Maschinelle Übersetzung mit synchronen TAGs

Im Seminar „Effizientes Parsen und Generierien mit
Baumadjunktionsgrammatiken (TAGs)“
Wintersemester 1996/97

Prof. Wolfgang Wahlster, Dr. Tilman Becker

Ralf Engel

29. Januar 1997

Inhaltsverzeichnis

1 Aufbau

Im Rahmen des Seminars „Effizientes Parsen und Generierien mit Baumadjunktionsgrammatiken (TAGs)“ stellt diese Ausarbeitung vor, wie TAGs zur maschinellen Übersetzung natürlicher Sprache eingesetzt werden können. Dabei werden synchrone TAGs (STAGs) verwendet, die synchron zum normalen Parsebaum einen zweiten TAG-Baum für die Generierung in der Zielsprache aufbauen.

Nach einer kurzen Einführung, wird der Artikel „Using Lexicalized Tags for Machine Translation“ von Anne Abeillé, Yves Schabes und Aravind K. Joshi vorgestellt, der 1990 erschienen ist [ASJ90]. Dieser enthält eine ganze Reihe von Beispielen in Form von Satzpaaren, die illustieren sollen, wie ein Übersetzungssystem für Französisch und Englisch mit synchronen Tags funktionieren könnte. Im Vordergrund steht dabei, wie bestimmte Probleme, die bei der Übersetzung auftreten, mit STAGs behandelt werden können. Kaum eingegangen wird dagegen darauf, wie ein Parser für STAGs realisisert werden könnte.

Eher theoretische Überlegungen zur Mächtigkeit und Parsbarkeit von synchronen TAGs stellt Stuart M. Shieber in seinem Artikel „Restricting the Weak-Generative Capacity of Synchronous Tree-Adjoining Grammars“ vor [Shi94]. Er zeigt dabei Probleme auf, die mit der bisherigen formalen Definition von STAGs auftreten und schlägt eine leicht abgewandelte Definition vor, bei der diese Probleme nicht mehr auftreten. Allerdings schränkt diese neue Definition die praktische Verwendbarkeit etwas ein, weshalb er am Ende seines Artikel noch mögliche Lösungsansätze vorstellt.

Anschließend wird noch ein Ansatz von Karin Harbusch und Peter Poller vorgestellt, um ein Teil der von S. Shieber erörterten Probleme zu lösen. Mit Hilfe von dynamischen Links, werden die Möglichkeiten von STAGs erweitert [HP96a]. Karin Harbusch und Peter Poller haben im übrigen im Rahmen des Verbmobilprojekts eine Implementierung für STAG-basierte Übersetzung geschrieben [HP96b].

Abschließend folgt noch ein kurzer Vergleich mit dem Shake-and-Bake Übersetzungsverfahren, das nicht wie STAGs direkt mit den Grammatikbäumen arbeitet, sondern lediglich die syntaktische Beziehungen der einzelnen Wörter (wie Subjekt- oder Objektbeziehungen) betrachtet.

2 Einführung

Maschinelle Übersetzungssysteme haben im Prinzip alle einen ähnlichen strukturellen Aufbau. Erst wird der Satz in der Quellsprache analysiert. Das Ergebnis dieser Analyse wird in eine Datenstruktur transferiert, die zur

Erzeugung des Satzes in der Zielsprache verwendet wird. Übersetzungssysteme können also grob in drei Module eingeteilt werden: Analysekomponente, Transferkomponente und Generierungskomponente.

Die einzelnen Übersetzungssysteme unterscheiden sich nun im Aufwand, der bei der Analyse betrieben wird, d.h. welchen Abstraktionsgrad die Struktur erreicht, die von der Analysekomponente generiert wird. So reicht das Spektrum von einfacher grammatikalischer Analyse des Ausgangssatzes über sprachabhänige semantische Repräsentation des Satzes (wie z.B. bei Verbmobil) bis hin zur sprachunabhänigen semantischen Repräsentation, auch Interlingua genannt. Im allgemeinen ist der Aufwand, den die Transferkomponente hat umso größer, je einfacher die Analyse ausgefallen ist (natürlich bei gleicher Übersetzungsleistung).

