Synchronous Rewriting for Natural Language Processing

Giorgio Satta
University of Padua, Italy

Prague, July 23rd 2014

Introduction

In a **synchronous grammar** two or more derivation processes can be carried out in a synchronous way

Example: A synchronous grammar can generate strings of the form $(n \geq 1)$

$$\cdots \quad \underbrace{\underbrace{a^n \quad \cdots \quad b^n}_{\text{context-free}} \quad \cdots \quad \underbrace{c^n \quad \cdots \quad d^n}_{\text{context-free}}}_{\text{synchronized}} \quad \cdots$$

Introduction (cont'd)

The term **synchronous grammar** has been introduced in the 90's in the **computational linguistics** community [Shieber and Schabes, 1990]

The technical idea can be found way back in the **formal language** literature, in relation to **parallel rewriting**

Introduction (cont'd)

Synchronous rewriting is a **hot research topic** in computational linguistics, with applications in

- machine translation
- syntax-semantics interface
- parsing of discontinuous phrase structures
- non-projective dependency parsing

We see a **convergence process** across different areas in natural language processing toward the use of synchronous rewriting

Synchronous context-free grammars

Synchronous context-free grammars are very popular among synchronous rewriting formalisms

Rooted in **theory of compilers**:

- Syntax-directed transduction grammars [Lewis and Stearns, 1968]
- Syntax-directed translation schemata [Aho and Ullman, 1969]

Synchronous context-free grammars (cont'd)

Exploited in **statistical machine translation**

- Inversion Transduction Grammars [Wu, 1997]
- Head Transducer Grammars [Alshawi et al., 2000]
- Tree-to-string models [Yamada and Knight, 2001]
- Multi-Text Grammars [Melamed, 2003]
- Hierarchical phrase-based models [Chiang, 2005]

Achieve state-of-the-art performance [Chiang, 2005]

Synchronous production

A (context-free) **synchronous production** is a pair of context-free productions that must always be used **together**

A **pairing relation** (bijection) is defined over the nonterminals in the two right-hand sides

Example: $A_i, B_j, \ldots$ nonterminals, $a, b, \ldots$ terminals

$$[\, A_1 \rightarrow a\, B_1^{\boxed{1}}\ C_1^{\boxed{2}}\ b\, D_1^{\boxed{3}}\ E_1^{\boxed{4}},$$
$$A_2 \rightarrow D_2^{\boxed{3}}\ c\, B_2^{\boxed{1}}\ E_2^{\boxed{4}}\ C_2^{\boxed{2}}\ d\,]$$

Synchronous context-free grammar (cont'd)

A **synchronous context-free grammar** (SCFG) is based on a set of synchronous productions

Example: English to Japanese [Yamada and Knight, 2001]

$$
\begin{array}{lll}
s_1: & [\mathrm{VB} \rightarrow \mathrm{PRP}^{\boxed{1}}\ \mathrm{VB1}^{\boxed{2}}\ \mathrm{VB2}^{\boxed{3}}, & \mathrm{VB} \rightarrow \mathrm{PRP}^{\boxed{1}}\ \mathrm{VB2}^{\boxed{3}}\ \mathrm{VB1}^{\boxed{2}}] \\
s_2: & [\mathrm{VB2} \rightarrow \mathrm{VB}^{\boxed{1}}\ \mathrm{TO}^{\boxed{2}}, & \mathrm{VB2} \rightarrow \mathrm{TO}^{\boxed{2}}\ \mathrm{VB}^{\boxed{1}}\ \mathrm{ga}] \\
s_3: & [\mathrm{TO} \rightarrow \mathrm{TO}^{\boxed{1}}\ \mathrm{NN}^{\boxed{2}}, & \mathrm{TO} \rightarrow \mathrm{NN}^{\boxed{2}}\ \mathrm{TO}^{\boxed{1}}] \\
s_4: & [\mathrm{PRP} \rightarrow \mathrm{he}, & \mathrm{PRP} \rightarrow \mathrm{kare\ ha}] \\
s_5: & [\mathrm{VB1} \rightarrow \mathrm{adores}, & \mathrm{VB1} \rightarrow \mathrm{daisuki\ desu}] \\
s_6: & [\mathrm{VB} \rightarrow \mathrm{listening}, & \mathrm{VB} \rightarrow \mathrm{kiku\ no}] \\
s_7: & [\mathrm{TO} \rightarrow \mathrm{to}, & \mathrm{TO} \rightarrow \mathrm{wo}] \\
s_8: & [\mathrm{NN} \rightarrow \mathrm{music}, & \mathrm{NN} \rightarrow \mathrm{ongaku}]
\end{array}
$$

