Dynamic Programming for Lazy Bastards

Peter F. Stadler

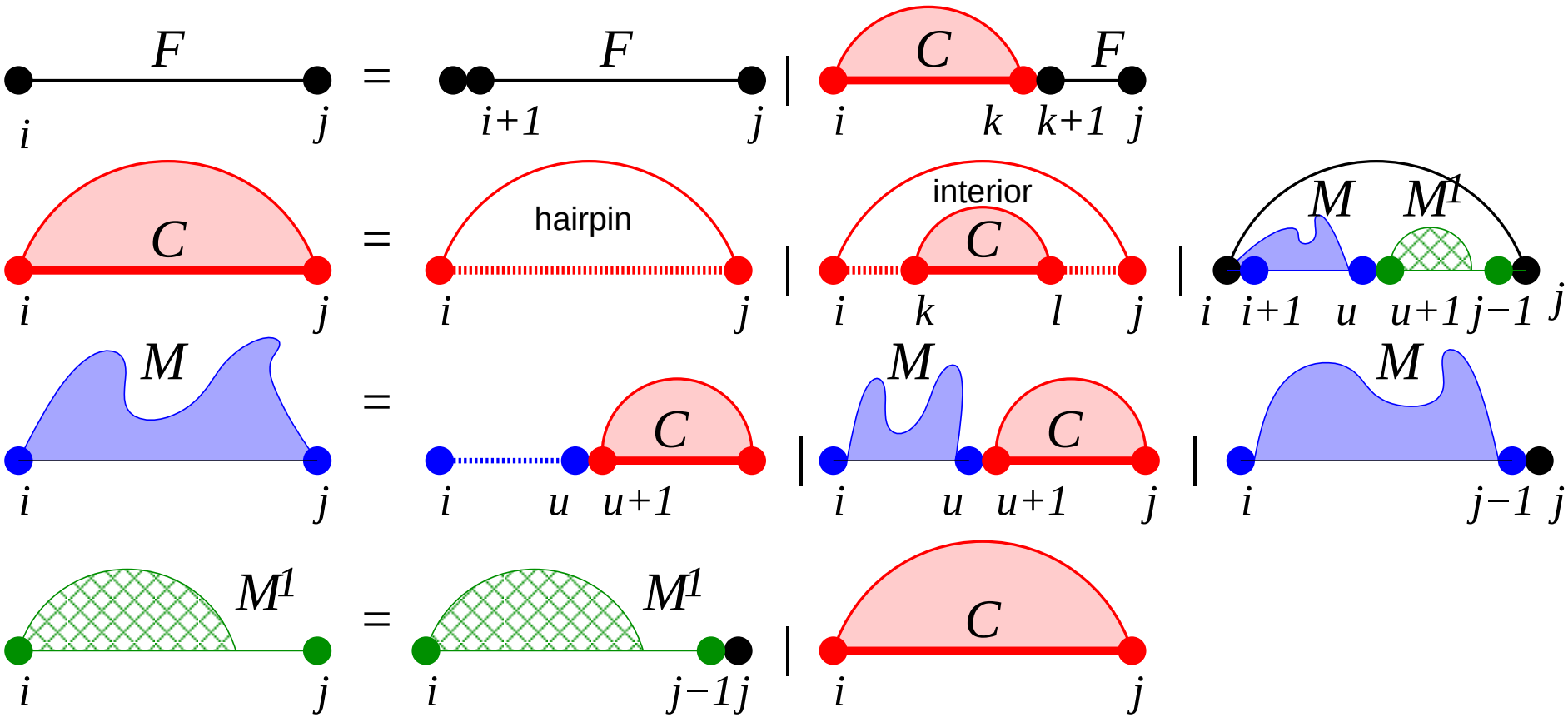
Bioinformatics Group, Dept. of Computer Science &
Interdisciplinary Center for Bioinformatics,
University of Leipzig
Max Planck Institute for Mathematics in the Sciences
RNomics Group, Fraunhofer Institute for Cell Therapy and Immunology
Institute for Theoretical Chemistry, Univ. of Vienna (external faculty)
Center for non-coding RNA in Technology and Health, U. Copenhagen
The Santa Fe Institute (external faculty)

Vienna, Jul 08 2015

Joint work with ...

Christian Höner zu Siderdissen

RNA Folding by Dynamic Programming



$F \to xF|CF \quad C \to h|iCi'|xMM^1x' \quad M \to uC|MC|Mx \quad M^1 \to M^1x|C$

Dynamic Programming

"Dynamic Programming is when you come up with a real-life problem for breakfast, have a formal description at lunch, and a running program that solves it by dinner time."

Robert Giegerich at a Turrach seminar of the Vienna Theoretical Chemistry group in the mid 1990s.

ADP for MCFGs

joint work (to be published **very** soon) with Maik Riechert and Johannes Waldmann (HTWK Leipzig)

- MCFG are a generalization of CFGs
- original definition by Seki has the (for us annoying) feature of emitting terminals from rewrite functions rather than the productions themselves (as in usual frameworks for context sensitive grammars) We use a weakly equivalent formulation (same languages, other parse trees)

MCFGs

Consider the palindromes language $\mathcal{L} = \{\mathtt{a}^i\mathtt{b}^j\mathtt{a}^i\mathtt{b}^j \mid i,j \geq 0\}$ of overlapping palindromes of the form $\mathtt{a}^i\mathtt{b}^j\mathtt{a}^i$ and $\mathtt{b}^j\mathtt{a}^i\mathtt{b}^j$, with $\mathtt{b}^j$ and $\mathtt{a}^i$ missing, respectively. It cannot be expressed by a CFG. The MCFG $\mathcal{G} = (\{Z, A, B\}, \{\mathtt{a}, \mathtt{b}\}, Z, R, P)$ with $\mathcal{L}(\mathcal{G}) = \mathcal{L}$ serves as a running example:

$$\dim(Z) = 1 \qquad \dim(A) = \dim(B) = 2$$

$$Z \to \mathrm{f}_Z[A, B]$$

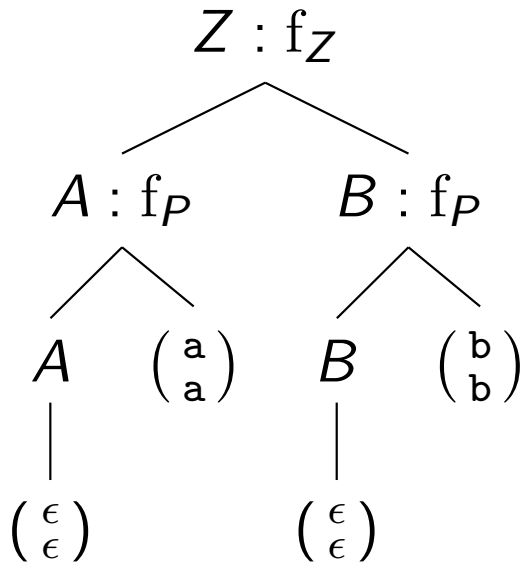
$$\mathrm{f}_Z[\begin{pmatrix} A_1 \\ A_2 \end{pmatrix}, \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}] = A_1 B_1 A_2 B_2$$