Bei der Übersetzung mittels synchroner TAGs handelt es sich um ein Verfahren, das relativ wenig Aufwand in die Analyse steckt. Es wird lediglich der Satz in der Ausgangssprache mit Hilfe einer TAG-Grammatik geparst. Für jeden adjungierten bzw. substituierten Elementarbaum gibt es nun einen zweiten zugeordneten Elementarbaum in der Zielsprache. Dieser wird dazu verwendet einen zweiten Baum synchron zum Parsebaum aufzubauen. Dies geschieht, indem der zugeordnete zweite Elemetarbaum in den synchron aufgebauten Baum adjungiert bzw. substituiert wird. Die Stelle an dem dieser adjungiert bzw. substituiert wird, wird durch einen Link gekennzeichnet. Wenn dann der Ausgangssatz fertig geparst ist, muß die Generierungskomponente nur noch den synchron aufgebauten zweiten Baum durchlaufen und dabei den Satz in der Zielsprache generieren.

Der Programmieraufwand dafür hält sich in Grenzen, da das Verfahren nicht allzu komplex ist. Allerdings müssen natürlich die Elementarbaumpaare erstellt werden und zusätzlich müssen noch weitere Module, wie z.B. eine Sematikkomponente zur Desambiguierung, vorhanden sein, wenn die Übersetzungsresultate nicht allzu schlecht ausfallen sollen.

3 Synchrone TAGs für die maschinelle Übersetzung

3.1 Grundlagen

Grundlage für den Vorschlag von Abeillé, Schabes und Joshi für ein Übersetzungssystem bilden lexikalische TAGs (LTAGs), die fürs Englische und Französische entwickelt wurden. Da LTAGs sowohl zur Analyse als auch zur Generierung von Sätzen verwendet werden können, reicht es für ein Übersetzungssystem, den Ableitungsbaum, der bei der Analyse des Satzes in der Ausgangssprache entsteht, in einen Ableitungsbaum für die Zielsprache zu transferieren. Die Autoren greifen dabei auf synchrone TAGs (STAGs) zurück, die auch noch eingesetzt werden, um natürlichsprachliche Sätze in eine logische Form zu bringen [SS90]. Inzwischen werden STAGs auch zur Übersetzung vom Koreanischen ins Englische und umgekehrt verwendet.

Eine Beschreibung einer konkreten STAG-Realisierung findet sich unter [Pri97]

STAGs bestehen aus Baumpaaren, die über Links gekoppelt sind, wobei der linke Baum wie ein normaler TAG-Baum zum Parsen verwendet wird, während der rechte Baum synchron dazu den Baum für die Generierung in der Zielsprache aufbaut. In Abbildung 1 sind einige Baumpaare als Beispiel dargetellt. Die gestrichelten Linien zwischen den Bäumen werden als Links bezeichnet.

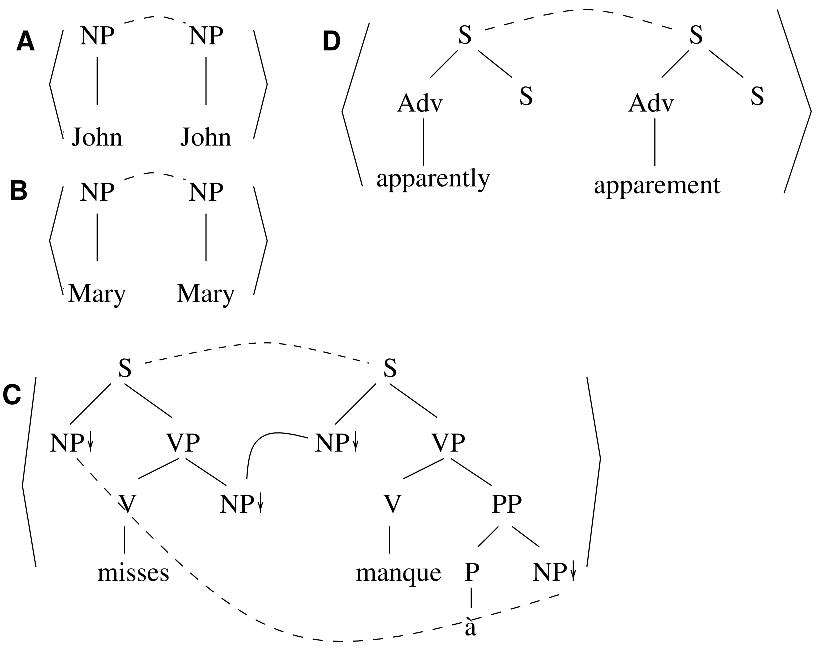

Abbildung 1: Beispiel für Baumpaare, die eine STAG bilden

Als nächstes muß noch definiert werden, wie die einzelnen Baumpaare adjungiert bzw. substiuiert werden können. Ein kleiner, nicht-deterministischer Algorithmus, der diese Aufgabe erfüllt, wird hier angegeben. Dieser Algorithmus dient nur zur Definition von STAGs, das heißt eine konkrete Implementierung kann ein völlig anderes Vorgehen wählen.