Derivation

A SCFG generates pairs of strings/trees, representing the desired **translation**

The **rewrite** relation applies a synchronous production to **simultaneously** rewrite two paired nonterminals

Pairing relation must be **updated** after each application of a synchronous production

Derivation (cont'd)

Example:

$$[\text{VB}^{\boxed{1}},\ \text{VB}^{\boxed{1}}]$$

$$\xRightarrow{s_1}_G \quad [\text{PRP}^{\boxed{2}}\ \text{VB1}^{\boxed{3}}\ \text{VB2}^{\boxed{4}},\ \ \text{PRP}^{\boxed{2}}\ \text{VB2}^{\boxed{4}}\ \text{VB1}^{\boxed{3}}]$$

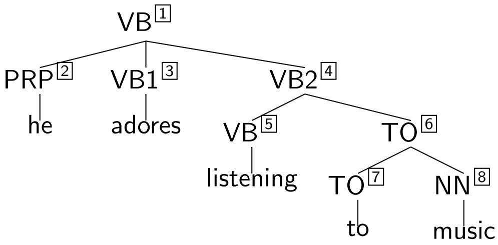
$$\xRightarrow{s_2}_G \quad [\text{PRP}^{\boxed{2}}\ \text{VB1}^{\boxed{3}}\ \text{VB}^{\boxed{5}}\ \text{TO}^{\boxed{6}},\ \ \text{PRP}^{\boxed{2}}\ \text{TO}^{\boxed{6}}\ \text{VB}^{\boxed{5}}\ \text{VB1}^{\boxed{3}}]$$

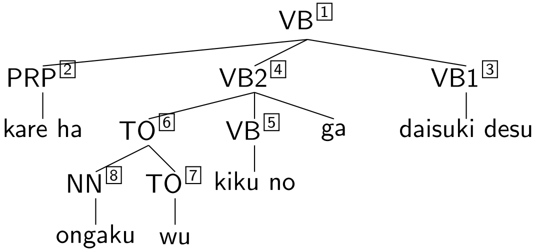
$$\xRightarrow{s_4}_G \quad [\text{he VB1}^{\boxed{3}}\ \text{VB}^{\boxed{5}}\ \text{TO}^{\boxed{6}},\ \ \text{kare ha TO}^{\boxed{6}}\ \text{VB}^{\boxed{5}}\ \text{VB1}^{\boxed{3}}]$$

$$\xRightarrow{s_5}_G \quad [\text{he adores VB}^{\boxed{5}}\ \text{TO}^{\boxed{6}},\ \ \text{kare ha TO}^{\boxed{6}}\ \text{VB}^{\boxed{5}}\ \text{daisuki desu}]$$

$$\vdots$$

Derivation (cont'd)





Translation relation

Translation relation:
Set of all string pairs generated by G

$$T(G) = \{[u, v] \mid [S^{\boxed{1}}, S^{\boxed{1}}] \overset{*}{\Rightarrow}_G [u, v]\}$$

Probabilistic SCFGs

In a **probabilistic SCFG**, each synchronous production is associated with a probability

$$p_G([A_1 \rightarrow \alpha_1,\ A_2 \rightarrow \alpha_2])$$

Normalization condition for each pair $[A_1, A_2]$

$$\sum_{\alpha_1, \alpha_2} p_G([A_1 \rightarrow \alpha_1,\ A_2 \rightarrow \alpha_2]) \;=\; 1$$

Probabilistic PSCFGs (cont'd)

