BIDIREKTIONALE UTTERANCE–MEANING–TRANSDUCER FÜR ZAHLWORTE DURCH KOMPOSITIONALE MINIMALISTISCHE GRAMMATIKEN

Peter beim Graben, Werner Meyer, Ronald Römer und Matthias Wolff

BTU–Cottbus–Senftenberg
peter.beimgraben@b-tu.de

Kurzfassung: Sprachgesteuerte Benutzerschnittstellen stehen oftmals vor der Aufgabe, physikalische Größenangaben und andere Zahlworte verstehen und äußern zu müssen. Wir beschränken uns hier auf die sprachliche Produktion und das Verstehen von Zahlworten. Um nicht sämtliche Zahlwörter in einer Datenbank (dem mentalen Lexikon) vorhalten zu müssen, sollen morphologische Komposita durch eine geeignete Grammatik ableitbar sein. Wir kodieren sprachliche Zeichen als geordnete Tripel von Exponenten, syntaktischen Typen und Semantiken. Der syntaktische Typ wird durch eine minimalistische Grammatik (MG) beschrieben. Für die Semantik verwenden wir den Lambda–Kalkül der arithmetischen Termalgebra. Auf dem MG–Lexikon operieren syntaktische Erzeugungsfunktionen, die durch linguistische Inferenzregeln vorgegeben sind. Sprachproduktion und Sprachverstehen lassen sich dann durch einen bidirektionalen Utterance–Meaning–Transducer (UMT) beschreiben. Bei der Sprachproduktion gehen wir von einer semantischen Repräsentation einer Zahl als arithmetische Termstruktur aus. Durch Datenbankabfrage wird daraus eine Zeichenfolge, welche die passenden Lexikon–Einträge enthält. Daraus berechnet der UMT die Morphologie der Äußerung. Beim Sprachverstehen wird zur Eingabe die abgefragte Zeichenfolge auf dem Stack eines Priority–Queue–Parsers abgespeichert. Durch Anwendung der minimalistischen Inferenzregeln erzeugt der UMT dann schrittweise die semantische Termstruktur.

1 Motivation und Problemstellung

Sprachgesteuerte Benutzerschnittstellen stehen oftmals vor der Aufgabe, physikalische Größenangaben und andere Zahlworte verstehen und äußern zu müssen [1]. So könnte ein Benutzer seiner kognitiven Heizungssteuerung mitteilen wollen: „heize auf 22,5 Grad“; während diese gegebenenfalls eine Sensorerfassung zu kommunizieren hätte: „die gegenwärtige Raumtemperatur beträgt 18,3 Grad“.

Zahlworte sind ein interessanter Forschungsgegenstand der kognitiven Linguistik und der Sprachtechnologie, weil sie einzelsprachlich sehr unterschiedlich realisiert sein können, aber trotzdem eine klar definierte Semantik besitzen. Klassische Beispiele bieten die verschiedenen Morphemstellungen im Deutschen („zweiundvierzig“ $= 2 + 40$) und im Englischen („fourty-two“ $= 40; 2$) oder unterschiedliche Basissysteme im Deutschen („achtzig“ $= 8 \times 10$) und im Französischen („quatre-vingts“ $= 4 \times 20$) [2].

Wir beschränken uns hier auf sprachliche Produktion und Verstehen von Zahlworten wie „ein“, „zwei“, „zehn“ oder „zweiundvierzig“, die linguistisch als (unbestimmte) Artikel aufzufassen sind, wie die Gegenüberstellungen von „der Hund“, „ein Hund“, oder „zweiundvierzig Hunde“ ergibt. Um nicht sämtliche Zahlwörter in einer Datenbank (dem mentalen Lexikon)

vorhalten zu müssen, sollen morphologische Komposita wie „zweiundvierzig“ durch eine geeignete Grammatik ableitbar sein, die einerseits der komplexen Morphologie von Zahlworten in einzelnen Sprachen gerecht wird und andererseits ungrammatische Wortbildungen wie „zweizig“ im Deutschen oder „twoty“ im Englischen auszuschließen vermag [2].

2 Lösungsansatz

Unser Lösungsansatz für die Beschreibung von Zahlwortgrammatik und Zahlensemantik kombiniert Ansätze aus der Computerlinguistik, der formalen Logik und der abstrakten Algebra.