1. Wähle ein initiales Baumpaar $I :=< I_L, I_R, \frown_I>$ aus, wobei $\frown_I$ die Relation ist, die die Links zwischen den beiden Bäumen I_L und I_R beschreibt

2. Höre auf oder mache weiter (nur wenn noch ein Link in I vorhanden ist)

3. Wähle einen Link $t_L \frown_S t_R$ aus, wobei t_L und t_R die beiden Endpunkte des Links sind.

4. Wähle ein auxiliares Baumpaar $A :=< A_L, A_R, \frown_A>$ aus, so daß A_L in I_L an der Stelle t_L und A_R in I_R an der Stelle t_R adjungieren bzw. substituieren können

5. Adjungiere bzw. substituiere die ausgewählten Bäume A_L und A_R an den entsprechenden Stellen t_L und t_R und setze I entsprechend neu, wobei der ausgewählte Link $\frown_S$ gelöscht wird. Die restlichen Links bleiben erhalten.
   $I :=< I_L[A_L/t_L], I_R[A_R/t_R], (\frown_I \cup \frown_A) \backslash \frown_S>$, wobei $I[A/t]$ hier die Adjunktion des Baumes A an der Stelle t in I meint.

6. Gehe nach 2.

Der Algorithmus kann auch anhand eines kleinen Beispiels nachvollzogen werden, das in den Abbildungen 2 bis 4 dargestellt ist. Dabei wird auf die Beispielbäume aus Abbildung 1 zurückgegriffen, wobei Baum C als Initialbaum genommen wird.

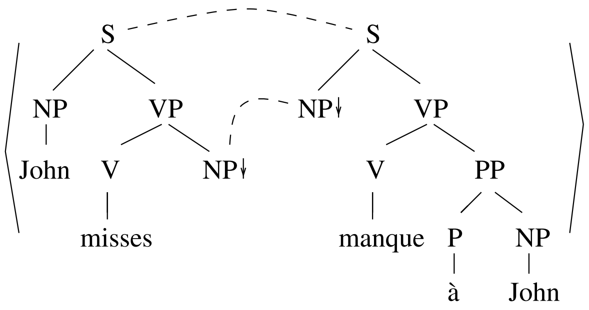

Abbildung 2: In den Initailbaum C wurde der Baum für *John* hineinsubstituiert. Zu beachten ist hierbei, daß der Link zwischen den NPs gelöscht wurde.

Ein Bemerkung noch zu möglichen Ansätzen für eine konkrete Implementierung:
Im Grunde kann wie in der oben gegebenen Definition vorgegangen werden. Aber naheliegender ist es, einen bestehenden TAG-Parser zu nehmen, der einen Ableitungsbaum generiert, und anhand dieses Ableitungbaumes den synchronen Baum aufzubauen. Shieber zeigt allerdings in seinem Artikel, daß dies nicht so einfach geht. Doch darauf wird noch in Abschnitt 4 näher eingegangen.

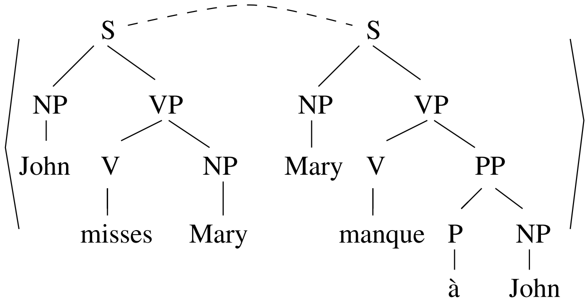

Abbildung 3: Anschließend wird der Baum für *Mary* hineinsubsituiert.

