protein sequences. support Sequences. s_{as} such a start symbol S , and rules R (above). r_{as} $r_{as} = \langle S \rangle = \langle S \rangle = \langle S \rangle \langle crnpity \rangle$ $r_{as} = \langle S \rangle = a \langle S \rangle b \langle S \rangle \langle crnpity \rangle$ $r_{as} = \langle S \rangle = a \langle S \rangle = a \langle S \rangle b \langle S \rangle \langle crnpity \rangle$	$ \begin{array}{c} \mbox{Context sensitive grammars} & \mbox{unrestricted grammars} & \mbox{unrestricted grammars} & \mbox{unrestricted grammars} & \mbox{grammars} & \mbox{grammars} & \mbox{grammars} & \mbox{grammars} & \mbox{and for arbitrary sequences of term} \\ \hline \mbox{Notes} & \mbox{equal} & \mbox{Regular grammars} & \mbox{and context free grammars} & \mbox{are also called term rewrit} & \mbox{equal} & \mbox{only part of programming language syntax is des} & \mbox{lunrestricted grammars} & \mbox{are also called term rewrit} & \mbox{equal} & \mbox{commars} & \mbox{are also called term rewrit} & \mbox{equal} & \mbox{commars} & \mbox{are also called term rewrit} & \mbox{equal} & \mbox{commars} & \mbox{are also called term rewrit} & \mbox{commars} & \mbox{are automata} & \mbox{The FMR-1 triplet region contains repeats of the triple} & GCG CTG, GCG CGG CTG, GCG CG CG CG CGG CTG, GCG CG CG CGG CTG, GCG CG CG$	• Stochastic regular grammars • Stochastic context free grammars; the CYK algorithm
sestor@dina.kvi.dt py models stochastic context free grammars stochastic context free grammars stochastic context free grammars p_{1} p_{2} p_{2} p_{3} $p_{$	of of <i>G</i> .	
second Gdina.kvi.det stochastic context free grammars stochastic context free grammars stochastic context free grammars $P_{age 1}$ $P_{age 1}$	empty)	
second definition of the grammars subsecting dima kvidet for models P_{age1} P_{age1} for P_{age1} $P_$	(empty)	
e). Page 1 V	(empty)	
second editina Avridek arkov models by stochastic context free grammars by stochastic context free grammars in trees archy archy res archy res archy res	(empty)	
©dina.kvl.dk	(empty)	
@dina.kvl.dk Page 1 KVL	(empty)	
Page 1 RC		
Page 1		
mastic grammars and RNA nutshell runshell rimary sequence structure is described by hidden Markov models idden Markov models are stochastic regular grammars secondary sequence structure (of RNA) is described by stochastic context free grammars and finite state automata context free grammars and finite state automata stochastic regular grammars iochastic regular grammars econstruction of derivation trees iochastic context free grammars; the CYK algorithm stochastic context free grammars; the CYK algorithm iochastic context free grammars; the CYK algorithm stochastic sequence structure for stochastic sequence structure free grammars; the CYK algorithm mmar $G = (N, T, R, S)$ $S \rightarrow aS$ $S \rightarrow b$ $S \rightarrow b$		• Stochastic regular grammars • Stochastic context free grammars; t A grammar $G = (N, T, R, S)$
mastic grammars and RNA nutshell runshell Intervention of described by hidden Markov models idden Markov models are stochastic regular grammars idden Markov models context free grammars generalize regular grammars is described by stochastic context free grammars iecondary sequence structure (of RNA) is described by stochastic context free grammars and finite state automata intervention iontext free grammars and finite state automata its context free grammars its context free grammars iontext free grammars and reconstruction of derivation trees its context free grammars; the CYK algorithm Faget ischastic context free grammars; the CYK algorithm Faget Faget Faget		• Stochastic regular grammars • Stochastic context free grammars; t A grammar $G = (N, T, R, S)$
mastic grammars and RNA result of Grina.Nut. nutshell Initian y sequence structure is described by hidden Markov models idden Markov models are stochastic regular grammars iontext free grammars generalize regular grammars iecondary sequence structure (of RNA) is described by stochastic context free grammars iontext free grammars iontext free grammars and inite state automata iontext free grammars iontext free grammars and reconstruction of derivation trees iontext free grammars; the CYK algorithm iontext free grammars; the CYK algorithm Markor computational bakey (1995-12.20 Pape 1 Nummar $G = (N, T, R, S)$ $S \rightarrow aS$ S S		• Stochastic regular grammars • Stochastic context free grammars; t A grammar $G = (N, T, R, S)$