2.1 Zahlwortsemantik

Die Zahlensemantik eines Stellenwertsystems, wie z. B. des Dezimalsystems, besteht darin, eine Zahl, z. B. 42, als Summe von Vielfachen der Potenzen einer Basis g (z. B. $g = 10$ im Dezimalsystem) darzustellen:

$$42 = 4 \times 10 + 2 \times 1 = \sum_{k=1}^{n} a_k g^{k-1} \,. \tag{1}$$

Die Schreibweise (1) kann direkt in eine baumartige Termstruktur überführt werden.

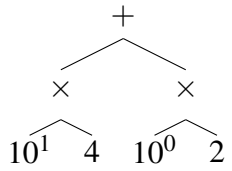

Abbildung 1 – Arithmetischer Termbaum für 42.

Mit den binären Operatoren $+(x,y) = x+y$ und $\times(x,y) = x \times y$ ergibt sich ein Ausdruck der arithmetischen Termalgebra [3]

$$+(\times(10^1,4),\times(10^0,2)))\,, \tag{2}$$

den wir als *Bedeutung* des Zahlwortes „zweiundvierzig“ auffassen wollen. Um mit Bedeutungen rechnen zu können, verwenden wir den logischen Lambda–Kalkül der kompositionalen Semantik und schreiben die binären Operatoren $+, \times$ in der Schönfinkel–Darstellung, also

$$+(x,y) \;=\; +(y)(x) = x+y \tag{3}$$
$$\times(x,y) \;=\; \times(y)(x) = x \times y\,. \tag{4}$$

Dadurch erhält z. B. das Morphem „und“ in „zweiundvierzig“ die funktionale Bedeutung $\lambda y.\lambda x.+(y)(x)$, die nacheinander auf zwei Argumente, 2 und 40, angewandt, den Term

$$[((\lambda y.\lambda x.+(y)(x))(40)](2) = ((\lambda x.+(40)(x))(2) = +(40)(2) \tag{5}$$

ergibt.

2.2 Minimalistische Zahlwortgrammatik

Nach Kracht [3] ist ein *Zeichen* ein Triple (e,t,s) bestehend aus seinem *Exponenten* e, seinem (syntaktischen) *Typ* t und seiner *Semantik* s. Für die Typinformation verwenden wir Stablers minimalistische Grammatik (MG) [4] in der Kodifizierung [5].

Eine MG besteht aus einer Datenbank, dem *mentalen Lexikon*, in welchem Zeichen als Listen syntaktischer, phonetischer und semantischer Merkmale kodiert sind, und zwei strukturerzeugenden Funktionen, namens „merge“ und „move“. Syntaktische Merkmale im Lexikon sind z. B. die *Grundkategorien* `d`: Determinierer (Artikel), `v`: Verb, und `n`: Substantiv. Grundkategorien sind syntaktische Köpfe, die andere Kategorien als Komplemente oder Adjunkte selegieren. Dies wird durch *Selektor*–Kategorien `=x` ausgedrückt, die mit den *selektierten* Kategorien `x` durch „merge“ verschmolzen werden. Außerdem unterscheidet man *Lizensierer* `+f` und *Lizenznehmer* `-f`, die die Bewegung von Maximalprojektionen durch „move“ triggern. In der Standard–Formulierung der MG ist die Reihenfolge von Grundkategorien, Selektoren, Lizensierern und Lizenznehmern in der lexikalischen Merkmalsliste durch reguläre Ausdrücke festgelegt [4, 5].

Seien s,t Exponenten (Sequenzen von Morphemen), φ,ψ semantische Ausdrücke im Lambda–Kalkül [6], f ein Merkmal, γ,δ Merkmalslisten, $\cdot \in \{::,:\}$ und α_i,β_j Zeichenketten (i,j,k,l natürliche Zahlen), dann bilden z. B. $(s,::f\gamma,\varphi)$ und $(t,\cdot f,\psi)$ zwei Zeichen im obigen Sinne. Eine Liste von Zeichen wird als *Ausdruck* bezeichnet. Das erste Zeichen eines Ausdrucks heißt dessen *Kopf* und steuert den Strukturaufbau durch die beiden Funktionen „merge“ und „move“.