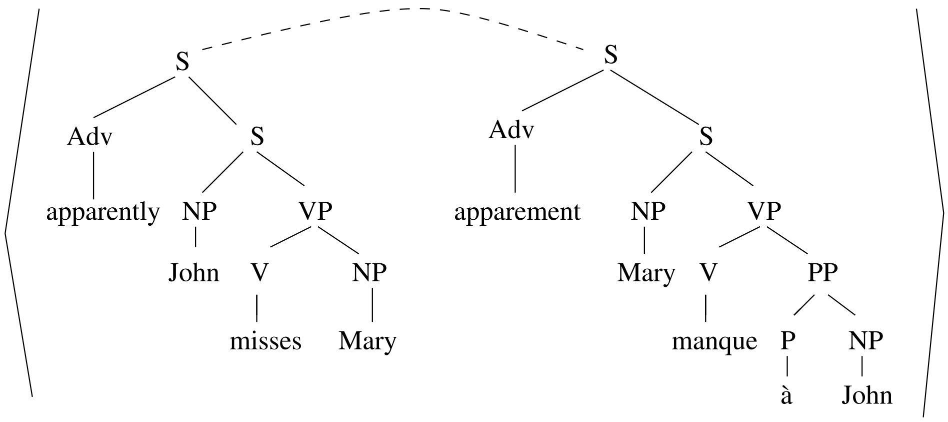

Abbildung 4: Zum Schluß noch eine Adjunktion, wobei der Link zwischen den S-Knoten aus dem Baumpaar D stammt.

Nachdem nun die technischen Grundlagen für synchrone TAGs gelegt sind, werden anhand von Beispielsätzen einige typische Phänomene bzw. Probleme vorgestellt, die bei der Übersetzung vom Englischen ins Französische und umgekehrt auftreten.

3.2 Behandlung von Wortartwechseln

Häufig kommt es vor, daß es für ein Wort in der einen Sprache kein passendes Wort mit der gleichen Wortart in der anderen Sprache gibt. Ein Beispiel hierfür ist das englische Adverb *just*, das im Französischen am besten mit dem Verb *venir de* übersetzt wird. Da beide Wörter in den Grammatiken als Verbadjunktionen realisiert sind, stellt dies kein größeres Problem dar (siehe Abbildung 5). Auf den Fall, daß ein Wort in der einen Sprache gleich mehreren Wörtern bzw. ganzen Satzteilen in der anderen Sprache entspricht, wird im nächsten Abschnitt näher eingegangen. So entspricht dem englischen *hopefully* der französische Satzteil *on espère que* (Abbildung 6).

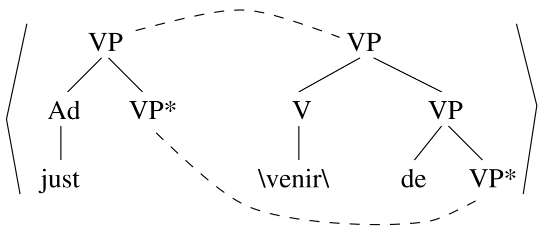

Abbildung 5: Baumpaar für *just* und *venir de*

3.3 Behandlung von Idiomen bzw. zusammengesetzten Ausdrücken

Dadurch daß keine einzelnen Wörter, sondern ganze Bäume transferiert werden, können zusammengesetzte Ausdrücke, wie z.B. *took a walk* ↔ *faire une promenade* oder *to be fond of* ↔ *aimer* recht elegant gehandhabt werden. Die Baumpaare erfassen dabei gleich mehrere Wörter auf einmal (siehe Abbildungen 6, 7,8). Durch die Links können auch noch eingefügte Wörter, wie z.B. Adjektive, richtig behandelt werden. Ein Beispiel hierfür sei *John is very fond of music* ↔ *John aime beaucoup la musique.* oder auch *to give a (weak) cough* ↔ *tousser (faiblement)* (Abbildungen 8,9).

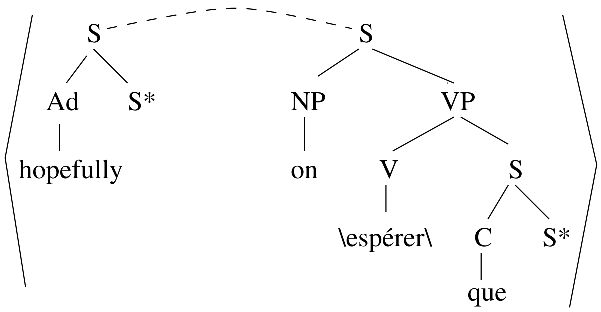

Abbildung 6: Baumpaar für *hopefully* und *on espère que*

3.3.1 Auftretende Probleme bei Idiomen

Leider ist die Übersetzung zusammengesetzter Ausdrücke nicht ganz so unproblematisch, wie die Autoren Glauben machen wollen.