nastic grammars and RNA second drug work runshell rimary sequence structure is described by hidden Markov models idden Markov models are stochastic regular grammars sometry sequence structure (of RNA) is described by stochastic context free grammars iecondary sequence structure (of RNA) is described by stochastic context free grammars secondary sequence structure (of RNA) is described by stochastic context free grammars iecondary sequence structure (of RNA) is described by stochastic context free grammars sometry is and finite state automata isochastic regular grammars and reconstruction of derivation trees isochastic context free grammars; the CYK algorithm service or computational below reservice isochastic context free grammars; the CYK algorithm Page 1 Marmar $G = (N, T, R, S)$ $S \rightarrow aS$		• Stochastic regular grammars • Stochastic context free grammars; t A grammar $G = (N, T, R, S)$
mastic grammars and RNA second d'ha Avd.dt. nutshell "immary sequence structure is described by hidden Markov models it described by hidden Markov models are stochastic regular grammars. iciden Markov models are stochastic regular grammars "immars. icondary sequence structure (of RNA) is described by stochastic context free grammars. "immars. derivation, parsing, and the Chomsky hierarchy iegular grammars and finite state automata "interaction of derivation trees "interaction of derivation trees icochastic regular grammars. the CYK algorithm mar Tot icochastic context free grammars. Total tree grammars. Total tree grammars. Total tree grammars.	R P	• Stochastic regular grammars • Stochastic context free grammars; t A grammar $G = (N, T, R, S)$
wastic grammars and RNA second@dina.kvidk rimmary sequence structure is described by hidden Markov models idden Markov models are stochastic regular grammars iontext free grammars generalize regular grammars secondary sequence structure (of RNA) is described by stochastic context free grammars and finite state automata iontext free grammars and finite state automata .ontext free grammars and reconstruction of derivation trees iontext free grammars; the CYK algorithm .ontext free grammars; the CYK algorithm		 Stochastic regular grammars Stochastic context free grammars; t
sesott@dina.twi.dk nutshell virimary sequence structure is described by hidden Markov models idden Markov models are stochastic regular grammars context free grammars generalize regular grammars iecondary sequence structure (of RNA) is described by stochastic context free grammars promars, derivation, parsing, and the Chomsky hierarchy regular grammars and finite state automata context free grammars and reconstruction of derivation trees itochastic context free grammars; the CYK algorithm itochastic context free grammars; the CYK algorithm	Context sensitive grammars $\alpha_1 W \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$, non- Unrestricted grammars $\alpha_1 W \alpha_2 \rightarrow \gamma$ Above, α, β etc. stand for arbitrary sequences of terminals and nonterminals. Notes • Regular grammars and context free grammars are widely used in computer s • Programming languages are described by the so-called <i>LR</i> or <i>LL</i> subclassi • Only part of programming language syntax is described by the <i>LR</i> -grammar • Unrestricted grammars are also called term rewrite systems • Chomsky's original goal was to describe natural languages. He (and everybo)	
sesof@dina.kvi.dk nutshell vimary sequence structure is described by hidden Markov models lidden Markov models are stochastic regular grammars context free grammars generalize regular grammars iecondary sequence structure (of RNA) is described by stochastic context free grammars brammars, derivation, parsing, and the Chomsky hierarchy legular grammars and finite state automata context free grammars and reconstruction of derivation trees itochastic regular grammars	Context sensitive grammars $\alpha_1 W \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$, non Unrestricted grammars $\alpha_1 W \alpha_2 \rightarrow \gamma$ Above, α, β etc. stand for arbitrary sequences of terminals and nonterminals. Notes • Regular grammars and context free grammars are widely used in computer s • Programming languages are described by the so-called <i>LR</i> or <i>LL</i> subclass • Only part of programming language syntax is described by the <i>LR</i> -gramman • Unrestricted grammars are also called term rewrite systems	
