Algebraic Dynamic Programming for Multiple Context-Free Languages

Masterarbeit

Maik Riechert

Hochschule für Technik, Wirtschaft und Kultur Leipzig
Fakultät Informatik, Mathematik und Naturwissenschaften

Gutachter und Betreuer:
Prof. Dr. Johannes Waldmann, HTWK Leipzig
Prof. Dr. Peter F. Stadler, Universität Leipzig

12. Juli 2013

Motivation: RNA Sekundärstrukturvorhersage

Eingabe: Primärstruktur

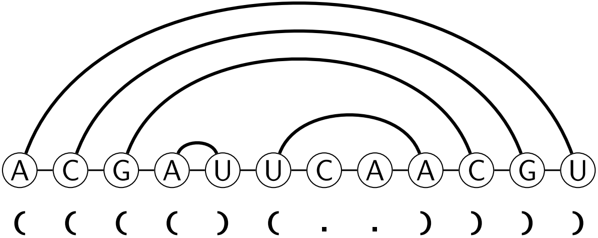
ACGAUUCAACGU

Motivation: RNA Sekundärstrukturvorhersage

Eingabe: Primärstruktur

ACGAUUCAACGU

Suchraum: Sekundärstrukturen (ohne Pseudoknoten)



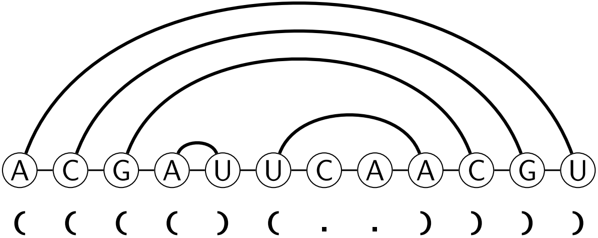
A-U
G-C
G-U

Motivation: RNA Sekundärstrukturvorhersage

Eingabe: Primärstruktur

ACGAUUCAACGU

Suchraum: Sekundärstrukturen (ohne Pseudoknoten)



A-U
G-C
G-U

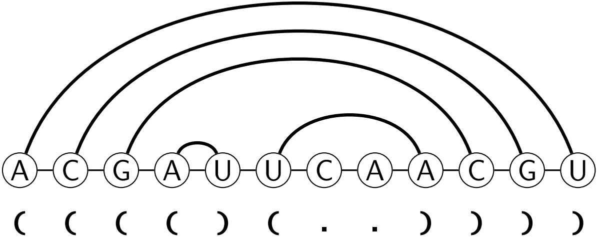
Bewertung: z.B. Anzahl Basenpaare

Motivation: RNA Sekundärstrukturvorhersage

Eingabe: Primärstruktur

ACGAUUCAACGU

Suchraum: Sekundärstrukturen (ohne Pseudoknoten)



A-U
G-C
G-U

Bewertung: z.B. Anzahl Basenpaare

Lösung: Sekundärstruktur mit „bester“ Bewertung

Suchraum = Ableitungsbäume (Searls, 1997)

Kontextfreie Grammatik (Chomsky, 1959)

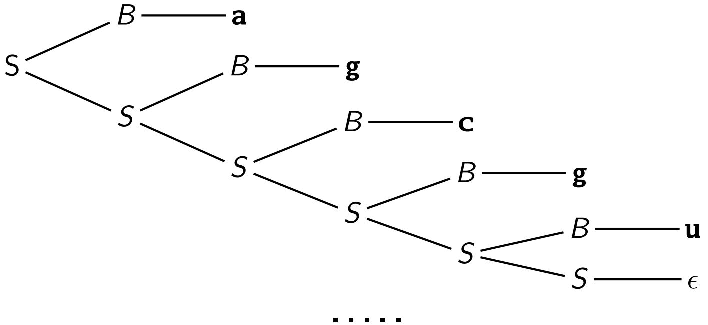
$$
\begin{aligned}
S &\rightarrow BS \mid PS \mid \epsilon \\
P &\rightarrow \mathbf{a}S\mathbf{u} \mid \mathbf{u}S\mathbf{a} \mid \mathbf{g}S\mathbf{c} \mid \mathbf{c}S\mathbf{g} \mid \mathbf{g}S\mathbf{u} \mid \mathbf{u}S\mathbf{g} \\
B &\rightarrow \mathbf{a} \mid \mathbf{u} \mid \mathbf{g} \mid \mathbf{c}
\end{aligned}
$$

Suchraum = Ableitungsbäume (Searls, 1997)

Kontextfreie Grammatik (Chomsky, 1959)

$$S \rightarrow BS \mid PS \mid \epsilon$$
$$P \rightarrow \mathbf{a}S\mathbf{u} \mid \mathbf{u}S\mathbf{a} \mid \mathbf{g}S\mathbf{c} \mid \mathbf{c}S\mathbf{g} \mid \mathbf{g}S\mathbf{u} \mid \mathbf{u}S\mathbf{g}$$
$$B \rightarrow \mathbf{a} \mid \mathbf{u} \mid \mathbf{g} \mid \mathbf{c}$$