Die MG–Funktion „merge“ ist definiert durch die Schlussschemata:

$$\frac{(s,::f\gamma,\varphi) \qquad (t,\cdot f,\psi),\alpha_1,\ldots,\alpha_k}{(st,:\gamma,\varphi\psi),\alpha_1,\ldots,\alpha_k}\ \text{merge-1} \tag{6}$$

$$\frac{(s,:f\gamma,\varphi),\alpha_1,\ldots,\alpha_k \qquad (t,\cdot f,\psi),\beta_1,\ldots,\beta_l}{(ts,:\gamma,\varphi\psi),\alpha_1,\ldots,\alpha_k,\beta_1,\ldots,\beta_l}\ \text{merge-2} \tag{7}$$

$$\frac{(s,\cdot=f\gamma,\varphi),\alpha_1,\ldots,\alpha_k \qquad (t,\cdot f\delta,\psi),\beta_1,\ldots,\beta_l}{(s,:\gamma,\varphi),\alpha_1,\ldots,\alpha_k,(t:\delta,\psi),\beta_1,\ldots,\beta_l}\ \text{merge-3}\,. \tag{8}$$

Entsprechend ist „move“ gegeben durch:

$$\frac{(s,:+f\gamma,\varphi),\alpha_1,\ldots,\alpha_{i-1},(t,:-f,\psi),\alpha_{i+1},\ldots,\alpha_k}{(ts,:\gamma,\varphi\psi),\alpha_1,\ldots,\alpha_{i-1},\alpha_{i+1},\ldots,\alpha_k}\ \text{move-1} \tag{9}$$

$$\frac{(s,:+f\gamma,\varphi),\alpha_1,\ldots,\alpha_{i-1},(t,:-f\delta,\psi),\alpha_{i+1},\ldots,\alpha_k}{(s,:\gamma,\varphi),\alpha_1,\ldots,\alpha_{i-1},(t,:\delta,\psi),\alpha_{i+1},\ldots,\alpha_k}\ \text{move-2}\,, \tag{10}$$

wobei nur ein Zeichen mit Lizenznehmer $-f$ in der durch den Kopf $+f$ lizensierten Kette vorkommen darf. Durch diesen „shortest movement constraint“ (SMC) werden syntaktische Lokalitätsanforderungen erfüllt [4, 5].

Wir konstruieren eine Zahlwort–MG, welche die intendierte Baumstruktur aus Abb. 1 nachbildet. Als erstes legen wir die Grundkategorie von Zahlwörtern fest. Wie oben bereits festgestellt, handelt es sich semantisch gesehen um *Quantoren*, die syntaktisch als Artikel, also von der Grundkategorie `d`, anzusehen sind. Desweiteren beachten wir die semantische Struktur der arithmetischen Termalgebra aus Abb. 1: Zehner selegieren Einer, u.s.w. Indem wir diese linguistischen Versatzstücke zusammenbringen, ergibt sich eine Spielzeugversion eines minimalistischen Lexikons in Tab. 1.

(zwei, :: `d`, 2)	(vier, :: `d -k`, 4)	(zig, :: `=d +k d`, $\lambda x.\times(10^1)(x)$)
(und, :: `=d =d c`, $\lambda y.\lambda x.+(y)(x)$)	(ε, :: `=c d`, ε)	

Tabelle 1 – Minimalistisches Lexikon für einige deutsche Zahlwörter.

Tabelle 1 beschreibt eine Datenbank, in der u.a. das elementare Zahlwort „zwei“ enthalten ist; dies ist der Exponent des Zeichens (zwei, :: `d`, 2) an 1. Stelle. An der 2. Stelle findet sich die

syntaktische Merkmalskette :: d, wobei „::“ auf ein lexikalisches Zeichen verweist (später abgeleitete Zeichen tragen stattdessen das Merkmal „:“), darauf folgt die Grundkategorie d, also Artikel. Das Zeichen $(\text{vier}, :: \texttt{d -k}, 4)$ ist ähnlich strukturiert, weisst allerdings noch das zusätzliche Merkmal -k, also einen Lizenznehmer, auf. Das Zeichen $(\text{zig}, :: \texttt{=d +k d}, \lambda x. \times (10^1)(x))$ steht semantisch für *Zehner* und wird durch einen Multiplikationsoperator $\lambda x. \times (10^1)(x)$, also die Funktion $x \mapsto 10^1 \times x$ ausgedrückt. Seine Syntax schreibt vor, dass „zig“ ein Zahlwort selegiert (=d), daraufhin eine Bewegung lizenziert (+k) und schließlich selbst von der Grundkategorie d ist. Das Zeichen $(\text{und}, :: \texttt{=d =d c}, \lambda y.\lambda x. + (y)(x))$ ist ein Additionsoperator $(x, y) \mapsto x + y$, welcher nacheinander zwei Zahlwort–Argumente selegiert (=d) und vom Typ c (Konjunktion) ist. Der letzte Lexikon–Eintrag $(\varepsilon, :: \texttt{=c d}, \varepsilon)$ ist phonetisch und semantisch leer und drückt lediglich eine syntaktische Typumwandlung von Konjunktion nach Artikel aus.