nutshell .uutshell rimary sequence structure is described by hidden Markov models lidden Markov models are stochastic regular grammars context free grammars generalize regular grammars secondary sequence structure (of RNA) is described by stochastic context free grammars secondary sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (of RNA) is described by stochastic context free grammars sequence structure (sequence structure) sequence structure (sequence st	Context sensitive grammars $\alpha_1 W \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$, non Unrestricted grammars $\alpha_1 W \alpha_2 \rightarrow \gamma$ Above, α, β etc. stand for arbitrary sequences of terminals and nonterminals.Notes• Regular grammars and context free grammars are widely used in computer s • Programming languages are described by the so-called LR or LL subclass• Only part of programming language syntax is described by the LR -grammar	
sestof@dina.kvl.dk nutshell rimary sequence structure is described by hidden Markov models idden Markov models are stochastic regular grammars ontext free grammars generalize regular grammars iecondary sequence structure (of RNA) is described by stochastic context free grammars iecondary sequence structure (of RNA) is described by stochastic context free grammars iecondary and the Chomsky hierarchy tegular grammars and finite state automata	Context sensitive grammars $\alpha_1 W \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$, non- Unrestricted grammars $\alpha_1 W \alpha_2 \rightarrow \gamma$ Above, α, β etc. stand for arbitrary sequences of terminals and nonterminals. Notes • Regular grammars and context free grammars are widely used in computer s • Programming languages are described by the so-called LR or LL subclass	 Context free grammars and reconstruction of derivation trees
sestof@dina.kvl.dk hastic grammars and RNA nutshell rimary sequence structure is described by hidden Markov models lidden Markov models are stochastic regular grammars Ontext free grammars generalize regular grammars Context free grammars generalize regular grammars Sontext free grammars generalize regular grammars Secondary sequence structure (of RNA) is described by stochastic context free grammars Frammars, derivation, parsing, and the Chomsky hierarchy regular grammars and finite state automata	$\begin{tabular}{ c c c c c } \hline Context sensitive grammars & $\alpha_1W\alpha_2 \longrightarrow \alpha_1\beta\alpha_2$, non $$$ Unrestricted grammars & $\alpha_1W\alpha_2 \longrightarrow \gamma$$ Above, $\alpha, β etc. stand for arbitrary sequences of terminals and nonterminals. $$$ Notes $$ e Regular grammars and context free grammars are widely used in computer $$ so-called LR or LL subclass $$ e Programming languages are described by the so-called LR or LL subclass $$ are described by the so-called LR or LL subclass $$ e Programming languages are described by the so-called LR or LL subclass $$ are described by the so-called LR or LL subclass $$ end{tabular}$	
sestof@dina.kvl.dk hastic grammars and RNA rutshell rimary sequence structure is described by hidden Markov models lidden Markov models are stochastic regular grammars context free grammars generalize regular grammars context free grammars generalize regular grammars iecondary sequence structure (of RNA) is described by stochastic context free grammars rammars, derivation, parsing, and the Chomsky hierarchy	$\begin{array}{ c c c c c c c c c c c c c c c c c c c$	 Regular grammars and finite state automata
sestof@dina.kvl.dk rutshell rimary sequence structure is described by hidden Markov models lidden Markov models are stochastic regular grammars ontext free grammars generalize regular grammars secondary sequence structure (of RNA) is described by stochastic context free grammars	$\begin{array}{ c c c c c } \mbox{Context sensitive grammars} & \alpha_1 W \alpha_2 - & & \\ \mbox{Unrestricted grammars} & & \alpha_1 W \alpha_2 - & \\ \mbox{α}, \beta \mbox{ etc. stand for arbitrary sequences of terminals and r} \end{array}$	 Grammars, derivation, parsing, and the Chomsky hierarchy
s described by hidden Markov models ochastic regular grammars "alize regular grammars "e (of RNA) is described by stochastic context free grammars	$\begin{array}{ c c c c } \mbox{Context sensitive grammars} & \alpha_1 W \alpha_2 - & & \\ \mbox{Unrestricted grammars} & & \alpha_1 W \alpha_2 - & \\ \mbox{α}, \beta \mbox{ etc. stand for arbitrary sequences of terminals and r} \end{array}$	Plan
s described by hidden Markov models ochastic regular grammars alize regular grammars	$\begin{array}{c c} \text{Context sensitive grammars} & \alpha_1 W \alpha_2 - \\ \\ \text{Unrestricted grammars} & \alpha_1 W \alpha_2 - \\ \end{array}$	 Secondary sequence structure (of RNA) is described by stochastic context free grammars