Ableitungsbaum = kodierte Sekundärstruktur

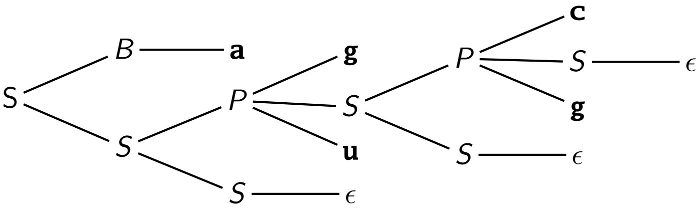


Suchraum = Ableitungsbäume (Searls, 1997)

Kontextfreie Grammatik (Chomsky, 1959)

$$S \rightarrow BS \mid PS \mid \epsilon$$
$$P \rightarrow \mathbf{a}S\mathbf{u} \mid \mathbf{u}S\mathbf{a} \mid \mathbf{g}S\mathbf{c} \mid \mathbf{c}S\mathbf{g} \mid \mathbf{g}S\mathbf{u} \mid \mathbf{u}S\mathbf{g}$$
$$B \rightarrow \mathbf{a} \mid \mathbf{u} \mid \mathbf{g} \mid \mathbf{c}$$

Ableitungsbaum = kodierte Sekundärstruktur



.(())

Algebraische Dynamische Programmierung (ADP)

$\approx$ Baumgrammatik + Algebra + Optimierungsziel (Giegerich, 2000)

$$
\begin{aligned}
S &\rightarrow \mathsf{f}_1(B, S) \mid \mathsf{f}_2(P, S) \mid \mathsf{f}_3() \\
P &\rightarrow \mathsf{f}_4(\mathbf{a}, S, \mathbf{u}) \mid \mathsf{f}_4(\mathbf{u}, S, \mathbf{a}) \mid \mathsf{f}_4(\mathbf{g}, S, \mathbf{c}) \mid \\
&\quad\ \mathsf{f}_4(\mathbf{c}, S, \mathbf{g}) \mid \mathsf{f}_4(\mathbf{g}, S, \mathbf{u}) \mid \mathsf{f}_4(\mathbf{u}, S, \mathbf{g}) \\
B &\rightarrow \mathsf{f}_5(\mathbf{a}) \mid \mathsf{f}_5(\mathbf{u}) \mid \mathsf{f}_5(\mathbf{g}) \mid \mathsf{f}_5(\mathbf{c})
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{f}_1(b, s) &= b + s \\
\mathsf{f}_2(p, s) &= p + s \\
\mathsf{f}_3() &= 0 \\
\mathsf{f}_4(b_1, s, b_2) &= 1 + s \\
\mathsf{f}_5(b) &= 0
\end{aligned}
$$

Algebraische Dynamische Programmierung (ADP)

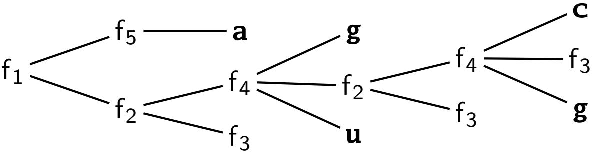
$\approx$ Baumgrammatik + Algebra + Optimierungsziel (Giegerich, 2000)

$$
\begin{aligned}
S &\rightarrow \mathsf{f}_1(B,S) \mid \mathsf{f}_2(P,S) \mid \mathsf{f}_3() \\
P &\rightarrow \mathsf{f}_4(\mathbf{a},S,\mathbf{u}) \mid \mathsf{f}_4(\mathbf{u},S,\mathbf{a}) \mid \mathsf{f}_4(\mathbf{g},S,\mathbf{c}) \mid \\
&\quad\ \mathsf{f}_4(\mathbf{c},S,\mathbf{g}) \mid \mathsf{f}_4(\mathbf{g},S,\mathbf{u}) \mid \mathsf{f}_4(\mathbf{u},S,\mathbf{g}) \\
B &\rightarrow \mathsf{f}_5(\mathbf{a}) \mid \mathsf{f}_5(\mathbf{u}) \mid \mathsf{f}_5(\mathbf{g}) \mid \mathsf{f}_5(\mathbf{c})
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{f}_1(b,s) &= b+s \\
\mathsf{f}_2(p,s) &= p+s \\
\mathsf{f}_3() &= 0 \\
\mathsf{f}_4(b_1,s,b_2) &= 1+s \\
\mathsf{f}_5(b) &= 0
\end{aligned}
$$

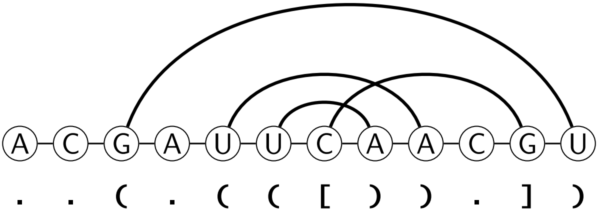
Evaluierung eines Terms = Bewertung

$$\mathsf{f}_1(\mathsf{f}_5(\mathbf{a}),\mathsf{f}_2(\mathsf{f}_4(\mathbf{g},\mathsf{f}_2(\mathsf{f}_4(\mathbf{c},\mathsf{f}_3(),\mathbf{g}),\mathsf{f}_3()),\mathbf{u}),\mathsf{f}_3())) = 2$$