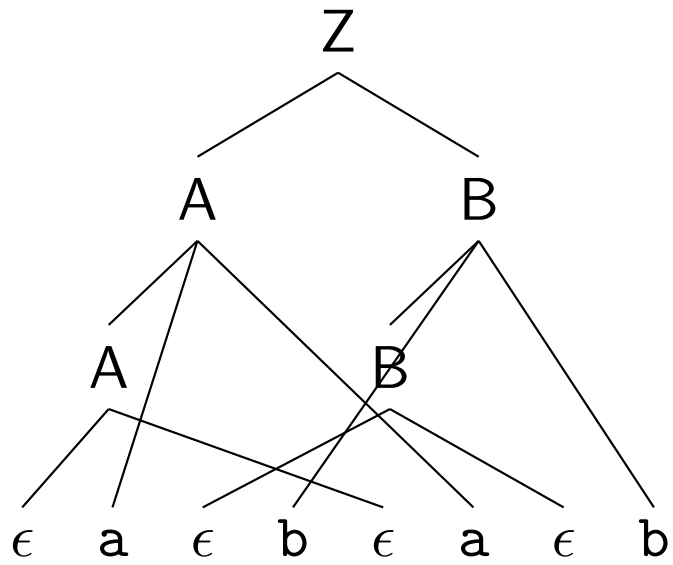
$$A \to \mathrm{f}_P[A, \begin{pmatrix} \mathtt{a} \\ \mathtt{a} \end{pmatrix}] \mid \begin{pmatrix} \epsilon \\ \epsilon \end{pmatrix}$$

$$\mathrm{f}_P[\begin{pmatrix} A_1 \\ A_2 \end{pmatrix}, \begin{pmatrix} c \\ d \end{pmatrix}] = \begin{pmatrix} A_1 c \\ A_2 d \end{pmatrix}$$

$$B \to \mathrm{f}_P[B, \begin{pmatrix} \mathtt{b} \\ \mathtt{b} \end{pmatrix}] \mid \begin{pmatrix} \epsilon \\ \epsilon \end{pmatrix}$$

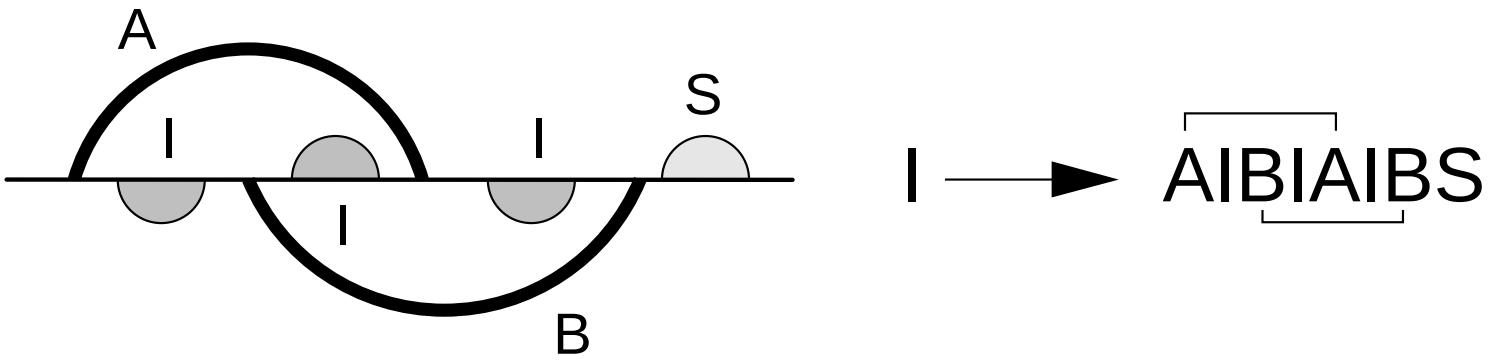
Visualization: Parse trees with crossings ...





ADP for MCFGs

why bother? Pseudoknot folding



and linguistics applications

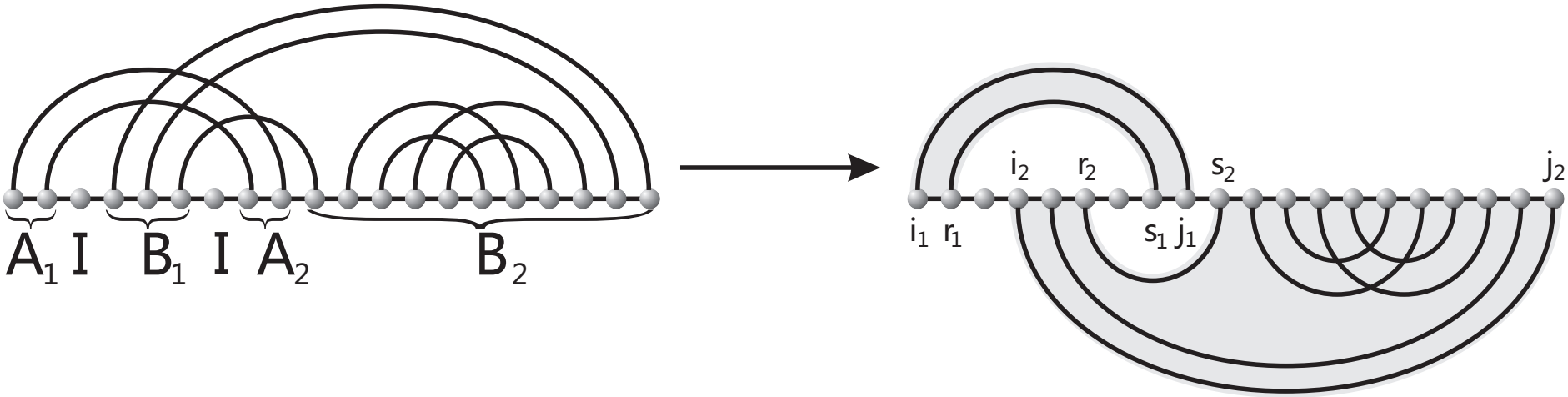
das mer d'chind em Hans es huus lönd hälfe aastriiche

that we the children Hans the house let help paint

Two-Dimensional Nonterminals

two dimensional nonterminals ...

... are in essence the "gap matrices" of Rivas&Eddy



MCFG Rule $I \rightarrow IA_1IB_1IA_2IB_2S$

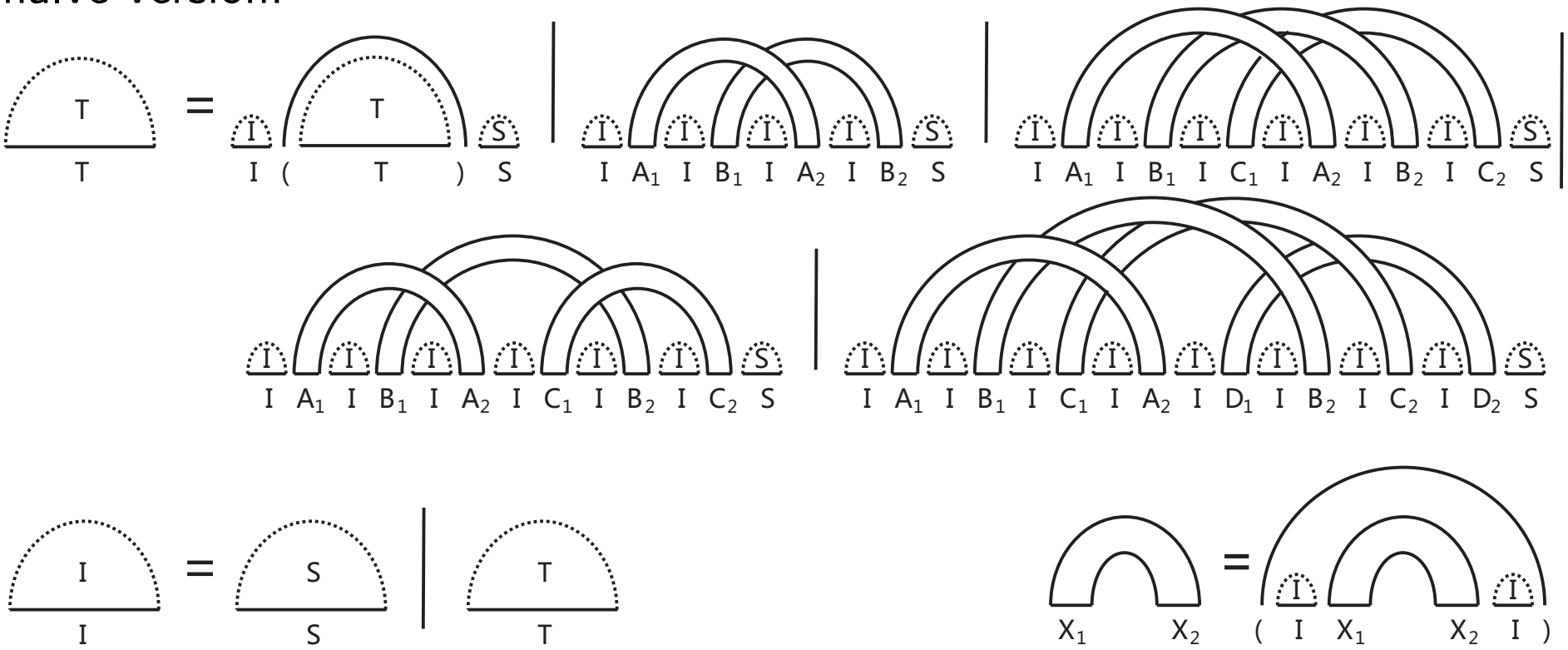
induces the fragment-pairs $[i_1, r_1]$, $[s_1, j_1]$ and $[i_2, r_2]$, $[s_2, j_2]$.