s described by hidden Markov models ochastic regular grammars alize regular grammars	mars $\alpha_1 W \alpha_2 - \alpha_1 W \alpha_2 - \alpha_1 W \alpha_2 - \alpha_1 W \alpha_2 - \alpha_2 W \alpha_2 W \alpha_2 - \alpha_2 W \alpha_2 W \alpha_2 - \alpha_2 W \alpha_2 W \alpha_2 W \alpha_2 W \alpha_2 - \alpha_2 W \alpha_2 - \alpha_2 W \alpha_2 - \alpha_2 W \alpha_2 W \alpha_2 W \alpha_2 - \alpha_2 W \alpha_2 W \alpha_2 W \alpha_2 - \alpha_2 W \alpha_2 W$	
s described by hidden Markov models ochastic regular grammars	$\alpha_1 W \alpha_2 -$	 Context free grammars generalize regular grammars
s described by hidden Markov models		 Hidden Markov models are stochastic regular grammars
sestoft@dina.kv.dk	W -	 Frimary sequence structure is described by nidden Markov models
sestoft@dina.kv.dk	W	
sestoft@dina.kvi.dk	Admis	In a nutshell
	By restricting the possible form of grammar rules, we get a hierarchy of increasingly powerful grammar classes:	Stochastic grammars and RNA
		sestoft

Page 3

KVL Seminar on computational biology 1999-12-20 Page 6	Expressive power in the Chomsky hierarchy Let $\alpha, \beta \in (N \cup T)^*$ denote arbitrary sequences of nonterminals and terminals. • Context sensitive grammars can describe copy languages, such as: $\{\alpha \alpha \mid \alpha \in (N \cup T)^*\}$ Context free grammars can describe palindrome languages (where α^{-1} is α reversed), such as: $\{\alpha \alpha^{-1} \mid \alpha \in (N \cup T)^*\}$ (Context free grammars can describe proper parenthetical nesting, as used in programming languages). Regular grammars can describe languages with uncorrelated repeats, such as: $\{a^n \alpha a^m \mid \alpha \in (N \cup T)^*\}$ But they cannot encode the additional requirement $n = m$.	Regular grammars and regular expressions It is customary to use regular expressions as a shorthand for regular grammars. Every regular expression corresponds to a regular grammar and vice versa. The FMR-1 grammar can be written as the regular expression: $gcgcgg((a + c)gg)^*cdg$ In uNix grep or emacs notation, that is: $gcgcgg([ac]gg)^*cdg$ More regular expressions: PROSITE patterns Regular expressions are used to describe 'signature' conserved protein sequences and their variants. Brackets [RK] indicate choice, braces {EDRKHPCG} choice from the complement, x matches anything. (Figure 9.3) More regular expression are used to describe 'signature' conserved protein sequences and their variants. Brackets [RK] indicate choice, braces {EDRKHPCG} choice from the complement, x matches anything. (Figure 9.3)
KVL Seminar on computational biology 1999-12-20 Page 8	 Derivation trees and parsing The derivation from a context free grammar may be shown as a tree. (The derivation from a regular grammar is a degenerate – linear – tree, not very interesting). Parsing: find a derivation tree for a given sequence, if any Given a grammar <i>G</i> and a sequence <i>α</i>. is sequence <i>α</i> derivable from <i>G</i>? if so, what derivation trees would produce sequence <i>α</i>? 	A context free grammar for palindromes over { a, b } $S \rightarrow aSa \mid bSb \mid aa \mid bb$ $S \rightarrow aSa = aaSaa = aabSbaa \Rightarrow aabaabaa$ Context free grammars and RNA secondary structure RNA sequences can form 'stem loops' when one part of the sequence matches another part ($a \leftrightarrow u, c \leftrightarrow g$). Three-base RNA stem loops' when one part of the sequence matches another part ($a \leftrightarrow u, c \leftrightarrow g$). Three-base RNA stem loops with 'nead' gaaa or gcaa can be described by: $S \rightarrow aW_1u \mid cW_1g \mid gW_1c \mid uW_1a$ $W_1 \rightarrow aW_2u \mid cW_2g \mid gW_2c \mid uW_2a$ $W_2 \rightarrow aW_3u \mid cW_3g \mid gW_3c \mid uW_3a$ $W_3 \rightarrow gaaa \mid gcaa$ $W_3 \rightarrow gaaa \mid gcaa$

Seminar on computational biology 1999-12-20

The the length of the sequence being parsed.	eing parsed.		-			Regular	Simple ('deterministic') PROSITE patterns	Stochastic Primary structure, probabilistically (HMM)
Grammar class	Recognizer	Time complexity	Space complexity			Context free	RNA secondary structure	RNA secondary structure, probabilistically
Regular grammars	Finite state automata	Linear	Constant					
Context free grammars	Pushdown automata	$O(n^3)$	$O(n^3)$		Chomsky	Chomsky normal form		
Context sensitive grammars	Linear bounded automata	NP-complete	PSPACE-complete		A grammar is	त्रr is on Chomsky r	normal form if all rules have fo	on Chomsky normal form if all rules have form $W \longrightarrow W_1 W_2$ or $W \longrightarrow a.$
Unrestricted grammars	Turing machines	Undecidable	Unbounded		Every cor	ntext free grammar	Every context free grammar can be transformed to Chomsky normal form.	nsky normal form.