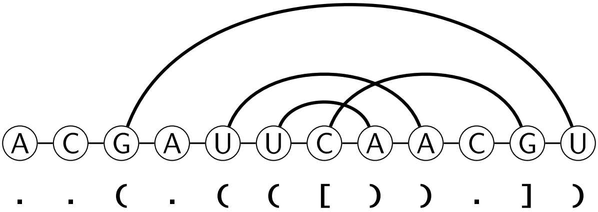
Erweiterung des Suchraums

Jetzt: RNA Sekundärstrukturen *mit* Pseudoknoten



Erweiterung des Suchraums

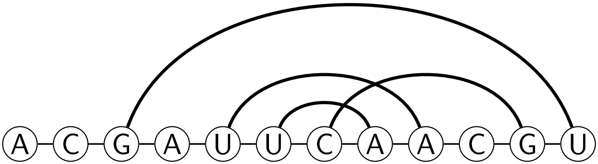
Jetzt: RNA Sekundärstrukturen *mit* Pseudoknoten



vereinfacht und verallgemeinert: $L = \mathbf{(}^i\mathbf{[}^j\mathbf{)}^i\mathbf{]}^j$ mit $i, j > 0$

Erweiterung des Suchraums

Jetzt: RNA Sekundärstrukturen *mit* Pseudoknoten



$$. \quad . \quad (\quad . \quad (\quad (\quad [\quad) \quad) \quad . \quad] \quad)$$

vereinfacht und verallgemeinert: $L = \mathbf{(}^i\mathbf{[}^j\mathbf{)}^i\mathbf{]}^j$ mit $i, j > 0$

Pumping Lemma für kontextfreie Sprachen verletzt

$\rightarrow$ Kontextfreie Grammatiken nicht anwendbar

$\rightarrow$ ADP nicht anwendbar

Multiple kontextfreie Grammatiken (Seki et al., 1991)

= Baumgrammatik + Umschreibealgebra über Worttupel

$$
\begin{aligned}
S &\to \mathrm{r}_1[M, N] \\
M &\to \mathrm{r}_2[M, \texttt{(}, \texttt{)}] \mid (\texttt{(}, \texttt{)}) \\
N &\to \mathrm{r}_2[N, \texttt{[}, \texttt{]}] \mid (\texttt{[}, \texttt{]})
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{r}_1[(m_1, m_2), (n_1, n_2)] &= m_1 n_1 m_2 n_2 \\
\mathrm{r}_2[(x_1, x_2), l, r] &= (x_1 l, r x_2)
\end{aligned}
$$

Multiple kontextfreie Grammatiken (Seki et al., 1991)

= Baumgrammatik + Umschreibealgebra über Worttupel

$$
\begin{aligned}
S &\rightarrow \mathrm{r}_1[M, N] \\
M &\rightarrow \mathrm{r}_2[M, \mathbf{(}, \mathbf{)}] \mid (\mathbf{(}, \mathbf{)}) \\
N &\rightarrow \mathrm{r}_2[N, \mathbf{[}, \mathbf{]}] \mid (\mathbf{[}, \mathbf{]})
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{r}_1[(m_1, m_2), (n_1, n_2)] &= m_1 n_1 m_2 n_2 \\
\mathrm{r}_2[(x_1, x_2), l, r] &= (x_1 l, r x_2)
\end{aligned}
$$

Beispielterm generiert durch M

$$\mathrm{r}_2[\mathrm{r}_2[(\mathbf{(}, \mathbf{)}), \mathbf{(}, \mathbf{)}], \mathbf{(}, \mathbf{)}] = (\mathbf{(}\mathbf{(}\mathbf{(}, \mathbf{)}\mathbf{)}\mathbf{)})$$

Multiple kontextfreie Grammatiken (Seki et al., 1991)

= Baumgrammatik + Umschreibealgebra über Worttupel

$$
\begin{aligned}
S &\to \mathrm{r}_1[M, N] \\
M &\to \mathrm{r}_2[M, (,)] \mid ((,)) \\
N &\to \mathrm{r}_2[N, [,]] \mid ([,])
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{r}_1[(m_1, m_2), (n_1, n_2)] &= m_1 n_1 m_2 n_2 \\
\mathrm{r}_2[(x_1, x_2), l, r] &= (x_1 l, r x_2)
\end{aligned}
$$

Beispielterm generiert durch M

$$\mathrm{r}_2[\mathrm{r}_2[((,)), (,)], (,)] = ((((,))))$$

Beispielterm generiert durch S

$$\mathrm{r}_1[\mathrm{r}_2[\mathrm{r}_2[((,)), (,)], (,)], \mathrm{r}_2[([,]), [,]]] = (((\,[[\,)))\,]]$$

Multiple kontextfreie Grammatiken (Seki et al., 1991)

= Baumgrammatik + Umschreibealgebra über Worttupel

$$
\begin{aligned}
S &\to \mathrm{r}_1[M, N] \\
M &\to \mathrm{r}_2[M, \mathbf{(}, \mathbf{)}] \mid (\mathbf{(}, \mathbf{)}) \\
N &\to \mathrm{r}_2[N, \mathbf{[}, \mathbf{]}] \mid (\mathbf{[}, \mathbf{]})
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{r}_1[(m_1, m_2), (n_1, n_2)] &= m_1 n_1 m_2 n_2 \\
\mathrm{r}_2[(x_1, x_2), l, r] &= (x_1 l, r x_2)
\end{aligned}
$$