Arcs connecting the two fragments of a pair are non-crossing, while arcs with both endpoints within the same fragment may be crossing

... such as those within $[s_2, j_2]$.

The `gfold` MCFG

naïve version:



Naïve Algorithm: MCFG Form

... rephrased in more traditional form:

$$
\begin{aligned}
I &\rightarrow S \mid T \\
S &\rightarrow (S)S \mid \ :S \mid \epsilon \\
T &\rightarrow I(T)S \\
T &\rightarrow IA_1IB_1IA_2IB_2S \\
T &\rightarrow IA_1IB_1IA_2IC_1IB_2IC_2S \\
T &\rightarrow IA_1IB_1IC_1IA_2IB_2IC_2S \\
T &\rightarrow IA_1IB_1IC_1IA_2ID_1IB_2IC_2ID_2S \\
\vec{X} &\rightarrow [(xIX_1, X_2I)x] \mid [(x,)x]\,,
\end{aligned}
$$

where $X \in \{A, B, C, D\}$ distinguishes the four types of pseudoknots.
... not useful in this form
$\Longrightarrow$ $O(n^{18})$ time and $O(n^4)$ space algorithm

More efficiency

An $O(n^6)$ and $O(n^4)$ space algorithm is obtainable by tabulating intermediate results, i.e., introducing additional non-terminals in the MCFG

$$\begin{aligned}
\vec{U} &\rightarrow [IX_1, IX_2] \\
\vec{V} &\rightarrow [U_1 U_1', U_2 U_2'] \\
\vec{W} &\rightarrow [U_1, U_1' U_2 U_2'] \mid [V_1, U_1 V_2 U_2]
\end{aligned}$$

where (U_1', U_2') is a marked copy of (U_1, U_2) used to identify the components which must later be expanded in a coupled way.

$$\begin{aligned}
T &\rightarrow I(T)S \mid I'S \\
I' &\rightarrow V_1 V_2 \mid U_1 V_1 U_2 V_2 \mid U_1 W_1 U_2 W_2
\end{aligned}$$

Bellmann's Principle

Definition

An evaluation algebra $\mathcal{E}_E$ satisfies Bellman's Principle if every function symbol $f \in F_E$ with arity k satisfies for all sets z_i of $\mathcal{E}_E$-parses the following two axioms:

(B1) $\mathrm{h}_E(z_1 \uplus z_2) = \mathrm{h}_E(\mathrm{h}_E(z_1) \uplus \mathrm{h}_E(z_2))$.

(B2) $\mathrm{h}_E[f(x_1, \ldots, x_k) \mid x_1 \in z_1, \ldots, x_k \in z_k]$
$= \mathrm{h}_E[f(x_1, \ldots, x_k) \mid x_1 \in \mathrm{h}_E(z_1), \ldots, x_k \in \mathrm{h}_E(z_k)]$.

Theorem

An MCF-ADP instance can be solved in polynomial time if its evaluations algebra satisfied Bellmann's principle and is "polynomially bounded" (i.e., every individual production can be evaluated in polynomial time).

A Toy Example for Pseudoknots

Generic DSL to specify ADP-MCFGs

$$
\begin{aligned}
S &\to (S)S \mid .S \mid \epsilon \\
S &\to U_1 V_1 U_2 V_2 \\
U_{12} &\to \left(\begin{smallmatrix} S \\ - \end{smallmatrix}\right)\left(\begin{smallmatrix} (\\ - \end{smallmatrix}\right) U_{12} \left(\begin{smallmatrix} - \\ S \end{smallmatrix}\right)\left(\begin{smallmatrix} - \\ ) \end{smallmatrix}\right) \\
V_{12} &\to \left(\begin{smallmatrix} S \\ - \end{smallmatrix}\right)\left(\begin{smallmatrix} (\\ - \end{smallmatrix}\right) V_{12} \left(\begin{smallmatrix} - \\ S \end{smallmatrix}\right)\left(\begin{smallmatrix} - \\ ) \end{smallmatrix}\right)
\end{aligned}
$$

Implementation: GenussFold

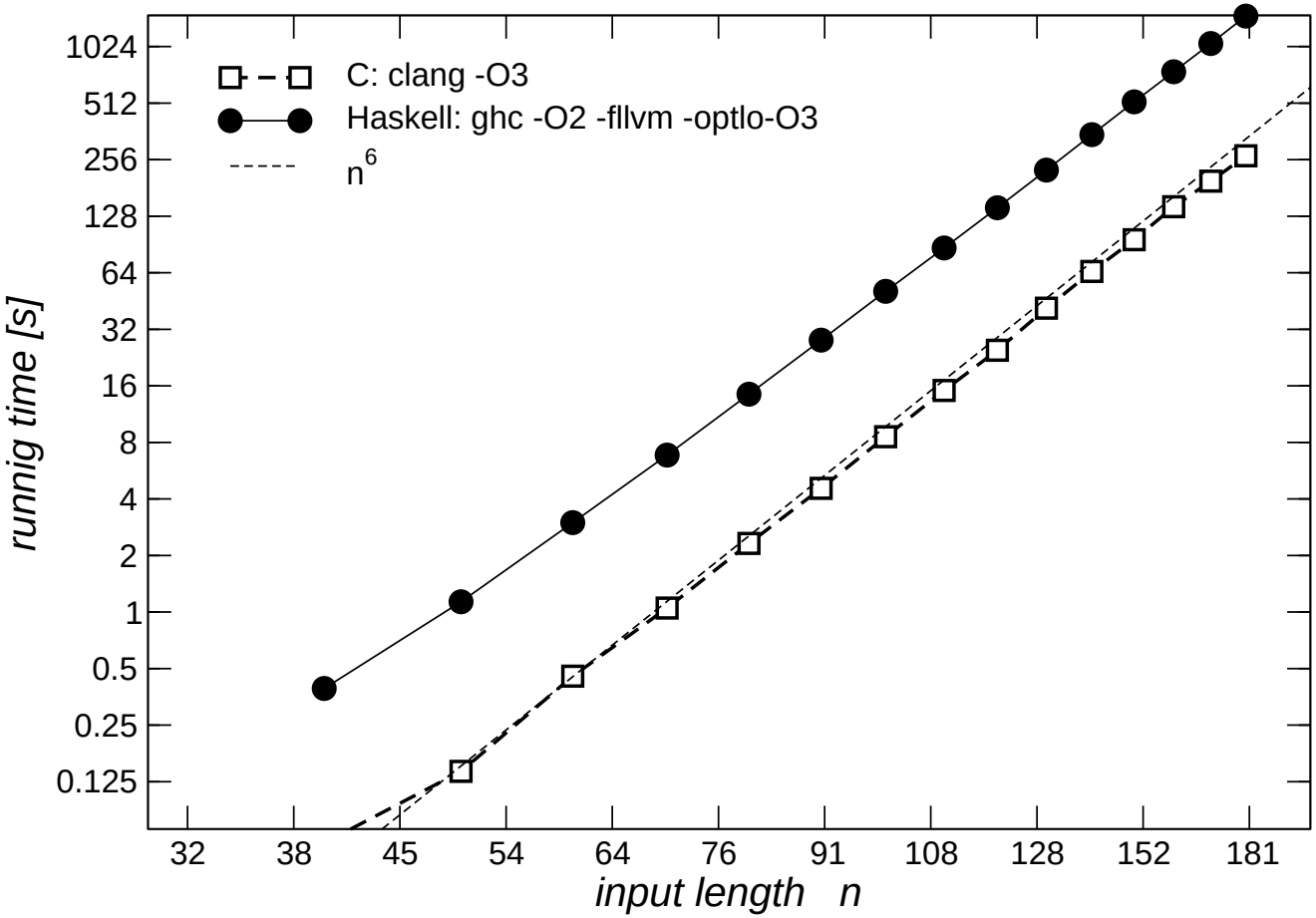
```
[formalLanguage|
Verbose

Grammar: PKN
N: S
{-
 - <U,2> is a split non-terminal.
 -
 - We explicitly introduce <U> and <V> as we want to have @pk1@ and @pk2@
 - in place. In principle, we could make use of an intermediate recursive
 - syntactic variable to ease the memory load, but this is simpler.
 -}
N: <U,2>
N: <V,2>
T: c
S: S
S -> unp <<< S c
S -> jux <<< S c S c
S -> nil <<< e
S -> pse <<< U V U V

<U,U> -> pk1 <<< [S,-] [c,-] <U,U> [-,S] [-,c]
<U,U> -> nll <<< [e,e]

<V,V> -> pk2 <<< [S,-] [c,-] <V,V> [-,S] [-,c]
<V,V> -> nll <<< [e,e]
//
Emit: PKN
|]
```