We can define several **joint distributions** (t_i trees, w_i strings, y = yield)

$$\begin{aligned} p_G([t_1,\ t_2]) &= \prod_{i=1}^{n} p_G(s_i) \\ p_G([w_1,\ w_2]) &= \sum_{\substack{[y(t_1),y(t_2)] \\ =[w_1,w_2]}} p_G([t_1, t_2]) \\ p_G([w_1,\ t_2]) &= \sum_{y(t_1)=w_1} p_G([t_1, t_2]) \\ &\vdots \end{aligned}$$

Synchronous Parsing

Synchronous parsing problem :

- Input: SCFG G and string pair (u, v)
- Output: **parse forest** with all tree pairs for (u, v) under G

Synchronous parsing exploited in **training** of statistical models

Synchronous Parsing (cont'd)

Standard **dynamic programming** algorithms for parsing of context-free grammars can be extended to SCFGs

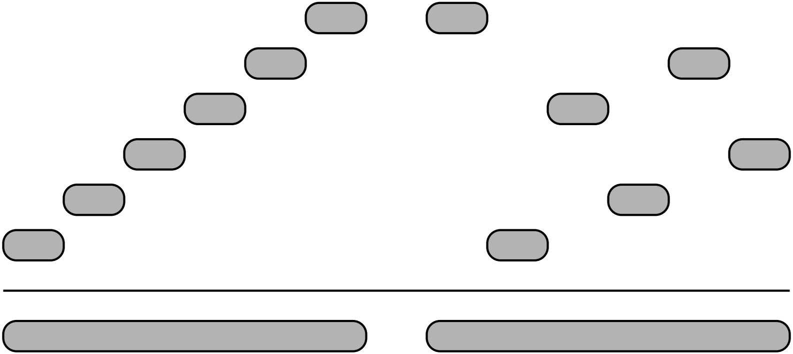
These algorithms run in time $\mathcal{O}(|G| \cdot n^{\sigma})$, with

- G the input grammar
- $n = \max\{|u|, |v|\}$
- σ the maximum number of nonterminals in a synchronous production

Unfortunately, there is no Chomsky normal form for SCFGs

Synchronous Parsing (cont'd)

Graphical representation of a synchronous rule; **paired nonterminals** at same level



Synchronous Parsing (cont'd)

Trace of a **parsing strategy** collecting nonterminals in left rule from left to right

Synchronous Parsing (cont'd)

Trace of a **parsing strategy**

Synchronous Parsing (cont'd)

Trace of a **parsing strategy**

Translation

Translation problem

- Input: SCFG G and string w
- Output: **Parse forest** with all trees for translations of w under G

Can be solved using a **generalization** of the Bar-Hillel construction for intersection of context-free languages with regular languages

Translation algorithm

Input: SCFG G, string w

Algorithm:

- construct M_1 such that $L(M_1) = \{w\}$
- construct M_2 such that $L(M_2) = \Sigma^*$
- construct $G_\cap$ by intersection of G with M_1 and M_2
- output context-free grammar obtained as **right projection** of $G_\cap$

Output is a **compact representation** of the parse forest of all trees for translations of w

Prefix-probability

Prefix-probability problem

- Input: Probabilistic SCFG G and string pair (u, v)
- Output: **Probability** of set

$$\{(u, vx) \mid x \text{ some string of terminals}\}$$

Prefix-probabilities exploited in computation of next-word conditional distributions for guiding **speech** applications

Production factorization

In the **worst case**, we cannot **factorize** a synchronous production into productions of smaller rank [Aho and Ullman, 1969]

Example:

$$[\, A_1 \rightarrow B_1^{\boxed{1}}\, C_1^{\boxed{2}}\, D_1^{\boxed{3}}\, E_1^{\boxed{4}}, \quad A_2 \rightarrow D_2^{\boxed{3}}\, B_2^{\boxed{1}}\, E_2^{\boxed{4}}\, C_2^{\boxed{2}} \,]$$

Try to factorize the two components of nonterminals C and D with **fresh nonterminal** F:

$$[\, A_1 \rightarrow B_1^{\boxed{1}}\, F_1^{\boxed{5}}\, E_1^{\boxed{4}}, \qquad A_2 \rightarrow F_2^{\boxed{5}}\, B_2^{\boxed{1}}\, E_2^{\boxed{4}}\, F_3^{\boxed{5}} \,]$$
$$[\, F_1 \rightarrow C_1^{\boxed{2}}\, D_1^{\boxed{3}}, \quad F_2 \rightarrow D_2^{\boxed{3}}, \quad F_3 \rightarrow C_2^{\boxed{2}} \,]$$