Beispielterm generiert durch M

$$\mathrm{r}_2[\mathrm{r}_2[(\mathbf{(}, \mathbf{)}), \mathbf{(}, \mathbf{)}], \mathbf{(}, \mathbf{)}] = (\mathbf{(((}, \mathbf{)))})$$

Beispielterm generiert durch S

$$\mathrm{r}_1[\mathrm{r}_2[\mathrm{r}_2[(\mathbf{(}, \mathbf{)}), \mathbf{(}, \mathbf{)}], \mathbf{(}, \mathbf{)}], \mathrm{r}_2[(\mathbf{[}, \mathbf{]}), \mathbf{[}, \mathbf{]}]] = \mathbf{((([[)))]]}$$

Bewertung?

Status Quo

Kein Framework vorhanden zum Lösen von kombinatorischen Optimierungsproblemen mit Suchräumen, die durch multiple kontextfreie Grammatiken beschrieben werden

Status Quo

Kein Framework vorhanden zum Lösen von kombinatorischen Optimierungsproblemen mit Suchräumen, die durch multiple kontextfreie Grammatiken beschrieben werden

Folgen

- Problemspezifische Implementierungen
- Ad-hoc Anpassungen von ADP-Implementierungen

Status Quo

Kein Framework vorhanden zum Lösen von kombinatorischen Optimierungsproblemen mit Suchräumen, die durch multiple kontextfreie Grammatiken beschrieben werden

Folgen

- Problemspezifische Implementierungen
- Ad-hoc Anpassungen von ADP-Implementierungen

Probleme

- schwer wiederverwendbar
- aufwändig und fehleranfällig

Ziele der Arbeit

1. Neudefinierung von ADP für multiple kontextfreie Sprachen

2. Entwicklung eines Prototyps

Ziele der Arbeit

1. Neudefinierung von ADP für multiple kontextfreie Sprachen
   - Leichte Übersetzung von multiplen kontextfreien Grammatiken
   - Anwendbarkeit von dynamischer Programmierung
   - Bestimmung asymptotischer Laufzeit von Probleminstanzen
2. Entwicklung eines Prototyps

Ziele der Arbeit

1. Neudefinierung von ADP für multiple kontextfreie Sprachen
   - Leichte Übersetzung von multiplen kontextfreien Grammatiken
   - Anwendbarkeit von dynamischer Programmierung
   - Bestimmung asymptotischer Laufzeit von Probleminstanzen
2. Entwicklung eines Prototyps
   - Leichte Benutzbarkeit (Maß: Ähnlichkeit zu formaler Notation)
   - Referenz für zukünftige Implementierungen
   - Beschränkung auf Wortpaare

Neudefinierung von ADP

Zwei Ansätze

1. Multiple kontextfreie Grammatik mit Bewertungsalgebren
2. Baumgrammatik über Kreuzprodukt von Umschreibe- und Bewertungsalgebren

Neudefinierung von ADP – Ansatz 1

$$S \rightarrow \mathrm{r}_1(B, S) \mid \mathrm{r}_1(P, S)$$

Umschreibealgebra:

$\mathrm{r}_1(x, y) = xy$

Bewertungsalgebra:

$\mathrm{r}_1(x, y) = x + y$

Neudefinierung von ADP – Ansatz 1

$$S \rightarrow \mathrm{r}_1(B, S) \mid \mathrm{r}_1(P, S)$$

Umschreibealgebra:
$\mathrm{r}_1(x, y) = xy$

Bewertungsalgebra:
$\mathrm{r}_1(x, y) = x + y$

$$S \rightarrow \mathrm{r}_1(B, S) \mid \mathrm{r}_2(P, S)$$

Neudefinierung von ADP – Ansatz 1

$$S \rightarrow \mathrm{r}_1(B, S) \mid \mathrm{r}_1(P, S)$$

Umschreibealgebra:

$\mathrm{r}_1(x, y) = xy$

Bewertungsalgebra:

$\mathrm{r}_1(x, y) = x + y$

$$S \rightarrow \mathrm{r}_1(B, S) \mid \mathrm{r}_2(P, S)$$

Umschreibealgebra:

$\mathrm{r}_1(x, y) = xy$

$\mathrm{r}_2(x, y) = xy$

Bewertungsalgebra:

$\mathrm{r}_1(x, y) = x + y$

$\mathrm{r}_2(x, y) = 2x + y$

Neudefinierung von ADP – Ansatz 1

$$S \rightarrow \mathrm{r}_1(B, S) \mid \mathrm{r}_1(P, S)$$

Umschreibealgebra:

$$\mathrm{r}_1(x, y) = xy$$

Bewertungsalgebra:

$$\mathrm{r}_1(x, y) = x + y$$

$$S \rightarrow \mathrm{r}_1(B, S) \mid \mathrm{r}_2(P, S)$$

Umschreibealgebra:

$$\mathrm{r}_1(x, y) = xy$$

$$\mathrm{r}_2(x, y) = xy$$

Bewertungsalgebra:

$$\mathrm{r}_1(x, y) = x + y$$

$$\mathrm{r}_2(x, y) = 2x + y$$

$\Rightarrow$ Redundanz in Umschreibealgebra

$\Rightarrow$ Änderung in Bewertung führt zu Änderung in Umschreibung

Neudefinierung von ADP – Ansatz 2

$$S \to (\mathrm{f}_1, \mathrm{r}_1)(B, S) \mid (\mathrm{f}_1, \mathrm{r}_1)(P, S)$$