Implementation: `GenussFold`

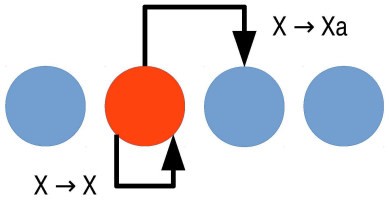


Products of DP Algorithms

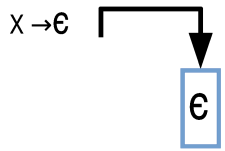
... can we avoid sweating when constructing complex grammars?

Atomic One-Tape Grammars

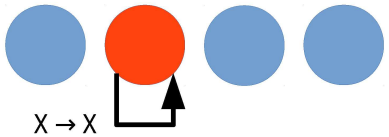
$$\mathcal{S} = (\{X\}, \{a\}, \{X \to Xa \mid X\}, X)$$



$$\mathcal{N} = (\{X\}, \{\varepsilon\}, \{X \to \varepsilon\}, X)$$



$$\mathcal{L} = (\{X\}, \{\}, \{X \to X\}, X)$$



$$\mathcal{S} + \mathcal{N} - \mathcal{L} = (\{X\}, \{a, \varepsilon\}, \{X \to Xa \mid \varepsilon\}, X)$$

Alignment via Multiplication

$$\mathcal{NW} = \mathcal{S} \otimes \mathcal{S} + \mathcal{N} * 2 - \mathcal{L} * 2$$

	G	L	O	B	A	L
L	-1	0	-5	-7	-9	-8
O	-3	-2	2	0	-2	-4
C	-5	-4	0	1	-1	-3
A	-7	-6	-2	-1	3	-2
L	-9	-5	-4	-3	-2	5

G	L	O	B	A	L
--	L	O	C	A	L

$$\binom{X}{Y} \rightarrow \binom{X}{Y}\binom{a}{a} \mid \binom{X}{Y}\binom{a}{\epsilon}$$
$$\mid \binom{X}{Y}\binom{\epsilon}{a} \mid \binom{\varepsilon}{\varepsilon}$$

Alignment via Multiplication

$$\mathcal{NW} = \mathcal{S} \otimes \mathcal{S} + \mathcal{N} * 2 - \mathcal{L} * 2$$
$$= (X \to Xa|X) \otimes (X \to Xa|X)$$

	G	L	O	B	A	L
L	-1	0	-5	-7	-9	-8
O	-3	-2	2	0	-2	-4
C	-5	-4	0	1	-1	-3
A	-7	-6	-2	-1	3	-2
L	-9	-5	-4	-3	-2	5

G	L	O	B	A	L
--	L	O	C	A	L

$$\binom{X}{Y} \to \binom{X}{Y}\binom{a}{a} \mid \binom{X}{Y}\binom{a}{\epsilon}$$
$$\mid \binom{X}{Y}\binom{\epsilon}{a} \mid \binom{\varepsilon}{\varepsilon}$$

Alignment via Multiplication

$$\mathcal{NW} = \mathcal{S} \otimes \mathcal{S} + \mathcal{N} * 2 - \mathcal{L} * 2$$
$$= (X \to Xa|X) \otimes (X \to Xa|X)$$

	G	L	O	B	A	L
L	-1	0	-5	-7	-9	-8
O	-3	-2	2	0	-2	-4
C	-5	-4	0	1	-1	-3
A	-7	-6	-2	-1	3	-2
L	-9	-5	-4	-3	-2	5

G	L	O	B	A	L
--	L	O	C	A	L

$$\binom{X}{Y} \to \binom{X}{Y}\binom{a}{a} \mid \binom{X}{Y}\binom{a}{\epsilon}$$
$$\mid \binom{X}{Y}\binom{\epsilon}{a} \mid \binom{\varepsilon}{\varepsilon}$$

$$\binom{X}{X} \to \binom{X}{X}\binom{a}{a}$$
$$\mid \binom{X}{X}\binom{a}{\epsilon}$$
$$\mid \binom{X}{X}\binom{\epsilon}{a}$$
$$\mid \binom{X}{X}$$

Alignment via Multiplication

	G	L	O	B	A	L
L	-1	0	-5	-7	-9	-8
O	-3	-2	2	0	-2	-4
C	-5	-4	0	1	-1	-3
A	-7	-6	-2	-1	3	-2
L	-9	-5	-4	-3	-2	5

G	L	O	B	A	L
--	L	O	C	A	L

$$\binom{X}{Y} \rightarrow \binom{X}{Y}\binom{a}{a} \mid \binom{X}{Y}\binom{a}{\epsilon} \mid \binom{X}{Y}\binom{\epsilon}{a} \mid \binom{\epsilon}{\epsilon}$$

$$\begin{aligned}\mathcal{NW} =& \mathcal{S} \otimes \mathcal{S} + \mathcal{N} * 2 - \mathcal{L} * 2 \\ =& (X \rightarrow Xa|X) \otimes (X \rightarrow Xa|X) \\ &+ (X \rightarrow \epsilon) * 2\end{aligned}$$

$$\binom{X}{X} \rightarrow \binom{X}{X}\binom{a}{a} \mid \binom{X}{X}\binom{a}{\epsilon} \mid \binom{X}{X}\binom{\epsilon}{a} \mid \binom{X}{X}$$

Alignment via Multiplication

$$\begin{aligned}\mathcal{NW} =& \mathcal{S} \otimes \mathcal{S} + \mathcal{N} * 2 - \mathcal{L} * 2 \\ =& (X \to Xa|X) \otimes (X \to Xa|X) \\ &+ (X \to \epsilon) * 2\end{aligned}$$