Neben den elementaren Zahlworten „zwei“ und „vier“, enthält die von der MG Tab. 1 erzeugte Sprache auch „vierzig“ und „zweiundvierzig“, wie Ableitung (11) zeigt.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\text{zig}, :: \texttt{=d +k d}, \lambda x. \times (10^1)(x)) \quad (\text{vier}, :: \texttt{d -k}, 4)}{(\text{zig}, : \texttt{+k d}, \lambda x. \times (10^1)(x))(\text{vier}, : \texttt{-k}, 4)}\,\text{merge-3}}{(\text{vierzig}, : \texttt{d}, \times(10^1)(4))}\,\text{move-1}}{(\text{undvierzig}, : \texttt{=d c}, \lambda x. + (\times(10^1)(4))(x))}\,\text{merge-1}}{(\text{zweiundvierzig}, : \texttt{c}, +(\times(10^1)(4))(2))}\,\text{merge-2}}{(\text{zweiundvierzig}, : \texttt{d}, +(\times(10^1)(4))(2))}\,\text{merge-1}\,. \tag{11}$$

Hingegen werden Komposita wie „zweizig“ als ungrammatisch ausgeschlossen,[1] weil der Lizenznehmer -k im Lexikon–Eintrag von „zwei“ nicht vorkommt.

Zum Vergleich konstruieren wir eine analoge Spielzeug–MG für das Englische:

$(\text{two}, :: \texttt{d}, 2)$	$(\text{four}, :: \texttt{d -k}, 4)$	$(\text{ty}, :: \texttt{=d +k d -l}, \lambda x. \times (10^1)(x))$
$(\varepsilon, :: \texttt{=d =d c}, \lambda y.\lambda x. + (y)(x))$	$(\varepsilon, :: \texttt{=c +l d}, \varepsilon)$	

Tabelle 2 – Minimalistisches Lexikon für einige englische Zahlwörter.

Die unterschiedliche Wortstellung zum Deutschen wird hier durch ein zusätzliches Lizensierer / Lizenznehmer–Paar +l/-l modelliert.

[1] Leider würde in der MG Tab. 1 auch „vierundvierzig“ ausgeschlossen. Dies ließe sich am einfachsten durch Aufnahme eines zweiten Lexikon–Eintrags für vier Einer $(\text{vier}, :: \texttt{d}, 4)$ beheben.

Die entsprechende Ableitung lautet fürs Englische:

$$\dfrac{(\text{ty},::\texttt{=d +k d -l},\lambda x.\times(10^1)(x)) \qquad (\text{four},::\texttt{d -k},4)}{(\text{ty},:\texttt{+k d -l},\lambda x.\times(10^1)(x))(\text{four},:\texttt{-k},4)}\ \text{merge-3}$$

$$\dfrac{(\text{ty},:\texttt{+k d -l},\lambda x.\times(10^1)(x))(\text{four},:\texttt{-k},4)}{(\text{fourty},:\texttt{d -l},\times(10^1)(4))}\ \text{move-1}$$

$$\dfrac{(\varepsilon,::\texttt{=d =d c},\lambda y.\lambda x.+(y)(x)) \qquad (\text{fourty},:\texttt{d -l},\times(10^1)(4))}{(\varepsilon,:\texttt{=d c},\lambda y.\lambda x.+(y)(x))(\text{fourty},:\texttt{-l},\times(10^1)(4))}\ \text{merge-3}$$

$$\dfrac{(\varepsilon,:\texttt{=d c},\lambda y.\lambda x.+(y)(x))(\text{fourty},:\texttt{-l},\times(10^1)(4)) \qquad (\text{two},::\texttt{d},2)}{(\text{two},:\texttt{c},\lambda x.+(2)(x))(\text{fourty},:\texttt{-l},\times(10^1)(4))}\ \text{merge-2}$$

$$\dfrac{(\varepsilon,::\texttt{=c +l d},\varepsilon) \qquad (\text{two},:\texttt{c},\lambda x.+(2)(x))(\text{fourty},:\texttt{-l},\times(10^1)(4))}{(\text{two},:\texttt{+l d},\lambda x.+(2)(x))(\text{fourty},:\texttt{-l},\times(10^1)(4))}\ \text{merge-1}$$

$$\dfrac{(\text{two},:\texttt{+l d},\lambda x.+(2)(x))(\text{fourty},:\texttt{-l},\times(10^1)(4))}{(\text{fourtytwo},:\texttt{d},+(2)(\times(10^1)(4)))}\ \text{move-1}\,. \tag{12}$$

3 Bidirektionaler Utterance–Meaning–Transducer

Mit der oben vorgestellten Kodierung von Syntax und Semantik kann ein bidrektionaler Utterance–Meaning–Transducer (UMT) implementiert werden.