Umschreibealgebra:
$\mathrm{r}_1(x, y) = xy$

Bewertungsalgebra:
$\mathrm{f}_1(x, y) = x + y$

Neudefinierung von ADP – Ansatz 2

$$S \rightarrow (\mathsf{f}_1, \mathsf{r}_1)(B, S) \mid (\mathsf{f}_1, \mathsf{r}_1)(P, S)$$

Umschreibealgebra:

$\mathsf{r}_1(x, y) = xy$

Bewertungsalgebra:

$\mathsf{f}_1(x, y) = x + y$

$$S \rightarrow (\mathsf{f}_1, \mathsf{r}_1)(B, S) \mid (\mathsf{f}_2, \mathsf{r}_1)(P, S)$$

Neudefinierung von ADP – Ansatz 2

$$S \rightarrow (\mathrm{f}_1, \mathrm{r}_1)(B, S) \mid (\mathrm{f}_1, \mathrm{r}_1)(P, S)$$

Umschreibealgebra:

$\mathrm{r}_1(x, y) = xy$

Bewertungsalgebra:

$\mathrm{f}_1(x, y) = x + y$

$$S \rightarrow (\mathrm{f}_1, \mathrm{r}_1)(B, S) \mid (\mathrm{f}_2, \mathrm{r}_1)(P, S)$$

Umschreibealgebra:

$\mathrm{r}_1(x, y) = xy$

Bewertungsalgebra:

$\mathrm{f}_1(x, y) = x + y$

$\mathrm{f}_2(x, y) = 2x + y$

Neudefinierung von ADP – Ansatz 2

$$S \rightarrow (\mathrm{f}_1, \mathrm{r}_1)(B, S) \mid (\mathrm{f}_1, \mathrm{r}_1)(P, S)$$

Umschreibealgebra:
$\mathrm{r}_1(x, y) = xy$

Bewertungsalgebra:
$\mathrm{f}_1(x, y) = x + y$

$$S \rightarrow (\mathrm{f}_1, \mathrm{r}_1)(B, S) \mid (\mathrm{f}_2, \mathrm{r}_1)(P, S)$$

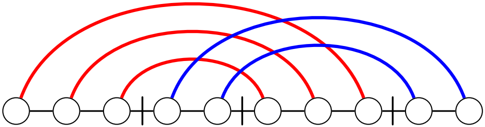
Umschreibealgebra:
$\mathrm{r}_1(x, y) = xy$

Bewertungsalgebra:
$\mathrm{f}_1(x, y) = x + y$
$\mathrm{f}_2(x, y) = 2x + y$

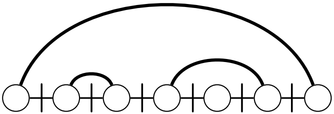
$\Rightarrow$ keine Redundanz in Umschreibealgebra
$\Rightarrow$ Änderung in Bewertung führt nicht zu Änderung in Umschreibung
$\Rightarrow$ etwas längere Produktionen

Beispiel – C2u Sekundärstrukturen

2 Strukturelemente



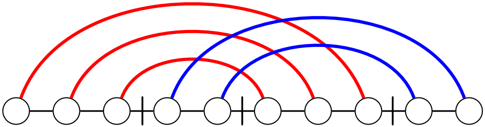
Typ H Pseudoknoten



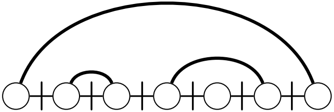
Struktur ohne Pseudoknoten

Beispiel – C2u Sekundärstrukturen

2 Strukturelemente



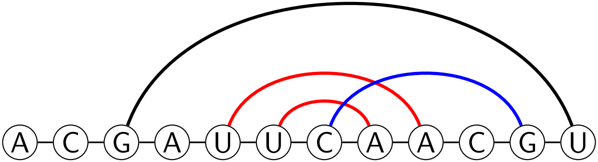
Typ H Pseudoknoten



Struktur ohne Pseudoknoten

C2u Sekundärstrukturen

= Verkettung und gegenseitige Einbettung von beiden Strukturelementen



Beispiel – C2u Sekundärstrukturen

$$S \rightarrow (\mathsf{f}_1, \mathsf{r}_1)(B, S) \mid (\mathsf{f}_2, \mathsf{r}_2)(P, S, S) \mid (\mathsf{f}_3, \mathsf{r}_3)()$$

$$P \rightarrow (\mathsf{f}_4, \mathsf{id}_2)((\mathbf{a}, \mathbf{u})) \mid (\mathsf{f}_4, \mathsf{id}_2)((\mathbf{u}, \mathbf{a})) \mid \ldots$$

$$B \rightarrow (\mathsf{f}_5, \mathsf{id}_1)(\mathbf{a}) \mid (\mathsf{f}_5, \mathsf{id}_1)(\mathbf{g}) \mid (\mathsf{f}_5, \mathsf{id}_1)(\mathbf{c}) \mid (\mathsf{f}_5, \mathsf{id}_1)(\mathbf{u})$$