	G	L	O	B	A	L
L	-1	0	-5	-7	-9	-8
O	-3	-2	2	0	-2	-4
C	-5	-4	0	1	-1	-3
A	-7	-6	-2	-1	3	-2
L	-9	-5	-4	-3	-2	5

G	L	O	B	A	L
--	L	O	C	A	L

$$\begin{pmatrix} X \\ Y \end{pmatrix} \to \begin{pmatrix} X \\ Y \end{pmatrix}\begin{pmatrix} a \\ a \end{pmatrix} \mid \begin{pmatrix} X \\ Y \end{pmatrix}\begin{pmatrix} a \\ \epsilon \end{pmatrix} \mid \begin{pmatrix} X \\ Y \end{pmatrix}\begin{pmatrix} \epsilon \\ a \end{pmatrix} \mid \begin{pmatrix} \varepsilon \\ \varepsilon \end{pmatrix}$$

$$\begin{pmatrix} X \\ X \end{pmatrix} \to \begin{pmatrix} X \\ X \end{pmatrix}\begin{pmatrix} a \\ a \end{pmatrix} \mid \begin{pmatrix} X \\ X \end{pmatrix}\begin{pmatrix} a \\ \epsilon \end{pmatrix} \mid \begin{pmatrix} X \\ X \end{pmatrix}\begin{pmatrix} \epsilon \\ a \end{pmatrix} \mid \begin{pmatrix} X \\ X \end{pmatrix} \mid \begin{pmatrix} \varepsilon \\ \varepsilon \end{pmatrix}$$

Alignment via Multiplication

	G	L	O	B	A	L
L	-1	0	-5	-7	-9	-8
O	-3	-2	2	0	-2	-4
C	-5	-4	0	1	-1	-3
A	-7	-6	-2	-1	3	-2
L	-9	-5	-4	-3	-2	5

G	L	O	B	A	L
--	L	O	C	A	L

$$\binom{X}{Y} \rightarrow \binom{X}{Y}\binom{a}{a} \mid \binom{X}{Y}\binom{a}{\epsilon} \mid \binom{X}{Y}\binom{\epsilon}{a} \mid \binom{\varepsilon}{\varepsilon}$$

$$\mathcal{NW} = \mathcal{S} \otimes \mathcal{S} + \mathcal{N} * 2 - \mathcal{L} * 2$$
$$= (X \rightarrow Xa|X) \otimes (X \rightarrow Xa|X)$$
$$+ (X \rightarrow \epsilon) * 2$$
$$- (X \rightarrow X) * 2$$

$$\binom{X}{X} \rightarrow \binom{X}{X}\binom{a}{a} \mid \binom{X}{X}\binom{a}{\epsilon} \mid \binom{X}{X}\binom{\epsilon}{a} \mid \binom{X}{X} \mid \binom{\varepsilon}{\varepsilon}$$

Alignment via Multiplication

	G	L	O	B	A	L
L	-1	0	-5	-7	-9	-8
O	-3	-2	2	0	-2	-4
C	-5	-4	0	1	-1	-3
A	-7	-6	-2	-1	3	-2
L	-9	-5	-4	-3	-2	5

G	L	O	B	A	L
--	L	O	C	A	L

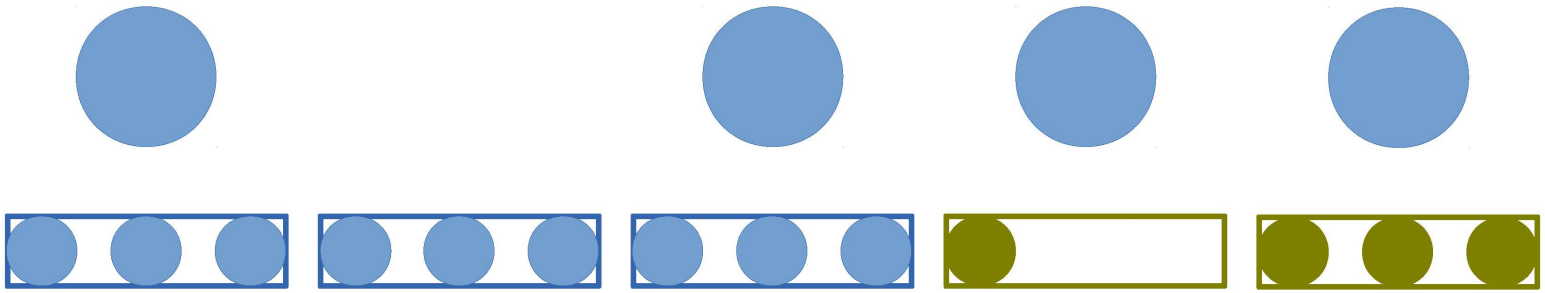
$$\binom{X}{Y} \rightarrow \binom{X}{Y}\binom{a}{a} \mid \binom{X}{Y}\binom{a}{\epsilon}$$
$$\mid \binom{X}{Y}\binom{\epsilon}{a} \mid \binom{\varepsilon}{\varepsilon}$$

$$\mathcal{NW} = \mathcal{S} \otimes \mathcal{S} + \mathcal{N} * 2 - \mathcal{L} * 2$$
$$= (X \rightarrow Xa|X) \otimes (X \rightarrow Xa|X)$$
$$+ (X \rightarrow \epsilon) * 2$$
$$- (X \rightarrow X) * 2$$

$$\binom{X}{X} \rightarrow \binom{X}{X}\binom{a}{a}$$
$$\mid \binom{X}{X}\binom{a}{\epsilon}$$
$$\mid \binom{X}{X}\binom{\epsilon}{a}$$

$$\mid \binom{\varepsilon}{\varepsilon}$$

Showcase Application: DNA–Protein Alignment



- 3 reading frames
- read 0–3 nucleotides (change rf & align locally)
- read 0-1 amino acids (align globally)
- Allow insertion/deletion editing
- 34 production rules

Practical application: C-insertion editing in *Physarum polycephalum*
Or:

Robert-Giegerich-style DP for reproducing original work by Ralf Bundschuh and collaborators