	Seminar on computational biology 1999-12-20	99-12-20		Page 9	KVL		Seminar on computational biology 1999-12-20	al biology 1999-12-20
Stochastic regular grammars					Finding t Dynamic A probabi	he most probable programming (tabu ilistic version of the stochastic context	Finding the most probable parse tree: the CYK algorithm Dynamic programming (tabulation). Analogous to the Viterbi: A probabilistic version of the Cocke-Younger-Kasami (CYK) a Input: A stochastic context free grammar <i>G</i> on Chomsky nor	Finding the most probable parse tree: the CYK algorithm Dynamic programming (tabulation). Analogous to the Viterbi algorithm, which finds the most probable alignment. A probabilistic version of the Cocke-Younger-Kasami (CYK) algorithm for context free grammar parsing (ca. 1968). Input: A stochastic context free grammar G on Chomsky normal form, and a sequence $x = x_1$ r.
For each non-terminal symbol w , assign probabilit. Think of W as a state and a as an emitted symbol.					Finding t Dynamic A probabi Input: A Let the g	he most probable programming (tabu listic version of the listic context stochastic context 'ammar <i>G</i> have te	; parse tree: the CYK algori Jation). Analogous to the Vitt Cocke-Younger-Kasami (CY Cocke-Younger-Kasami (CY Tree grammar G on Chomsky free grammar $T = \{V_1,$	Finding the most probable parse tree: the CYK algorithm Dynamic programming (tabulation). Analogous to the Viterbi algorithm, which finds the most probable a A probabilistic version of the Cocke-Younger-Kasami (CYK) algorithm for context free grammar parsing A probabilistic context free grammar G on Chomsky normal form, and a sequence $x = x_{1L}$. Input: A stochastic context free grammar G on Chomsky normal form, and a sequence $x = x_{1L}$. Let the grammar G have terminal symbols $T = \{V_1, \ldots, V_M\}$ and start symbol $S = V_1$.
The stochastic grammar emits symbols on state transitions. A hidden Markov model (HMM) emits symbols without state transitions, and emits nothing on state transitions.	Stochastic regular grammars For each non-terminal symbol W , assign probabilities to rules of form $W \longrightarrow a W_1$ or W – Think of W as a state and a as an emitted symbol.	orm $W \longrightarrow a W_1$	or $W \longrightarrow a.$		Finding t Dynamic A probabi Input: A : Let the gr Let $e_v(a)$ Let $t_v(y)$	he most probable programming (tab listic version of the stochastic context ammar <i>G</i> have te) be <i>p</i> if $V_v \longrightarrow c$	Finding the most probable parse tree: the CYK algorithm Dynamic programming (tabulation). Analogous to the Viterbi algorithm, A probabilistic version of the Cocke-Younger-Kasami (CYK) algorithm to A probabilistic context free grammar <i>G</i> on Chomsky normal form, Input: A stochastic context free grammar <i>G</i> on Chomsky normal form, Let the grammar <i>G</i> have terminal symbols $T = \{V_1, \ldots, V_M\}$ and s Let $e_v(a)$ be <i>p</i> if $V_v \rightarrow a$ with probability <i>p</i> , and 0 otherwise. Let $t_v(y, z)$ be <i>p</i> if $V_v \rightarrow V_y V_z$ with probability <i>p</i> , and 0 otherwise	ithm erbi algorithm, which finds the most probable a 'K) algorithm for context free grammar parsing y normal form, and a sequence $x = x_{1L}$. , V_M and start symbol $S = V_1$. herwise. nd 0 otherwise.