Outcome **is not** a SCFG

Production factorization (cont'd)

Worst cases **are rare/do not affect** translation performance [Zhang et al., 2006]

Factorization problem

- Input: synchronous production p of **rank** r
- Output: Set of synchronous productions **strongly equivalent** to p and with maximum rank **as small as possible**

Production factorization (cont'd)

Several **efficient** algorithms for the factorization problem have been discovered recently (r the rank)

- time $\mathcal{O}(r^2)$ [Zhang et al., 2006];
  use **shift-reduce** techniques
- time $\mathcal{O}(r \log(r))$ [Gildea et al., 2006];
  use **divide-and-conquer**
- time $\mathcal{O}(r)$ [Zhang and Gildea, 2007];
  use **dynamic** data structures

Generalized synchronous grammars

Generalize synchronous productions to **arbitrary dimensions**

$$[\, A \rightarrow B^{\boxed{1}}\ C^{\boxed{2}}\ D^{\boxed{3}}\ E^{\boxed{4}},\\ A \rightarrow C^{\boxed{2}}\ B^{\boxed{1}}\ E^{\boxed{4}}\ D^{\boxed{3}},\\ A \rightarrow E^{\boxed{4}}\ C^{\boxed{2}}\ D^{\boxed{3}}\ B^{\boxed{1}}\,]$$

Generalize by **mixing up indices** across dimensions

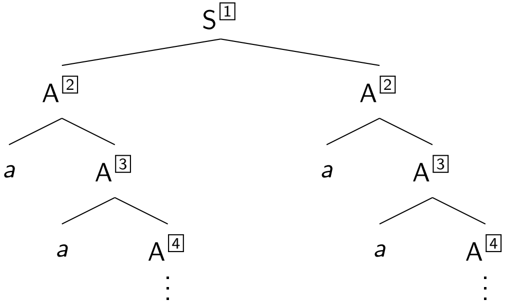
$$[\, A \rightarrow D^{\boxed{3}}\ C^{\boxed{2}}\ E^{\boxed{4}}\ D^{\boxed{3}}\ C^{\boxed{2}},\ D^{\boxed{3}}\\ A \rightarrow E^{\boxed{4}}\ B^{\boxed{1}},\\ A \rightarrow B^{\boxed{1}}\ E^{\boxed{4}}\ C^{\boxed{2}}\ B^{\boxed{1}}\,]$$

Generalized synchronous grammars (cont'd)

Example: copy language

$[S \rightarrow A^{\boxed{1}}\ A^{\boxed{1}}]$ $[A \rightarrow aA^{\boxed{1}},\ A \rightarrow aA^{\boxed{1}}]$

$[A \rightarrow \varepsilon,\ A \rightarrow \varepsilon]$ $[A \rightarrow bA^{\boxed{1}},\ A \rightarrow bA^{\boxed{1}}]$



Generalized synchronous grammars (cont'd)

The above class of grammars is generatively equivalent to several **mildly context-sensitive** formalisms

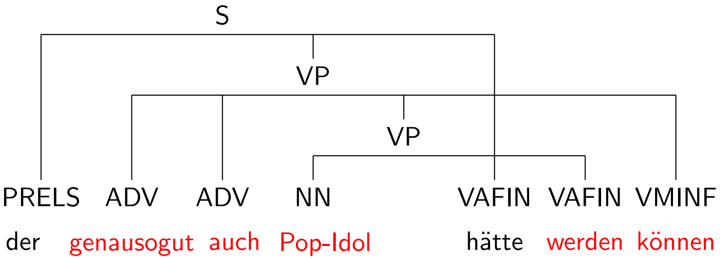
- Linear context-free rewriting system (LCFRS) [Vijay-Shanker et al., 1987]
- Multiple context-free grammar (MCFG) [Seki et al., 1991]
- Simple range concatenation grammar (s-RCG) [Boullier, 2004]
- Generalized multi-text grammars (GMTG) [Melamed et al., 2004]