DNA–Protein Alignment

```
Grammar: DL
R    -> skp <<< R    c
R    -> lcl <<< F{i}
F{i} -> lcl <<< L
L    -> skp <<< L    c
//
Grammar: D
F{i} -> stay <<< F{i}   c c c
F{i} -> rf1  <<< F{i+1} c c
F{i} -> rf2  <<< F{i+2} c
F{i} -> del  <<< F{i}
//
Grammar: Dd
F{i} -> nil <<< empty
//
Product: DnaPro
D >< P + DL >< Ps + Dd >< Pd - Ds >< Ps
//
```

```
Grammar: Ds
F{i} -> del <<< F{i}
//
Grammar: P
P -> amino <<< P a
P -> del   <<< P
//
Grammar: Pd
P -> nil <<< empty
//
Grammar: Ps
P -> del <<< P
//
```

```
DNA: gi|11466223|ref|NC_002508.1| Physarum polycephalum mitochondrion, complete genome @ Forward 17247
Protein: lcl|KC353356.1_cdsid_AGH24310.1 [gene=nad5] [protein=NADH dehydrogenase subunit 5] [protein_id=AGH24310.1] [locatio
DNA length: 4020 Protein length: 670
1 Nt shifts: 35   |||   2 Nt shifts: 0
Score: 582   Length-adjusted: 0.87

17248 AATAAATTAAAATGTTTTCATGTTTCCTTAATAGCATTTATAATTCTTTTTATGTTAGGTAGACATCTGGGAAGCAAATTGTCTCGGCTT 17337
    1 M  Y  L  L  I  V  F  L  P  L  L  G  S  I  T  A  G  F  F  G  R  S  L  G  K  Q  G  A  A  I      30

17338 TGCTATTACAAT---GTCATTTTATCACTTATAATTGTTTATATTATTTTATTCATGTTTTTTTTATGGTCAAATATAAGCTTT---AAT 17421
   31 I  T  T  S  C  V  A  L  S  S  L  F  S  M  V  A  F  Y  E  V  G  L  C  G  S  P  C  Y  I  R      60

17422 TTAGGTTCTTGGGTTTCTGTAGGTACTTTAGATATTAC-TACAAATTTATAATG---ATCCTTTATCCATTACTTTGGTACATTAATTCC 17507
   61 L  F  N  W  I  D  S  E  M  L  H  A  S  W  G  F  L  F  D  S  L  T  V  V  M  L  I  V  V  T      90

17508 TTTATTACTTTATTAATCTA-ATTTATTCTTATGA-TATTTACATGAAGATCCTAATTAGTTAAATTTTTTG---CTTATTTAGTTTTTT 17592
   91 I  V  S  S  L  V  H  L  Y  S  V  G  Y  M  S  H  D  P  H  L  P  R  F  M  S  Y  L  S  L  F     120

17593 CTCTTTTTCTATGTTTGTCTTGTTTTTGCTGGTAATTACTT-ATTATGTTTTTAGGATGGGAAGC-TGTGGATTAGCTTCTTATTT-CTT 17679
  121 T  F  F  M  L  M  L  V  T  G  D  N  F  V  Q  M  F  L  G  W  E  G  V  G  L  C  S  Y  L  L     150

17680 ATTAATTTCTGGAT-ACAAGAAATCAAGCAAT-CAGTCTGCTATTAAAGCAATTATTTTTAATCGTGA-GGTGATGCAGCTTTC---ATA 17763
  151 I  N  F  W  F  T  R  L  Q  A  N  K  S  A  I  K  A  M  I  M  N  R  I  G  D  F  G  L  S  L     180

17764 AGTGCTATGGGTCTTATTTATTATTTTTTAATTCTTTGATTTAGAAGATTTAGAATTACTTGTTCACAATATGAACATACTACTTTTAGC 17853
  181 G  M  M  A  I  F  F  I  F  K  S  V  D  F  I  T  V  F  A  L  S  P  Y  M  T  D  A  T  I  V     210

17854 TTTTTTCATATTCTTTTCAACAATTGA---ATGATAGCTCTTTTCCTATTTTTGCTGCTGTGCTAAATCAGCACAATCTTT---TTACAT 17937
  211 F  L  N  Y  E  V  H  A  L  T  L  I  C  I  L  L  F  V  G  A  V  G  K  S  S  Q  L  G  L  H     240

17938 CCTTGGTTACCTGATGCTATGAA-GGACCTACACCAGTTTCAGCATTATTACATTCTGCTACATG-GTAACAGCAGGTGTTTC-TTAATA 18024
  241 T  W  L  P  D  A  M  E  G  P  T  P  V  S  A  L  I  H  A  A  T  M  V  T  A  G  V  F  L  I     270

18025 TTAAGATC-TCTGTTATTTTCTCACATGCTCCTTATATTTC-ATTATTGTAGCTTGTATTGGCCTAAT-ACAGCTAATATTTCTTCTTTA 18111
  271 A  R  C  S  P  I  F  E  Y  A  P  T  A  L  L  V  V  T  I  V  G  A  M  T  A  F  F  A  A  T     300

18112 ACAGGTTA-TTACAATATGACATAAAACGTATTAT-GCATTTTCAACCTGTAGCCAACTTGGTTT-ATGATGTTTGCTAT-GGTATTGGT 18197
  301 T  G  L  L  Q  N  D  I  K  R  V  I  A  Y  S  T  C  S  Q  L  G  Y  M  V  F  A  C  G  I  S     330

18198 AATTATACTTT-GCTTTATTTCATTTAGTAAATAT-GCTTTCTTTAAAGCACTCTTATTTTATGT-GCCGGATCCGTTATCCACGCTACC 18284
  331 G  Y  S  V  G  M  F  H  L  M  N  H  A  F  F  K  A  L  L  F  L  S  A  G  C  V  I  H  A  L     360

18285 GGC---CATCAGGATATTCGGCGTATGGGAGT-TTATTTAAAGA-TTACCCATAACTTATGTTGCAATGCTCTT-GCTTCTTTATCTTTA 18368
  361 A  D  E  Q  D  M  R  R  M  G  G  I  V  K  I  V  P  F  T  Y  G  M  M  L  I  G  S  M  S  L     390

18369 ATTGGTTTTCT-TTCTTAAGTGGTTTTTATAG-AAGGATTTTCTTCTAGAAGCTACTTACAATATATTTGGTATGTCT---CTTATGTTA 18453
  391 M  G  F  P  F  L  T  G  F  Y  S  K  D  V  I  L  E  L  A  F  A  K  Y  T  I  D  G  T  F  A     420
```

Work in Progress: General CFGs

Nussinov algorithm for RNA folding:

$$S \to \$ \mid Sa \mid SB \qquad B \to aS\hat{a}$$

Sankoff's algorithm for joint alignment and folding:

$$\begin{pmatrix} S \\ S \end{pmatrix} \to \begin{pmatrix} \$ \\ \$ \end{pmatrix} \mid \begin{pmatrix} S \\ S \end{pmatrix}\begin{pmatrix} a \\ a \end{pmatrix} \mid \begin{pmatrix} S \\ S \end{pmatrix}\begin{pmatrix} a \\ \varepsilon \end{pmatrix} \mid \begin{pmatrix} S \\ S \end{pmatrix}\begin{pmatrix} \varepsilon \\ a \end{pmatrix}$$
$$\begin{pmatrix} S \\ S \end{pmatrix} \to \begin{pmatrix} S \\ S \end{pmatrix}\begin{pmatrix} B \\ B \end{pmatrix}$$
$$\begin{pmatrix} B \\ B \end{pmatrix} \to \begin{pmatrix} a \\ a \end{pmatrix}\begin{pmatrix} S \\ S \end{pmatrix}\begin{pmatrix} \hat{a} \\ \hat{a} \end{pmatrix}$$

$$\mathcal{SANK} = \mathcal{NW}(S) + (S \to SB,\, B \to aS\hat{a}) * 2 \equiv \mathcal{NW}(S) + \mathcal{NUS} * 2$$

Work in Progress: General CFGs

Difficulty 1: how to interpret terms like "$\binom{xB}{PQy}$" ???
Difficulty 2: a useful product must be associative
Promising Approach: consider Normal Forms for CFGs.
... we found an associative product for 2-Greibach Normal Forms, which however does not reduce to the natural product for linear grammars

see CHzS & PFS TCCB 2014 for details

DP for Set-Like Data Structures

Can we write algorithms like the dynamic programm for the TSP in ADP-style?

$$f([A,i]) = \min_{j \in A} f([A \setminus \{i\}, j]) + f(\langle j, i \rangle)$$

Analog of string (tree) parsing: recursive decomposition of the set A with "boundary" i

$$[A \setminus \{i\}, i] \mapsto [A \setminus \{i,j\}, j] \cup \langle j, i \rangle$$

Key objects: sets $A := [\text{int}(A), \partial A]$ with *boundary* or "interface" ∂A and *interior* $A \setminus \partial A$.

General Decompositions

$$[\mathrm{int}(A), \partial A] \mapsto \mathop{+\!\!+}_{i} [\mathrm{int}(A_i), \partial A_i]$$

with the following properties:

(C1) $\bigcup_i A_i = A$, i.e., the parts of A form a covering of A.

(C2) $\mathrm{int}(A_i) \cap \mathrm{int}(A_j) \neq \emptyset$ implies $i = j$, i.e., the interiors of the parts are disjoint.