3.1 Sprachproduktion

Bei der Sprachproduktion gehen wir von einer mentalen semantischen Repräsentation in Form einer algebraischen Termstruktur Abb. 1 aus. Im ersten Schritt werden die Terme durch eine Datenbankabfrage mit dem mentalen Lexikon abgeglichen. Dadurch erhalten wir für die deutsche MG Tab. 1 den annotierten Baum Abb. 2, der den syntaktischen Strukturaufbau steuert, indem der Pfad von der linken Ecke „zig“ über „vier“, „und“ bis zur „zwei“ durchlaufen wird.

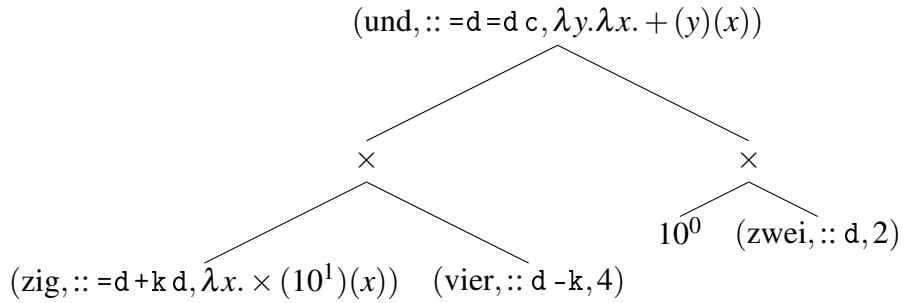

Abbildung 2 – Lexikalisch annotierter Termbaum für das mentale Konzept *42*.

Die dadurch entstehende syntaktische Ableitung (11) ergibt am Ende den Exponenten der Äußerung „zweiundvierzig“.

3.2 Sprachverstehen

Der erste Schritt beim Sprachverstehen ist die lexematische Segmentierung der Äußerung „zweiundvierzig“ in „zwei–und–vier–zig“. Zu diesen Exponenten werden durch Datenbankabfrage die vollständigen Zeichen $\{(\text{zwei},::\texttt{d},2);(\text{vier},::\texttt{d -k},4);(\text{zig},::\texttt{=d +k d},\lambda x.\times(10^1)(x));(\text{und},::\texttt{=d =d c},\lambda y.\lambda x.+(y)(x))\}$ aus dem mentalen Lexikon Tab. 1 assoziiert und auf ein Speicherband

geschrieben. Ein minimalistischer Bottom–Up–Parser arbeitet den Speicher dann entweder parallel in Form einer Chart ab [7] oder verwendet ein serielles Sortier–Orakel für eine Priority–Queue [8, 9]. Auch dabei wird die Ableitung (11) rekonstriert, aus deren letzten Schritt diesmal die Termsemantik $+(\times(10^1)(4))(2)$ ausgelesen wird.

4 Diskussion und Ausblick

Der hier vorgeschlagene birektionale UMT ist ein erster Schritt zu einer elektronischen Syntax–Semantik–Verarbeitung (ESSV) natürlicher Sprache in kognitiven Benutzerschnittstellen. Für einen Proof–of–Concept haben wir eine beschränkte Anwendung auf Zahlworte ausgewählt, die wir durch minimalistische Grammatiken (MG) und arithmetische Termsemantik beschreiben.

Zukünftige Anwendungen unseres UMT sollen weitere Bereiche der Sprache und verschiedene Einzelsprachen abdecken. Dafür werden wir eine Semantik der Merkmal–Werte–Relationen (MWR) [1, 10, 11] benötigen, die ggf. durch die entsprechende Attribut–Werte–Logik (AVL) von Johnson [12] auszuarbeiten ist, um auch hier kompositionale Verarbeitung durch den Lambda–Kalkül zu ermöglichen.