Ein nicht so schwerwiegendes Problem ist (aber erwähnt werden sollte es doch), daß bei zusammengesetzten Ausdrücken oft auch noch ein abgeleiter

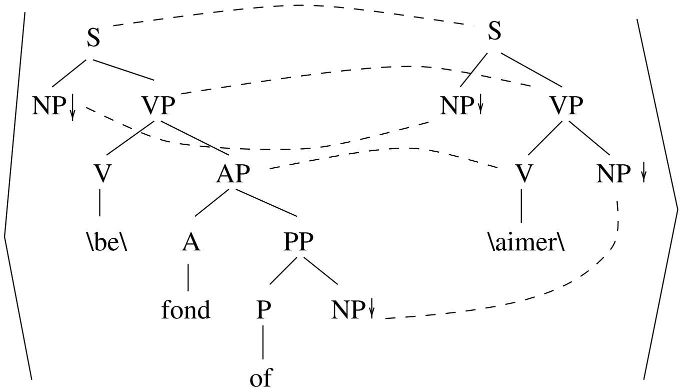

Abbildung 7: Baumpaar für *to be fond of* und *aimer*

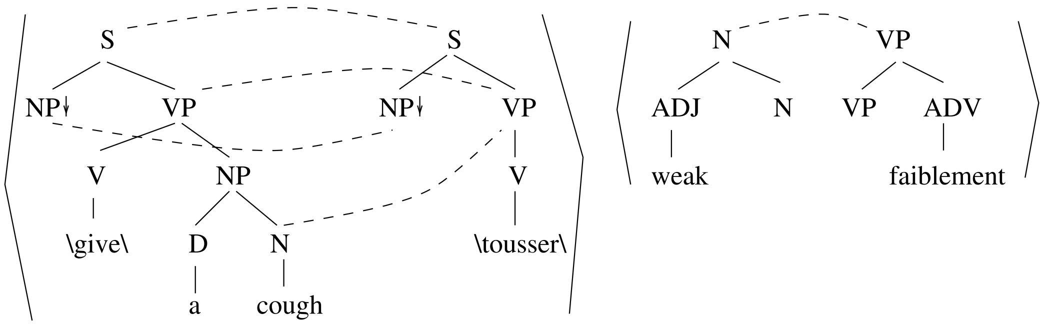

Abbildung 8: Baumpaar für *to give a cough* und *weak*

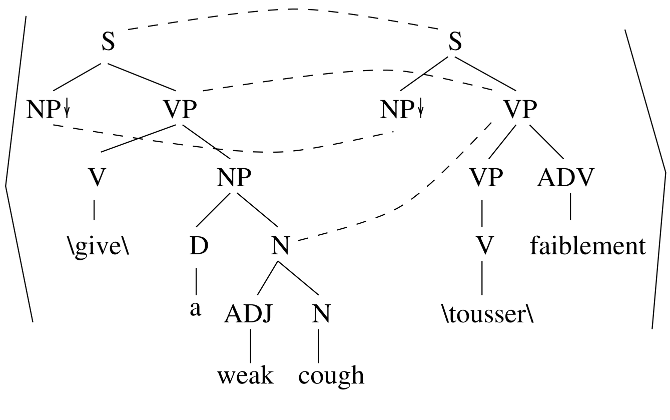

Abbildung 9: zusätzliche Adjektive in Idiomen

Baum entsteht, der von der direkten Wort-für-Wort Übersetzung herrührt. So etwa, wenn es für *give* und *cough* eigene direkte Übersetzungen gibt. Ein

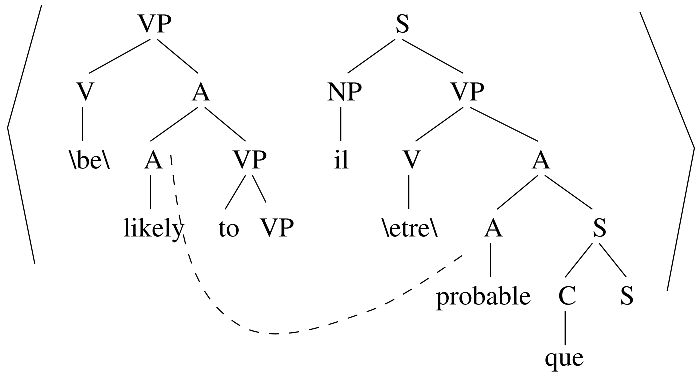

Abbildung 10: Baumpaar für *to be likely to* und *il est probable que*

einfacher Lösungsansatz bei mehreren gleichen abgeleiteten Bäumen ist, die bevorzugt zu nehmen, die weniger Adjunktionen bzw. Substitutionen haben, da diese dann spezifischer sind. Da sowieso eine Komponente vorhanden sein muß, die zwischen den verschiedenen möglichen Übersetzungen entscheidet, kann dies dort integriert werden.