Umschreibealgebra:

$$\begin{aligned}
\mathsf{id}_1(x) &= x \\
\mathsf{id}_2((x_1, x_2)) &= (x_1, x_2) \\
\mathsf{r}_1(b, s) &= bs \\
\mathsf{r}_2((p_1, p_2), s_1, s_2) &= p_1 s_1 p_2 s_2 \\
\mathsf{r}_3() &= \epsilon
\end{aligned}$$

Bewertungsalgebra:

$$\begin{aligned}
\mathsf{f}_1(b, s) &= s \\
\mathsf{f}_2(p, s_1, s_2) &= p + s_1 + s_2 \\
\mathsf{f}_3() &= 0 \\
\mathsf{f}_4((b_1, b_2)) &= 1 \\
\mathsf{f}_5(b) &= 0
\end{aligned}$$

Beispiel – C2u Sekundärstrukturen

$$S \to (\mathrm{f}_1, \mathrm{r}_1)(B, S) \mid (\mathrm{f}_2, \mathrm{r}_2)(P, S, S) \mid (\mathrm{f}_3, \mathrm{r}_3)() \mid (\mathrm{f}_6, \mathrm{r}_4)(M, M, S, S, S, S)$$

$$P \to (\mathrm{f}_4, \mathrm{id}_2)((\mathbf{a}, \mathbf{u})) \mid (\mathrm{f}_4, \mathrm{id}_2)((\mathbf{u}, \mathbf{a})) \mid \ldots$$

$$B \to (\mathrm{f}_5, \mathrm{id}_1)(\mathbf{a}) \mid (\mathrm{f}_5, \mathrm{id}_1)(\mathbf{g}) \mid (\mathrm{f}_5, \mathrm{id}_1)(\mathbf{c}) \mid (\mathrm{f}_5, \mathrm{id}_1)(\mathbf{u})$$

$$M \to (\mathrm{f}_7, \mathrm{r}_5)(M, P) \mid (\mathrm{f}_8, \mathrm{id}_2)(P)$$

Umschreibealgebra:

$$\begin{aligned}
\mathrm{id}_1(x) &= x \\
\mathrm{id}_2((x_1, x_2)) &= (x_1, x_2) \\
\mathrm{r}_1(b, s) &= bs \\
\mathrm{r}_2((p_1, p_2), s_1, s_2) &= p_1 s_1 p_2 s_2 \\
\mathrm{r}_3() &= \epsilon \\
\mathrm{r}_4((m_1, m_2), (n_1, n_2), \ldots) &= m_1 s_1 n_1 s_2 m_2 s_3 n_1 s_4 \\
\mathrm{r}_5((m_1, m_2), (p_1, p_2)) &= (m_1 p_1, p_2 m_2)
\end{aligned}$$

Bewertungsalgebra:

$$\begin{aligned}
\mathrm{f}_1(b, s) &= s \\
\mathrm{f}_2(p, s_1, s_2) &= p + s_1 + s_2 \\
\mathrm{f}_3() &= 0 \\
\mathrm{f}_4((b_1, b_2)) &= 1 \\
\mathrm{f}_5(b) &= 0 \\
\mathrm{f}_6(m_1, \ldots, s_4) &= m_1 + \ldots + s_4 \\
\mathrm{f}_7(m, p) &= m + p \\
\mathrm{f}_8(p) &= p
\end{aligned}$$

Beispiel – C2u Sekundärstrukturen

$$S \to (\mathsf{f}_1, \mathsf{r}_1)(B, S) \mid (\mathsf{f}_2, \mathsf{r}_2)(P, S, S) \mid (\mathsf{f}_3, \mathsf{r}_3)() \mid (\mathsf{f}_6, \mathsf{r}_4)({\color{red}M}, {\color{blue}M}, S, S, S, S)$$
$$P \to (\mathsf{f}_4, \mathsf{id}_2)((\mathbf{a}, \mathbf{u})) \mid (\mathsf{f}_4, \mathsf{id}_2)((\mathbf{u}, \mathbf{a})) \mid \ldots$$
$$B \to (\mathsf{f}_5, \mathsf{id}_1)(\mathbf{a}) \mid (\mathsf{f}_5, \mathsf{id}_1)(\mathbf{g}) \mid (\mathsf{f}_5, \mathsf{id}_1)(\mathbf{c}) \mid (\mathsf{f}_5, \mathsf{id}_1)(\mathbf{u})$$
$$M \to (\mathsf{f}_7, \mathsf{r}_5)(M, P) \mid (\mathsf{f}_8, \mathsf{id}_2)(P)$$

Umschreibealgebra:

$$\begin{aligned}
\mathsf{id}_1(x) &= x \\
\mathsf{id}_2((x_1, x_2)) &= (x_1, x_2) \\
\mathsf{r}_1(b, s) &= bs \\
\mathsf{r}_2((p_1, p_2), s_1, s_2) &= p_1 s_1 p_2 s_2 \\
\mathsf{r}_3() &= \epsilon \\
\mathsf{r}_4(({\color{red}m_1}, {\color{red}m_2}), ({\color{blue}n_1}, {\color{blue}n_2}), \ldots) &= {\color{red}m_1} s_1 {\color{blue}n_1} s_2 {\color{red}m_2} s_3 {\color{blue}n_2} s_4 \\
\mathsf{r}_5((m_1, m_2), (p_1, p_2)) &= (m_1 p_1, p_2 m_2)
\end{aligned}$$