Ein Problem bei der Sprachverarbeitung besteht darin, dass die vorgeschlagene Bottom–Up–Architektur weder inkrementell noch prediktiv verfährt. Das heißt, die gesamte zu verarbeitende Äußerung muss dem Prozessor vollständig vorliegen, bevor die syntaktische Analyse begonnen werden kann. Der zur Zeit beste Lösungsvorschlag ist der Top–Down–Recognizer von Stabler [8]. Allerdings erfordert dieser die vorherige Kompilation des MG–Lexikons in eine multiple kontextfreie Phrasenstrukturgrammatik (MCFG), die bislang ebenfalls nur in einem Bottom–Up–Verfahren zu gewinnen ist.

Eine weitere wichtige Verbesserung wäre der korrekte Umgang mit Ambiguität, z. B. durch den Einsatz stochastischer Grammatiken und Bayesianischer Verarbeitungsansätze [13]. Dafür bieten sich die von beim Graben und Gerth [9] vorgeschlagenen harmonischen minimalistischen Grammatiken (HMG) an.

Und schließlich sollte unser UMT in der Lage sein, sein minimalistisches Lexikon eigenständig zu erwerben [14], anstatt von Computerlinguisten in mühevollster Kleinstarbeit definiert werden zu müssen. Dies könnte eine interessante Anwendung für Reinforcement–Lernen in der Sprachtechnologie werden.

Literatur

[1] TSCHÖPE, C., F. DUCKHORN, M. HUBER, W. MEYER, und M. WOLFF: *A cognitive user interface for a multi-modal human-machine interaction*. In A. KARPOV, O. JOKISCH, und R. POTAPOVA (Hrsg.), *Speech and Computer*, S. 707 – 717. Springer, Cham, 2018.

[2] HURFORD, J. R.: *Numeral systems*. In *International Encyclopedia of the Social & Behavioral Sciences*, S. 10756 – 10761. Elsevier, 2001.

[3] KRACHT, M.: *The Mathematics of Language*. Nr. 63 in Studies in Generative Grammar. Mouton de Gruyter, Berlin, 2003.

[4] STABLER, E. P.: *Derivational minimalism*. In C. RETORÉ (Hrsg.), *Logical Aspects of Computational Linguistics*, Bd. 1328 d. Reihe *Lecture Notes in Computer Science*, S. 68 – 95. Springer, New York, 1997.

[5] STABLER, E. P. und E. L. KEENAN: *Structural similarity within and among languages*. *Theoretical Computer Science*, 293, S. 345 – 363, 2003.

[6] NIYOGI, S.: *A minimalist implementation of verb subcategorization*. In *Proceedings of the Seventh International Workshop on Parsing Technologies (IWPT-2001)*. 2001.

[7] HARKEMA, H.: *A recognizer for minimalist grammars*. In *Proceedings of the Sixth International Workshop on Parsing Technologies (IWPT 2000*, S. 111 – 122. 2000.

[8] STABLER, E. P.: *Top-down recognizers for MCFGs and MGs*. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, S. 39 – 48. Association for Computational Linguistics, Portland, Oregon, USA, 2011.

[9] BEIM GRABEN, P. und S. GERTH: *Geometric representations for minimalist grammars*. *Journal of Logic, Language and Information*, 21(4), S. 393 – 432, 2012.

[10] HUBER, M., C. KÖLBL, R. LORENZ, R. RÖMER, und G. WIRSCHING: *Semantische Dialogmodellierung mit gewichteten Merkmal-Werte-Relationen*. S. 25 – 32, 2009.

[11] WOLFF, M., W. MEYER, und R. RÖMER: *Modellierung von Bewältigungsverhalten mit Merkmal-Werte-Relationen*. In G. WIRSCHING (Hrsg.), *Tagungsband, 26. Konferenz Elektronische Sprachsignalverarbeitung*, S. 224 – 231. 2015.

[12] JOHNSON, M.: *Attribute-Value Logic and the Theory of Grammar*. Nr. 16 in CSLI Lecture Notes. CSLI, Stanford (CA), 1988.

[13] MAINGUY, T.: *A probabilistic top-down parser for minimalist grammars*. ArXiv cs.CL 1010.1826, 2010.

[14] STABLER, E. P., T. C. COLLIER, G. M. KOBELE, Y. LEE, Y. LIN, J. RIGGLE, Y. YAO, und C. E. TAYLOR: *The learning and emergence of mildly context sensitive languages*. In W. BANZHAF ET AL. (Hrsg.), *Advances in Artificial Life*, Bd. 2801 d. Reihe *Lecture Notes in Computer Science*, S. 525 – 534. Springer, Berlin, 2003.