Die Tatsache, daß mehrere Wörter bzw. ganze Idiome von einem Baum abgedeckt werden, kann größere Schwierigkeiten bei der praktischen Implementierung verursachen. Prinzipiell gibt es zwei Möglichkeiten, synchrone TAGs zu implementieren und je nachdem sind die Probleme etwas anders gelagert:

1. Die Baumpaare werden direkt zum Parsen des Ausgangssatzes verwendet. Dadurch daß die ganzen zusammengesetzten Ausdrücke mit ihren Variationen (z.B. geänderte Satzstellung bei Fragesätzen) die Grammatik ziemlich stark vergrößern, dürfte sich die Laufzeit beim Parsen nicht unbedingt zum positiven entwickeln.

2. Der Satz wird mit einer normalen TAG-Grammatik geparst. Die synchronen TAGs werden zum Parsen gar nicht verwendet, sondern der generierte Baum wird sozusagen als Vorlage genommen. Dabei wird ein Ableitungsbaum für die linken Bäume aus dem Baumpaarlexikon ermittelt, der den gleichen Baum erzeugt, den auch die TAG-Grammatik beim Parsen geliefert hat. Dies ist allerdings keine ganz triviale Aufgabe. Mit diesem Ableitungsbaum kann dann der synchrone Baum für die Zielsprache generiert werden. Zu beachten ist hierbei, daß die Bäume im Baumpaarlexikon aus den Bäumen der Grammtik gebildet werden können, d.h. daß z. B. kein Baum der Grammatik nur als Teilbaum in dem Baumpaarlexikon auftreten darf.

Einen anderen Weg den Problemen bei Bäumen, die mehrere Wörter abdecken, zu begegnen, haben Karin Harbusch und Peter Poller in [HP96a] vorgeschlagen. Die Bäume werden dabei in ihre Einzelbäume zerlegt und die Links dürfen dann über Baumpaargrenzen hinausreichen. Sie werden deshalb dynamische Links genannt und im Abschnitt 5 ausführlich beschrieben.

Tücken hat auch das Beispiel mit *to be likely to* $\leftrightarrow$ *il est probable que* (Abbildung 10). Dabei wird davon ausgegangen, daß in den initialen Bäumen (z.B. Abbildung 7) ein Link von VP nach S vorhanden ist (sonst kann ja die Adjunktion gar nicht in beiden Bäumen stattfinden). In den von den Autoren erstellten Beispielen ist das aber nicht der Fall. Der Link kann allerdings problemlos eingetragen werden (am besten geschieht so etwas automatisch).

3.4 Berücksichtigung der Featurestrukturen

Featurestrukturen werden normalerweise bei der Baumpaarauswahl nicht berücksichtigt. Für einige Fälle ist es jedoch nützlich sie trotzdem zu berücksichtigen, so kann bei einigen Wörten nicht zwischen Singular und Plural unterschieden werden (z.B. bei *fish*). Da das Verb u.U. Aufschluß darüber bietet, ob es sich um Singular oder Plural handelt und sich diese Information in der Featurestruktur widerspiegelt, kann ein Eintrag für *fish* z.B. so aussehen: *fish, N [num=pl]* $\leftrightarrow$ *poissons, N* bzw. *fish, N [num=sing]* $\leftrightarrow$ *poisson, N*.

Bis auf die Zeitinformationen werden normalerweise Featurestrukturen bei dem Transfer nicht übertragen, sondern werden beim Aufbau des Zielbaums entsprechend gesetzt. Dies ist nützlich wenn sich z.B. der Numerus bei der Übersetzung ändert. Ein Beispiel hierfür ist das englische *luggage* das im Französischen zu *baggages* wird, das im Plural steht. Das französische Verb bekommt dann die Information über Numerus vom französischen Subjekt und nicht vom englischen Verb.

3.5 Behandlung von Baumfamilien

Statt einzelne Baumpaare zu verwenden, können auch ganze Baumfamilien[1] Baumpaare bilden. Damit wird eine höhere Abstraktionsstufe erreicht, die Bäume müssen nicht mehr einzeln miteinander gelinkt werden.

