	o	o	o	-	o	o	o	-	-
u	-	o	o	-	-	-	-	-	-

B.4 $V_{p1} + C$

	a	e	i	u	å	o
-	o	o	o	o	o	o
d	o	-	o	o	o	o
f	-	-	-	-	o	o
g	-	-	o	o	o	o
k	o	o	o	o	-	-
l	o	o	o	o	o	o
m	o	o	o	o	o	o
n	o	o	o	o	o	o
ng	-	-	o	o	-	-
p	o	-	-	-	o	o
r	o	o	o	o	o	o
s	o	o	o	o	o	o
sj	-	-	o	-	-	-
t	o	o	o	o	o	o

B.5 $V_{p1} + V_{p2}$

	a	e	i	o	u	y	å
a	-	o	o	o	o	-	o
e	o	-	o	o	-	-	o
o	o	o	o	-	-	o	-
u	o	o	o	-	o	-	-
å	o	o	o	-	-	o	-