Bewertungsalgebra:

$$\begin{aligned}
\mathsf{f}_1(b, s) &= s \\
\mathsf{f}_2(p, s_1, s_2) &= p + s_1 + s_2 \\
\mathsf{f}_3() &= 0 \\
\mathsf{f}_4((b_1, b_2)) &= 1 \\
\mathsf{f}_5(b) &= 0 \\
\mathsf{f}_6(m_1, \ldots, s_4) &= m_1 + \ldots + s_4 \\
\mathsf{f}_7(m, p) &= m + p \\
\mathsf{f}_8(p) &= p
\end{aligned}$$

Prototyp **adp-multi** – Beispiel (C2u Sekundärstrukturen)

Umschreibealgebra:

```
id1 [x]                        = [x]
id2 [x1,x2]                    = ([x1],[x2])
r1 [b,s]                       = [b,s]
r2 [p1,p2,s1,s2]               = [p1,s1,p2,s2]
r3 [e]                         = [e]
r4 [m1,m2,n1,n2,s1,s2,s3,s4] = [m1,s1,n1,s2,m2,s3,n2,s4]
r5 [m1,m2,p1,p2]               = ([m1,p1],[p2,m2])
```

Bewertungsalgebra:

```
f1 b s                  = s
f2 p s1 s2              = p + s1 + s2
f3 _                    = 0
f4 _                    = 1
f5 _                    = 0
f6 m1 m2 s1 s2 s3 s4 = m1 + m2 + s1 + s2 + s3 + s4
f7 m p                  = m + p
f8 p                    = p
```

```
s = yieldSize1 (0, Nothing) $
    f1 <<< b ~~~ s                       >>> r1 |||
    f2 <<< p ~~~ s ~~~ s                 >>> r2 |||
    f3 <<< ""                            >>> r3 |||
    f6 <<< m ~~~ m ~~~ s ~~~ s ~~~ s ~~~ s >>> r4


p = f4 <<< ("a","u") >>> id2 |||
    f4 <<< ("u","a") >>> id2 |||
    ⋮

b = f5 <<< "a" >>> id1 ||| f5 <<< "g" >>> id1 |||
    f5 <<< "c" >>> id1 ||| f5 <<< "u" >>> id1


m = yieldSize2 (1, Nothing) (1, Nothing) $
    f7 <<< m ~~~ p >>> r5  |||
    f8 <<< p       >>> id2
```

```
> axiom (mk "agcguu") s
[0,1,1,1,1,2,2,1,2,1,2,2,2,1,2,2,1,2,2,2,3,2,3,2,3,2,3]
```

```
s = yieldSize1 (0, Nothing) $
    f1 <<< b ~~~ s                              >>> r1 |||
    f2 <<< p ~~~ s ~~~ s                        >>> r2 |||
    f3 <<< ""                                   >>> r3 |||
    f6 <<< m ~~~ m ~~~ s ~~~ s ~~~ s ~~~ s >>> r4
    ... h
```

Optimierungsziel:

```
h [] = []
h l  = [maximum l]
```

```
p = f4 <<< ("a","u") >>> id2 |||
    f4 <<< ("u","a") >>> id2 |||
    ⋮

b = f5 <<< "a" >>> id1 ||| f5 <<< "g" >>> id1 |||
    f5 <<< "c" >>> id1 ||| f5 <<< "u" >>> id1


m = yieldSize2 (1, Nothing) (1, Nothing) $
    f7 <<< m ~~~ p >>> r5  |||
    f8 <<< p       >>> id2
    ... h
```

```
> axiom (mk "agcguu") s
[3]
```

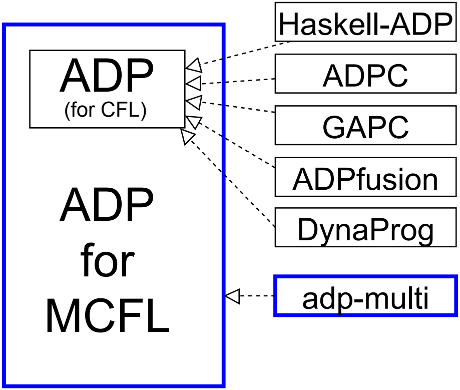
```
s = tabulated1 $
    yieldSize1 (0, Nothing) $
    f1 <<< b ~~~ s                               >>> r1 |||
    f2 <<< p ~~~ s ~~~ s                         >>> r2 |||
    f3 <<< ""                                    >>> r3 |||
    f6 <<< m ~~~ m ~~~ s ~~~ s ~~~ s ~~~ s >>> r4
    ... h

p = f4 <<< ("a","u") >>> id2 |||
    f4 <<< ("u","a") >>> id2 |||
    ⋮

b = f5 <<< "a" >>> id1 ||| f5 <<< "g" >>> id1 |||
    f5 <<< "c" >>> id1 ||| f5 <<< "u" >>> id1

m = tabulated2 $
    yieldSize2 (1, Nothing) (1, Nothing) $
    f7 <<< m ~~~ p >>> r5  |||
    f8 <<< p       >>> id2
    ... h
```