Discontinuous phrases

Generalized synchronous grammars can generate **discontinuous** phrases

Exploited to model languages with **free word order** structure [Levy, 2005, Maier and Søgaard, 2008]

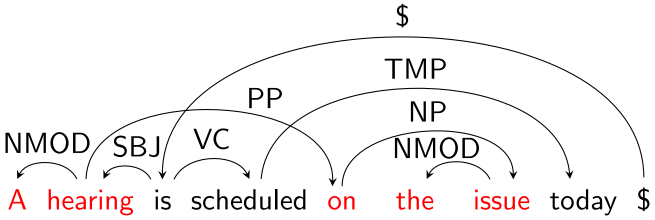
Example: NeGra treebank



Non-projective dependency structures

Generalized synchronous grammars also exploited to model (restricted) **non-projective dependency trees** [Kuhlmann and Nivre, 2006, Kuhlmann and Satta, 2009]

Example:



Parsing complexity

Existing algorithms for generalized synchronous grammars run in time $\mathcal{O}(|G| \cdot |w|^\sigma)$, with

- G the input grammar
- w the input string
- σ the maximum number of nonterminals in a synchronous production

Production factorization

Production factorization reduces asymptotical time complexity

For synchronous productions with two components, factorization takes time $\mathcal{O}(\sigma)$ [Gómez-Rodríguez and Satta, 2009]

In the general case, best known algorithms for factorization run in **exponential** time [Gómez-Rodríguez et al., 2009]

NP-hardness for the factorization problem **not known** [Gildea and Stefankovic, 2007]

Conclusions

Convergence toward the use of **synchronous rewriting** in several areas of computational linguistics

Parsing and translation problems for SCFGs can be **unified** using the **intersection framework**

The **factorization** problem for synchronous productions remains a **challenging** problem

A. V. Aho and J. D. Ullman.
1969.
Syntax directed translations and the pushdown assembler.
Journal of Computer and System Sciences, 3(1):37–56.

Hiyan Alshawi, Srinivas Bangalore, and Shona Douglas.
2000.
Learning dependency translation models as collections of finite state head transducers.
Computational Linguistics, 26(1):45–60, March.

Y. Bar-Hillel, M. Perles, and E. Shamir.
1964.
On formal properties of simple phrase structure grammars.
In Y. Bar-Hillel, editor, *Language and Information: Selected Essays on their Theory and Application*, chapter 9, pages 116–150. Addison-Wesley, Reading, Massachusetts.

E. Bertsch and M.-J. Nederhof.
2001.
On the complexity of some extensions of RCG parsing.
In *Proceedings of the Seventh International Workshop on Parsing Technologies*, pages 66–77, Beijing, China, October.

Pierre Boullier.
2004.
Range concatenation grammars.
In H. Bunt, J. Carroll, and G. Satta, editors, *New Developments in Parsing Technology*, volume 23 of *Text, Speech and Language Technology*, pages 269–289. Kluwer Academic Publishers.

D. Chiang.
2005.
A hierarchical phrase-based model for statistical machine translation.

In *Proc. of the 43rd ACL*, pages 263–270.

Daniel Gildea and Daniel Stefankovic.
2007.
Worst-case synchronous grammar rules.
In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 147–154, Rochester, New York, April. Association for Computational Linguistics.

D. Gildea, G. Satta, and H. Zhang.
2006.
Factoring synchronous grammars by sorting.
In *Proceedings of the 44th Annual Conference of the Association for Computational Linguistics (COLING/ACL-06)*, Sydney, Australia.

Carlos Gómez-Rodríguez and Giorgio Satta.

2009.
An optimal-time binarization algorithm for linear context-free rewriting systems with fan-out two.
In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 985–993, Suntec, Singapore, August. Association for Computational Linguistics.

C. Gómez-Rodríguez, M. Kuhlmann, G. Satta, and D. Weir. 2009.
Optimal reduction of rule length in linear context-free rewriting systems.
In *Proc. of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies Conference (NAACL'09:HLT)*, Boulder, Colorado.
To appear.

D.E. Knuth.
1977.
A generalization of Dijkstra's algorithm.
Information Processing Letters, 6(1):1–5, February.