The two kinds of machines are interconvertible (by introducing extra states and transitions). Thus discount of a biogenium on a using on UMM is preside of previous a using a state.	sign probabilities to rules of fe nitted symbol. Is on state transitions.	orm $W \longrightarrow a W_1$ ions, and emits not	or $W \longrightarrow a.$ hing on state transitio	φ. 	Finding t Dynamic A probabi Input: A : Let the gr Let $t_v(a)$ Output: \prime Algorith	he most probable programming (tab listic version of the stochastic context ammar <i>G</i> have ter) be <i>p</i> if $V_v \longrightarrow c$) be <i>p</i> if $V_v \longrightarrow c$ 2) be <i>p</i> if V_v table γ such that A table γ such that	; parse tree: the CYK algori Jation). Analogous to the Vite Cocke-Younger-Kasami (CY Cocke-Younger-Kasami (C) cocke-Younger-Kasami (C) cocke-Yo	Finding the most probable parse tree: the CYK algorithm Dynamic programming (tabulation). Analogous to the Viterbi algorithm, which finds the most probable alignment. A probabilistic version of the Cocke-Younger-Kasami (CYK) algorithm for context free grammar parsing (ca. 1968). Input: A stochastic context free grammar <i>G</i> on Chomsky normal form, and a sequence $x = x_{1L}$. Input: A stochastic context free grammar <i>G</i> on Chomsky normal form, and a sequence $x = x_{1L}$. Let the grammar <i>G</i> have terminal symbols $T = \{V_1, \ldots, V_M\}$ and start symbol $S = V_1$. Let $t_v(a)$ be p if $V_v \longrightarrow a$ with probability p , and 0 otherwise. Let $t_v(y, z)$ be p if $V_v \longrightarrow V_y V_z$ with probability p , and 0 otherwise. Let $t_v(x, z)$ be p if $V_v \longrightarrow V_v(i, j, v)$ is the probability of the most probable parse tree that derives x_{ij} from V_v . Algorithm: Put $\gamma(i, i, v) = e_v(x_i)$ for $i = 1L$ and $v = 1M$.
	sign probabilities to rules of to nitted symbol. Is on state transitions. symbols without state transit	orm $W \longrightarrow a W_1$ ions, and emits not	or $W \longrightarrow a.$ hing on state transitio	<u>~</u>	Finding the Dynamic proj A probabilisti Input: A stoc Let the gram Let $e_v(a)$ be Let $t_v(y, z)$ Output: A tal Algorithm: F	he most probable programming (tab listic version of the stochastic context ammar <i>G</i> have ter) be <i>p</i> if $V_v \longrightarrow c$) be <i>p</i> if $V_v \longrightarrow c$ 2) be <i>p</i> if V_v that A table γ such that A table γ such that I $(L - 1)$ and <i>j</i>	Finding the most probable parse tree: the CYK algorithm Dynamic programming (tabulation). Analogous to the Viterbi algori A probabilistic version of the Cocke-Younger-Kasami (CYK) algoritt Input: A stochastic context free grammar <i>G</i> on Chomsky normal for Let the grammar <i>G</i> have terminal symbols $T = \{V_1, \ldots, V_M\}$ a Let $e_v(a)$ be <i>p</i> if $V_v \longrightarrow a$ with probability <i>p</i> , and 0 otherwise. Let $t_v(y, z)$ be <i>p</i> if $V_v \longrightarrow V_y V_z$ with probability <i>p</i> , and 0 other Let $t_v(y, z)$ be <i>p</i> if $V_v \longrightarrow V_y V_z$ with probability <i>p</i> , and 0 otherwise. Let $t_v(x, z)$ be <i>p</i> if $V_v \longrightarrow V_y V_z$ with probability <i>p</i> , and 0 otherwise. Couput: A table γ such that $\gamma(i, j, v)$ is the probability of the most Algorithm: Put $\gamma(i, i, v) = e_v(x_i)$ for $i = 1L$ and $v = 1M$, put	ithm erbi algorithm, which finds the most probable a 'K) algorithm for context free grammar parsing 'y normal form, and a sequence $x = x_{1L}$. , V_M } and start symbol $S = V_1$. herwise. herwise. nd 0 otherwise. of the most probable parse tree that derives x_i . = 1 M .