Optimierungsziel:

```
h [] = []
h l  = [maximum l]
```

```
> axiom (mk "agcguu") s
[3]
```

Zusammenfassung



Ausblick

- Fortgeschrittenere Implementierungen
- Untersuchung größerer Sprachklassen

Dynamische Programmierung

Für kontextfreie Grammatiken: Cocke-Younger-Kasami (1970er)

Beispiel

Eingabewort $w = \mathbf{agcgu}$
Matrix für Nichtterminal P:

j

		0					5
	0	F	F	F	F	F	T
			F	F	T	F	T
i				F	F	T	F
					F	F	T
						F	F
	5						F

$P[i, j] =$ `True`, wenn Ableitung für $w[i..j]$ existiert, sonst `False`

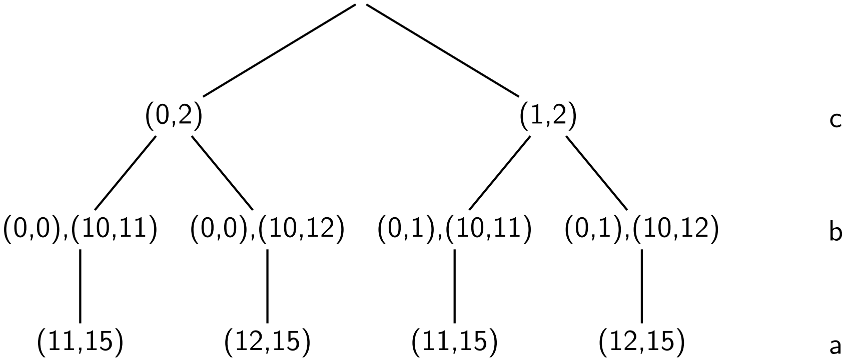
Prototyp **adp-multi** – Minimale/maximale Wortlängen

```
f <<< a ~~~ b ~~~ c >>> r

type Dim2 = [(Int , Int)] -> ([(Int , Int)], [(Int , Int)])

r :: Dim2
r [a,b1,b2,c] = ([b1,c],[b2,a])

infos = [ ParserInfo1 0 Nothing
        , ParserInfo2 (0,1) (Just 2,Just 2)
        , ParserInfo1 1 (Just 4) ]

y = determineYieldSize2 r infos

r [(1,1),(2,1),(2,2),(3,1)] = ([(2,1),(3,1)],[(2,2),(1,1)])

y = ParserInfo2 ( 0+0 , 1+1 )
                ( liftM2 (+) (Just 2) Nothing
                , liftM2 (+) (Just 2) (Just 4) )

y = ParserInfo2 (0,2) (Nothing,Just 6)
```

Prototyp **adp-multi** – Teilwortkonstruktion

```
f <<< a ~~~ b ~~~ c >>> r
r [a,b1,b2,c] = ([b1,c],[b2,a])

infos = [ ParserInfo1 1 Nothing
        , ParserInfo2 (0,1) (Just 2,Just 2)
        , ParserInfo1 1 (Just 4) ]

subword = [0,2,10,15]

y = constructSubwords2 r infos subword
--                 b1     c                b2     a
ranges = [ (0,2,[(2,1),(3,1)]) , (10,15,[(2,2),(1,1)]) ]

-- Start mit c
subwordsC = [ (0,2)         ,          (1,2)       ]

-- für jedes Teilwort von c: neue Ranges ohne (3,1)
      [ (0,0,[(2,1)]) , ~ ]   [ (0,1,[(2,1)]) , ~ ]

-- weiter mit b, a
```

Prototyp **adp-multi** – Teilwortkonstruktion



Pfad im Baum

=

Aufteilung des Teilwortpaares auf Parser der Produktion

Prototyp **adp-multi** – Teilwortkonstruktion

```
y = [
      SubwordTree [0,2] [
        SubwordTree [0,0,10,11] [
          SubwordTree [11,15] []
        ],
        SubwordTree [0,0,10,12] [
          SubwordTree [12,15] []
        ]
      ],
      SubwordTree [1,2] [
        SubwordTree [0,1,10,11] [
          SubwordTree [11,15] []
        ],
        SubwordTree [0,1,10,12] [
          SubwordTree [12,15] []
        ]
      ]
    ]
```

Prototyp **adp-multi** – Teilwortkonstruktion

Haskell-ADP

$i \quad k_1 \quad k_2 \quad j$

```
f <<< p1 ~~~ p2 ~~~ p3
```

```
for k2 = i to j
   for k1 = i to k2
```

~~~ erzeugt Teilwortgrenze (Laufindex) anliegender Parser

adp-multi

$i \quad k_1 \quad k_2 \quad j$

```
r [q1,q21,q22] = [q21 , q1 , q22]

f <<< q1 ~~~ q2 >>> r
```

```
for k1 = i to j
   for k2 = k1 to j
```

>>> erzeugt Teilwortgrenzen aller Parser
~~~ nimmt Teilwortindizes an