Marco Kuhlmann and Joakim Nivre.
2006.
Mildly non-projective dependency structures.
In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 507–514, Sydney, Australia, July. Association for Computational Linguistics.

M. Kuhlmann and G. Satta.
2009.
Treebank grammar techniques for non-projective dependency parsing.
In *Proc. of the 12th EACL*, Athens, Greece.
To appear.

R. Levy.
2005.
Probabilistic models of word order and syntactic discontinuity.
Ph.D. thesis, Stanford University.

P. M. Lewis and R. E. Stearns.
1968.
Syntax-directed transduction.
Journal of the Association for Computing Machinery, 15(3):465–488.

Wolfgang Maier and Anders Søgaard.
2008.
Treebanks and mild context-sensitivity.
In Philippe de Groote, editor, *Proceedings of the 13th Conference on Formal Grammar (FG-2008)*, pages 61–76, Hamburg, Germany. CSLI Publications.

I. Dan Melamed, Giorgio Satta, and Ben Wellington.
2004.
Generalized multitext grammars.
In *Proceedings of ACL-04*.

I. Dan Melamed.
2003.
Multitext grammars and synchronous parsers.
In *Proceedings of the Human Language Technology Conference and the North American Association for Computational Linguistics (HLT-NAACL)*, pages 158–165, Edmonton, Canada.

Mark-Jan Nederhof and Giorgio Satta.
2008.
Computing partition functions of PCFGs.
Research on Language & Computation, 6(2):139–162.

M.-J. Nederhof.
2003.
Weighted deductive parsing and Knuth's algorithm.
Computational Linguistics, 29(1):135–143.

Rebecca Nesson and Stuart M. Shieber.
2006.
Simpler TAG semantics through synchronization.
In *Proceedings of the 11th Conference on Formal Grammar*, Malaga, Spain, 29-30 July.

Rebecca Nesson, Giorgio Satta, and Stuart M. Shieber.
2008.
Optimal *k*-arization of synchronous tree-adjoining grammar.
In *Proceedings of ACL-08: HLT*, pages 604–612, Columbus, Ohio, June. Association for Computational Linguistics.

O. Rambow and G. Satta.

1999.
Independent parallelism in finite copying parallel rewriting systems.
Theoretical Computer Science, 223:87–120.

G. Satta and E. Peserico.
2005.
Some computational complexity results for synchronous context-free grammars.
In *Proc. of the 2005 Conference on Empirical Methods in Natural Language Processing*, Vancouver, Canada.

H. Seki, T. Matsumura, M. Fujii, and T. Kasami.
1991.
On multiple context-free grammars.
Theoretical Computer Science, 88:191–229.

Stuart Shieber and Yves Schabes.
1990.

Synchronous tree adjoining grammars.
In *Proceedings of the 13^th International Conference on Computational Linguistics (COLING'90)*, Helsinki, August.

Stuart M. Shieber.
1994.
Restricting the weak-generative capacity of synchronous tree-adjoining grammars.
Computational Intelligence, 10(4):371–385.

Takeaki Uno and Mutsunori Yagiura.
2000.
Fast algorithms to enumerate all common intervals of two permutations.
Algorithmica, 26(2):290–309.

K. Vijay-Shanker, D. J. Weir, and A. K. Joshi.
1987.

Characterizing structural descriptions produced by various grammatical formalisms.
In *Proc. of the 25th ACL*, pages 104–111, Stanford, CA.

Dekai Wu.
1997.
Stochastic inversion transduction grammars and bilingual parsing of parallel corpora.
Computational Linguistics, 23(3):377–404, September.

Kenji Yamada and Kevin Knight.
2001.
A syntax-based statistical translation model.
In *Proceedings of ACL-01*.

H. Zhang and D. Gildea.
2007.
Factorization of synchronous context-free grammars in linear time.

In *NAACL Workshop on Syntax and Structure in Statistical Translation (SSST)*, pages 25–32, RFochester.

H. Zhang, L. Huang, D. Gildea, and K. Knight.
2006.
Synchronous binarization for machine translation.
In *Proc. of HLT/NAACL 2006 Conference*, New York.