Konkret sieht das so aus, daß z.B. ein Verb beim Parsen einen Baum liefert. Dieser Baum wird allerdings nicht direkt verwendet, sondern ist einer Baumfamilie zugeordnet. Zu dieser Baumfamilie gibt es nun in der Zielsprache eine Menge synchroner Bäume. Aus dieser Menge werden nur die Bäume

[1] Baumfamilie: Menge aller Bäume, die einer Wortklasse zugeordnet sind

ausgewählt, die die gleichen Satzfeatures wie der Baum in der Ausgangssprache haben. (Satzfeatures beschreiben, ob es sich um einen Fragesatz, Relativsatz, Passivsatz etc. handelt.)

Falls es keinen Baum mit einem Feature gibt, wird dises Feature einfach ignoriert. Damit wird z.B. erreicht, daß der englische Passivsatz *John was given a book by Mary* im Französischen zu dem Aktivsatz *Mary a donné un livre à Jean* wird, da eine entsprechende Passivkonstruktion im Französischen nicht möglich ist.

Ebenso wird bei der Übersetzung eines englischen Fragesatzes das Feature, das beschreibt, ob eine Präposition vorgezogen wird oder nicht, im Französischen ignoriert, da es dort immer vorgezogen wird. So wird aus dem englischen Satz *Who did Mary give a book to ?* im Französischen *A qui Mary a-t-elle donné un livre ?*.

In einigen Fällen ist es auch nötig, Features, die zwar in beiden Sprachen prinzipiell vorhanden sind, aber mit bestimmten nicht Wörtern verwendet werden dürfen, zu ignorieren. So darf der englische Satz *Advantage was taken of this affair by John* in diesem Fall nicht mit *Parti a été tiré de cette affaire par Jean* (obwohl prinzipiell syntaktisch im Französischen möglich), sondern muß mit *Jean a tiré parti de cette affaire* übersetzt werden.

4 Theoretische und prinzipielle Überlegungen von S. Shieber zu STAGs

Nachdem bislang eher praktische Aspekte bei der maschinellen Übersetzung mittels STAGs im Vordergrund standen, weist S. Shieber in seinem Artikel „Restricting the Weak-Generative Capacity of Synchronous Tree-Adjoining Grammars“ [Shi94] auf mehr theoretische und allgemeine Aspekte bei STAGs hin. So zeigt er, daß STAGs mächtiger sind als normale TAGs und beleuchtet Probleme bei einer konkreten Parser-Implementierung. Um diese Probleme zu umgehen stellt er eine etwas andere formale Definition für synchrone TAGs auf. Diese bringt allerdings neue Schwierigkeiten mit sich. Darauf wie diese zum Teil umgangen werden können, geht er noch am Ende seines Artikels ein.

4.1 Mächtigkeit von STAGs

Auf den ersten Blick etwas überraschend ist, daß STAGs mächtiger sind als normale TAGs. Überraschend deshalb, weil nur die linken Bäume der Baumpaare, die ja normale TAG-Bäume darstellen, zum Parsen des Ausgangssatzes verwendet werden. Die größere Mächtigkeit rührt nun daher, daß an dem rechten Baum eines Baumpaares Contraints angebracht werden können, die sich nicht auf den linken Baum übertragen lassen. Abbildungen 11 bis 13 verdeutlichen dies anhand einer STAG, die nur die Sprache $a^n b^n c^n d^n e^n f^n g^n h^n$ mit $n \geq 0$ erkennt. Von normalen TAGs ist nachgewiesen, daß sie das nicht können.