c	Sochastic regular grammars For each non-terminal symbol W , assign probabilities to rules of form $W \longrightarrow aW_1$ or $W \longrightarrow a$. Think of W as a state and a as an emitted symbol. The stochastic grammar emits symbols on state transitions. A hidden Markov model (HMM) emits symbols without state transitions, and emits nothing on state transitions. The two kinds of machines are interconvertible (by introducing extra states and transitions). The two kinds of machines are interconvertible (by introducing extra states and transitions).	orm $W\longrightarrow aW_1$ ions, and emits not a states and transif	or $W \longrightarrow a$. hing on state transitio ions). stochastic regular gra	imar.	Finding t Dynamic A probabi Input: A : Let the gr Let $t_v(y)$. Cutput: ι Algorithr For $i = 1$	he most probable programming (tab listic version of the stochastic context ammar <i>G</i> have ter) be <i>p</i> if $V_v \longrightarrow c$ 2) be <i>p</i> if $V_v \longrightarrow c$ 2) be <i>p</i> if V_v and <i>i</i> 1(<i>L</i> - 1) and <i>j</i> 1(<i>L</i> - 1) and <i>j</i>	* parse tree: the CYK algori Jation). Analogous to the Vite Cocke-Younger-Kasami (CY Cocke-Younger-Kasami (CY minal symbols $T = \{V_1,, white weights (CY), white weights (CY), with probability p, and 0 othV_y V_z with probability p, x = i \cdot \gamma(i, j, v) is the probability p= e_v(x_i) for i = 1 \dots L and v = i \dots L= (i + 1) \dots L and v = 1 \dots Ly_{x^2} \sum_{k=i}^{j-1} (\gamma(i, k))$	able parse tree: the CYK algorithm (tabulation). Analogous to the Viterbi algorithm, which finds the most probable a f the Cocke-Younger-Kasami (CYK) algorithm for context free grammar parsing text free grammar <i>G</i> on Chomsky normal form, and a sequence $x = x_{1L}$. e terminal symbols $T = \{V_1, \ldots, V_M\}$ and start symbol $S = V_1$. $\rightarrow a$ with probability p , and 0 otherwise. $\rightarrow V_y V_z$ with probability p , and 0 otherwise. that $\gamma(i, j, v)$ is the probability of the most probable parse tree that derives x_i . $v) = e_v(x_i)$ for $i = 1L$ and $v = 1M$. d j = (i + 1)L and $v = 1M$, put $\gamma(i, j, v) = \max_{y,z} \lim_{k=i}^{j-1} (\gamma(i, k, y) \cdot \gamma(k + 1, j, z) \cdot t_v(y, z))$
c	sign probabilities to rules of t nitted symbol. symbols without state transit nvertible (by introducing extr nvertible (by introducing extr	orm $W \longrightarrow a W_1$ ions, and emits not a states and transif aquence $lpha$ using a	or $W \longrightarrow a.$ hing on state transitio ions). stochastic regular gra	nmar.	Finding t Dynamic A probabi Input: A : Let the gr Let $e_v(a)$ Let $t_v(y)$ Algorithn For $i = 1$	he most probable programming (tablistic version of the listic version of the stochastic context ammar <i>G</i> have ten) be <i>p</i> if $V_v \longrightarrow c$) be <i>p</i> if $V_v \longrightarrow c$ 1 be <i>p</i> if $V_v = -i$ 1 be <i>i</i> if $V_v = -i$ 1 be <i>p</i> if $V_v = -i$ 1 be <i>i</i> if $V_v = -i$	Finding the most probable parse tree: the CYK algorithm Dynamic programming (tabulation). Analogous to the Viterbi algorithm, wh A probabilistic version of the Cocke-Younger-Kasami (CYK) algorithm for α A probabilistic context free grammar G on Chomsky normal form, an Input: A stochastic context free grammar G on Chomsky normal form, an Let the grammar G have terminal symbols $T = \{V_1, \ldots, V_M\}$ and star Let $e_v(a)$ be p if $V_v \longrightarrow a$ with probability p , and 0 othenwise. Let $t_v(y, z)$ be p if $V_v \longrightarrow V_y V_z$ with probability p , and 0 otherwise. Let $t_v(y, z)$ be p if $V_v \longrightarrow V_y V_z$ with probability p , and 0 otherwise. Coutput: A table γ such that $\gamma(i, j, v)$ is the probability of the most probat Algorithm: Put $\gamma(i, i, v) = e_v(x_i)$ for $i = 1L$ and $v = 1M$. For $i = 1(L - 1)$ and $j = (i + 1)L$ and $v = 1M$, put $\gamma(i, j, v) = \max_{y,z} \max_{k=i}^{j-1} (\gamma(i, k, y) \cdot \gamma(k + 1, j))$. The probability of the most probable derivation of x from G is $\gamma(1, L, 1)$.	Ithm erbi algorithm, which finds the most probable a 'K) algorithm for context free grammar parsing ' normal form, and a sequence $x = x_{1L}$. . , V_{M} } and start symbol $S = V_1$. herwise. nd 0 otherwise. nd 0 otherwise. if the most probable parse tree that derives x_i . = 1M. f, put f, put $x, y) \cdot \gamma (k + 1, j, z) \cdot t_v(y, z))$ G is $\gamma(1, L, 1)$.