(C3) $\mathrm{int}(A_i) \subseteq \mathrm{int}(A)$, i.e., the interiors behave like isotonic functions.

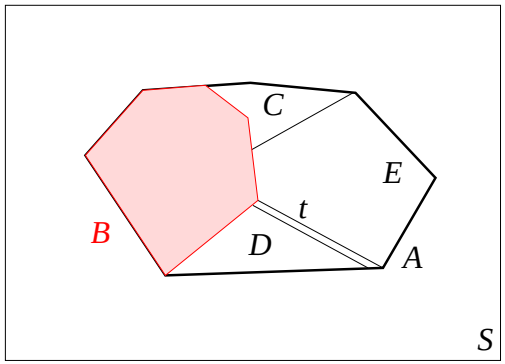
Free Outside Algorithms

Generic implicit definition of an outside object

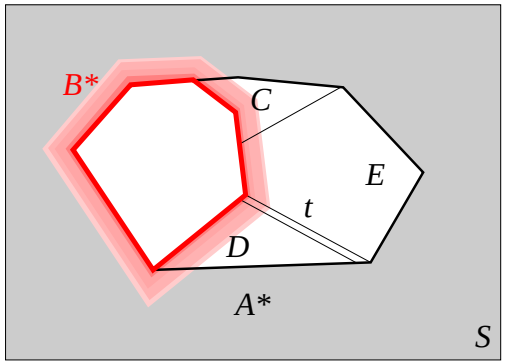
$$[\mathrm{int}(X), \partial A] \to [\mathrm{int}(A), \partial A] +\!\!+ [\mathrm{int}(A^*), \partial^* A^*]$$

with $\mathrm{int}(A^*) := X \setminus (\mathrm{int}(A) \cup \partial A)$ and $\partial A = \partial^* A^*$ (up to ordering information)

Inside



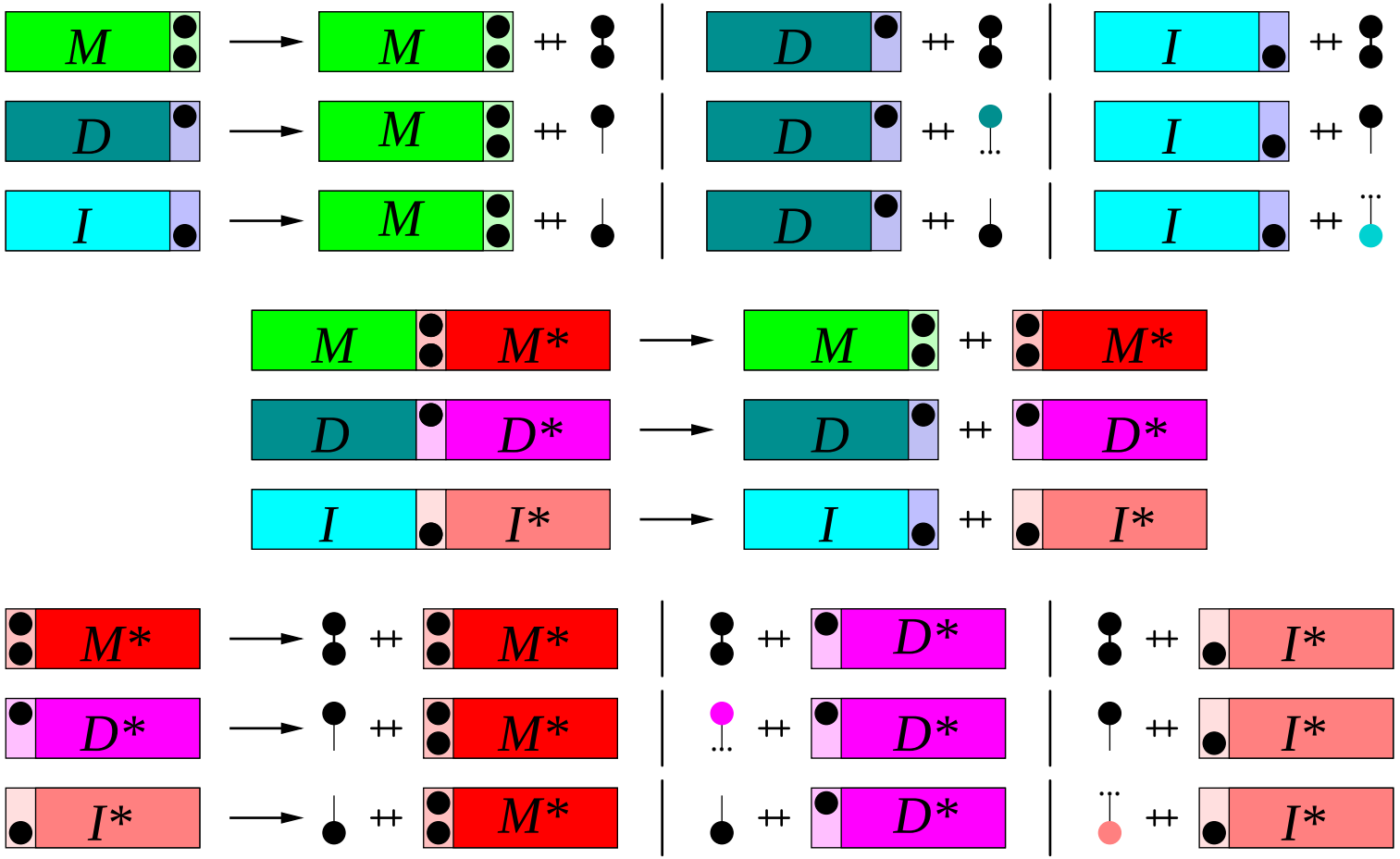
Outside



Inside $\quad A \mapsto \left(\mathop{+\!\!+}_{i} A_i \right) +\!\!+ \left(\mathop{+\!\!+}_{j} \langle t_j \rangle \right)$

Outside $\quad A_k^* \mapsto A^* +\!\!+ \left(\mathop{+\!\!+}_{i \neq k} A_i \right) +\!\!+ \left(\mathop{+\!\!+}_{j} \langle t_j \rangle \right)$

Example 1: Outside Alignment



Example 2: RNA folding

$$S \to B \mid \bullet S \mid BS \mid \bullet, \qquad B \to (S)$$

$$\text{“}S \to BS\text{”} := \{S_{ij} \mapsto B_{ik} S_{k+1,j} | 1 \le i \le k < j \le n\}$$
$$\text{“}B \to (S)\text{”} := B_{ij} \mapsto \langle i, j \rangle \mathbin{+\!\!+} S_{i+1,j-1}$$

$$\begin{aligned} \text{“}X \to \mathrm{S} \mathbin{+\!\!+} \mathrm{S}^*\text{”} &:= \{S_{ij} \mathbin{+\!\!+} T_{i-1,j+1} | 1 \le i \le j \le n\} \\ \text{“}X_{ij} \to \mathrm{B} \mathbin{+\!\!+} \mathrm{B}^*\text{”} &:= \left\{B_{ij} \mathbin{+\!\!+} B^*_{ij} | 1 \le i \le j \le n\right\} \end{aligned} \tag{1}$$