	sign probabilities to rules of t nitted symbol. symbols without state transit nvertible (by introducing extr nvertible (by introducing of se	orm $W\longrightarrow aW_1$ ions, and emits not a states and transit equence $lpha$ using a	or $W \longrightarrow a.$ hing on state transitio .ions). stochastic regular gra	imar.	Finding t Dynamic A probabi Input: A : Let the gr Let $e_v(a)$ Let $t_v(y)$. Dutput: / Algorithr For $i = 1$ The prob	he most probable programming (tablistic version of the listic version of the stochastic context ammar G have ter) be p if $V_v \longrightarrow c$) be p if $V_v \longrightarrow c$ table γ such that λ table γ such that n : Put $\gamma(i, i, v) =$ $\dots (L - 1)$ and j $\dots (L - 1)$ and j $\dots (L - 1)$ and j	Finding the most probable parse tree: the CYK algorithm Dynamic programming (tabulation). Analogous to the Viterbi. A probabilistic version of the Cocke-Younger-Kasami (CYK) a Input: A stochastic context free grammar <i>G</i> on Chomsky nor Let the grammar <i>G</i> have terminal symbols $T = \{V_1, \ldots, V$ Let $e_v(a)$ be p if $V_v \longrightarrow a$ with probability p , and 0 otherw Let $t_v(y, z)$ be p if $V_v \longrightarrow v_y V_z$ with probability p , and 0 otherw Let $t_v(y, z)$ be p if $V_v \longrightarrow v_y V_z$ with probability p , and 0 otherw Let $t_v(y, z)$ be p if $V_v \longrightarrow v_y V_z$ with probability of the Algorithm: Put $\gamma(i, i, v) = e_v(x_i)$ for $i = 1L$ and $v = 1$ For $i = 1(L - 1)$ and $j = (i + 1)L$ and $v = 1M$, pu $\gamma(i, j, v) = \max_{y, z} \max_{k=i}^{j-1} (\gamma(i, k, y))$ The probability of the most probable derivation of x from <i>G</i> is Also, a traceback $\tau(i, j, v)$ is built as in the Viterbi algorithm	Ithm erbi algorithm, which finds the most probable a 'Y) algorithm for context free grammar parsing Y normal form, and a sequence $x = x_{1L}$. . V_{M} and start symbol $S = V_1$. herwise. nd 0 otherwise. nd 0 otherwise. if the most probable parse tree that derives x_i = 1 M . I, put I, put $i, y) \cdot \gamma (k + 1, j, z) \cdot t_v(y, z))$ G is $\gamma(1, L, 1)$. ithm.

Ş

Grammar classes and 'parsers' (recognizers)

For each grammar class there is a characteristic 'machine type' that solves the parsing problem for that class.

Stochastic context free grammars

For each non-terminal symbol W, assign a distribution to the set of rules $W \longrightarrow \alpha.$

More expressive grammar classes require more powerful recognizers (and more time and space).

Grammar class	Recognizer	Time complexity	Time complexity Space complexity
Regular grammars	Finite state automata	Linear	Constant
Context free grammars	Pushdown automata	$O(n^3)$	$O(n^3)$
Context sensitive grammars	Linear bounded automata NP-complete	NP-complete	PSPACE-complete
Unrestricted grammars	Turing machines	Undecidable	Unbounded

Ř