Example 2: RNA folding

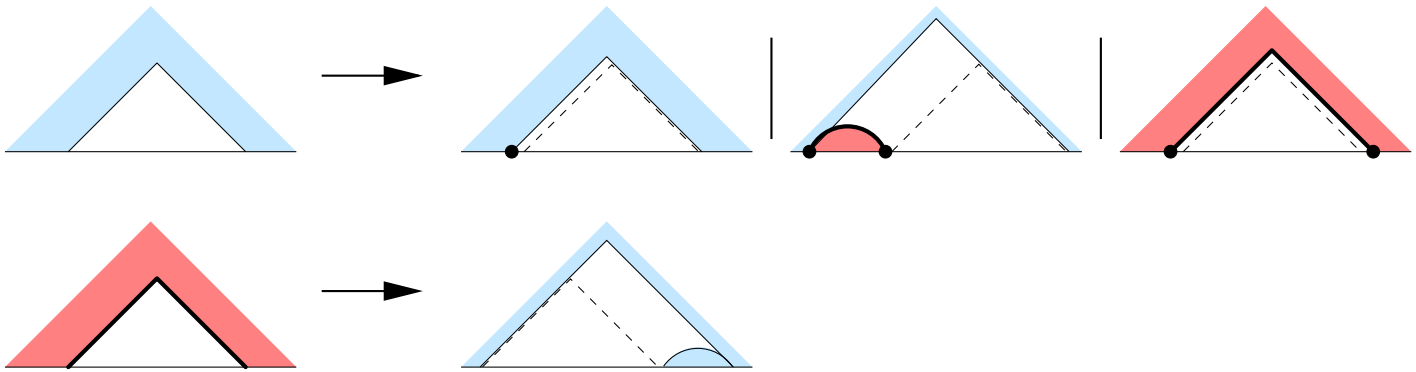
inside		outside
$S_{ij} \mapsto B_{ij}$	yields	$B^*_{ij} \mapsto S^*_{ij}$
$S_{ij} \mapsto \langle i \rangle \mathbin{+\!\!+} S_{i+1,j}$	yields	$S^*_{i+1,j} \mapsto \langle i \rangle \mathbin{+\!\!+} S^*_{ij}$
$S_{ij} \mapsto B_{ik} S_{k+1,j}$	yields	$B^*_{ik} \mapsto S^*_{ij} \mathbin{+\!\!+} S_{k+1,j}$
	and	$S^*_{k+1,j} \mapsto S^*_{ij} \mathbin{+\!\!+} B_{ik-1}$
$B_{ij} \mapsto S_{i+1,j-1} \mathbin{+\!\!+} \langle i, j \rangle$	yields	$S^*_{i+1,j-1} \mapsto B^*_{ij} \mathbin{+\!\!+} \langle i, j \rangle$

Example 2: RNA folding

... with $B^* \to C$ and $S^* \to T$ and renaming of indices
(all of which takes place implicitly)

$$T_{ij} \mapsto \langle i \rangle \mathbin{+\!\!+} T_{i-1,j}$$
$$T_{ij} \mapsto T_{k,j} \mathbin{+\!\!+} B_{k+1,i}$$
$$T_{ij} \mapsto C_{ij} \mathbin{+\!\!+} \langle i, j \rangle$$

$$C_{ij} \mapsto T_{i-1,j+1}$$
$$C_{ij} \mapsto T_{i-1,l+1} \mathbin{+\!\!+} S_{j+1,l}$$

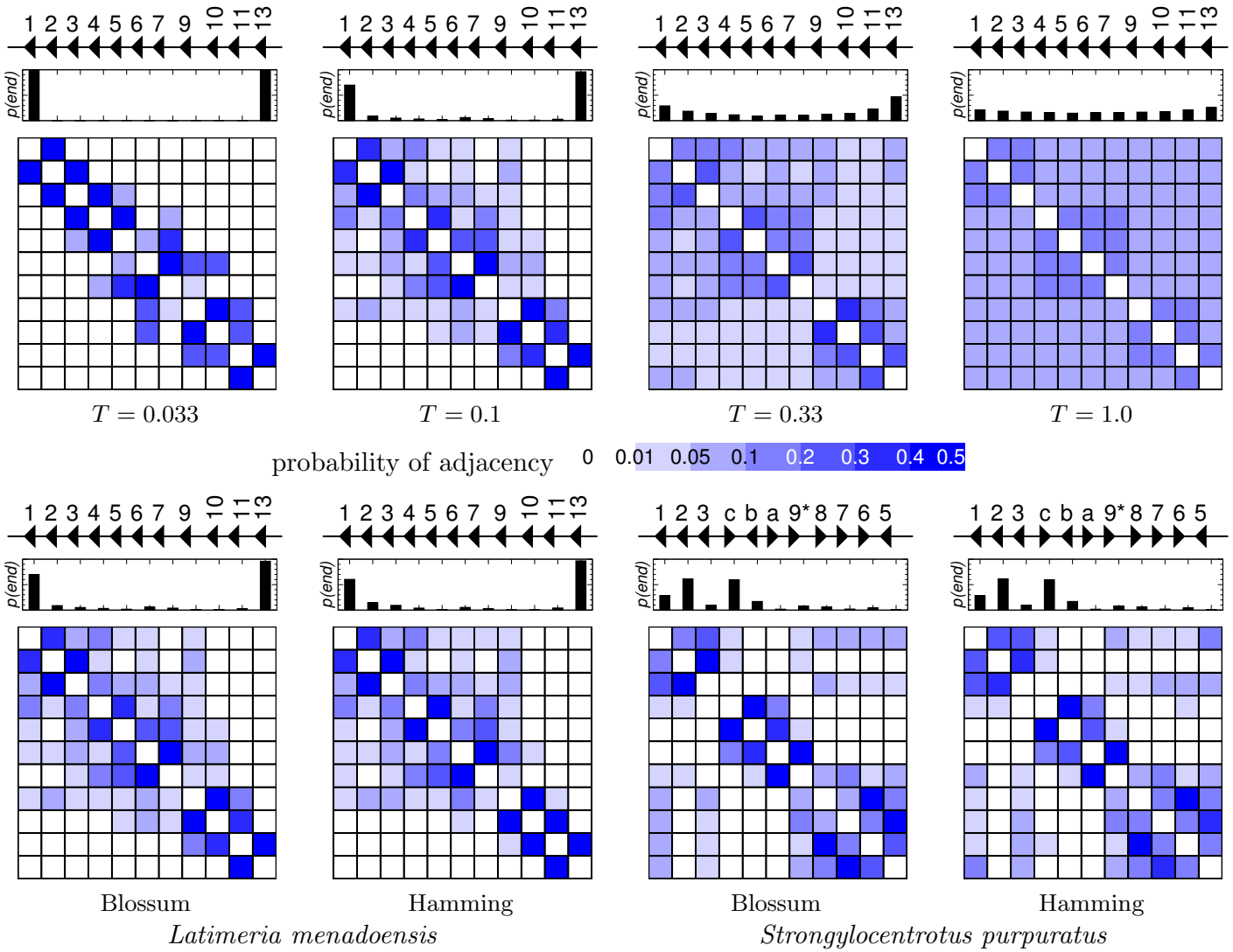


Trade-Off: Outside recursions are "mechnically" derivable **but** we need to specify what parsing means in terms of the decomposition rules for the forward algorithm.
... just once for a particular type of traversal, such as sets with 1 or 2 endpoints

An application to Gene Clusters

Did the HOX gene clusters evolve by local tandem duplications only?

Approach: compute probability of adjacencies along similarity-weighted Hamiltonian paths



Take Home Messages

- algebraic formulation of dynamic programming on strings simplied algorithm design A LOT
- products of grammars make life easier for complex multi-tape CFGs
- MCFGs can be handled in ADP ... makes use of the multi-tape machinery to handle rewrites
  (convenient DSL to write CFG-style MCFGs)
- generalization to other data-types is possible:
  requires redefinition of parsing
- ... shows that outside recursions are uniquely derivable
- practical applications e.g. on all Hamiltonian path problems
- we are building an infrastructure to support our own and our computers' lazyness

Thanks ...

- Christian Höner zu Siederdissen
- Christian Reidys, Fenix W.D. Huang (SDU Odense), Jørgen Andersen, Robert Penner (Århus), Markus Nebel (U KL)
- Maik Riechert, Johannes Waldmann (HTWK Leipzig)
- Ivo L. Hofacker, Christoph Flamm & the Vienna gang
- Sonja J. Prohaska