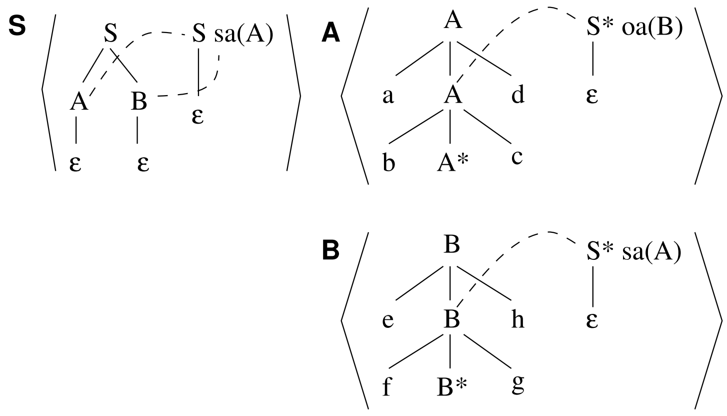

Abbildung 11: Eine STAG für $a^n b^n c^n d^n e^n f^n g^n h^n$

Erklärung der Funktionsweise

Bei der STAG in Abbildung 11 muß, wenn das Baumpaar A adjungiert wird, auch immer Baumpaar B adjungiert werden. Sonst bleibt ein unerfüllter Constraint *oa(B)* zurück. Das Baumpaar B kann nur adjungiert werden, wenn direkt vorher Baumpaar A adjungiert wurde, da sonst der Constraint *sa(B)* an S verletzt werden würde. Dieser Constraint verschwindet nur, wenn das Baumpaar A adjungiert wurde und erscheint wieder beim Adjungieren von Baumpaar B. Abbildung 12 und 13 zeigen die einzelnen Adjunktionsschritte.

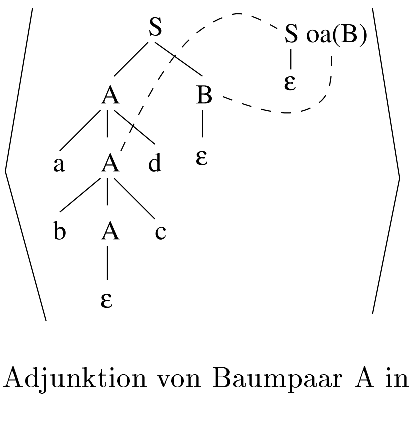

Abbildung 12: Adjunktion von Baumpaar A in Baumpaar S

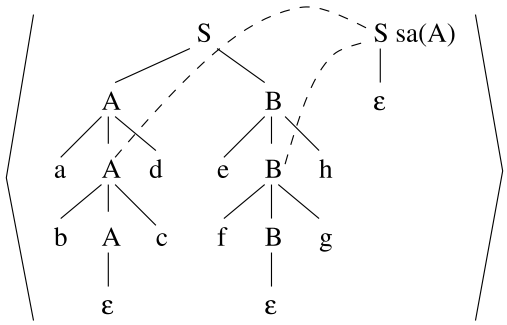

Abbildung 13: Adjunktion von Baumpaar B in Baumpaar S

4.2 Implementierbarkeit

Schwierigkeiten gibt es bei der Implementierung eines Algorithmuses für STAG. Leider ist es nicht damit getan, einen normalen TAG-Parser zu nehmen, den Ausgangssatz zu parsen und mit dem so gewonnen Ableitungsbaum den synchronen Baum aufzubauen mit dem dann der Satz in der Zielsprache generiert werden kann. Das Problem dabei ist, daß es auf die Adjunktionsreihenfolge ankommen kann, die im Ableitungsbaum nicht vermerkt ist. Ein linker Ableitungsbaum kann daher unterschiedliche rechte Ableitungsbäume erzeugen. Dies tritt immer dann auf, wenn zwei Links an der gleichen Stelle im rechten Baum enden und an beide Links adjungiert wird.

Der geschilderte Sachverhalt soll an dem ambigen englischen Satz *John blinked intentionally twice.* veranschaulicht werden. Dieser Satz soll diesmal nicht übersetzt werden, sondern in eine logische Repräsentationssprache gebracht werden (was ja auch eine Art der Übersetzung ist). Das Ergebnis kann entweder *int(twice(blink(john)))* oder *twice(int(blink(john)))* lauten, je nach Adjunktionsreihenfolge (siehe Abbildungen 14 bis 19).

Am Ende seines Artikels stellt Shieber eine recht simple Lösung für dieses Problem bereit, die er allerdings nur auf seine neue Definition anwendet. Anwendbar ist sie auch für die alte Definition, so daß die seiner Meinung nach schlechte Implementierbarkeit eigentlich kein Grund für eine Änderung der Definition wäre. Deshalb stelle ich diesen Lösungsansatz schon jetzt vor.

Der Ausgangssatz wird dabei normal nur mit den linken Bäumen geparst. Wenn dann versucht wird nach diesem Ableitungsbaum den dazu synchronen Baum aufzubauen, stellt sich bei jedem Knoten die Frage in welcher Reihenfolge die Töchter abgearbeitet werden. Die Reihenfolge ist lediglich bei den Töchtern entscheidend die Links haben, die an einem gemeinsamen Punkt enden. Wenn diese Töchter in der Reihenfolge permutiert werden, ergeben sich die verschiedenen möglichen synchronen Bäume.

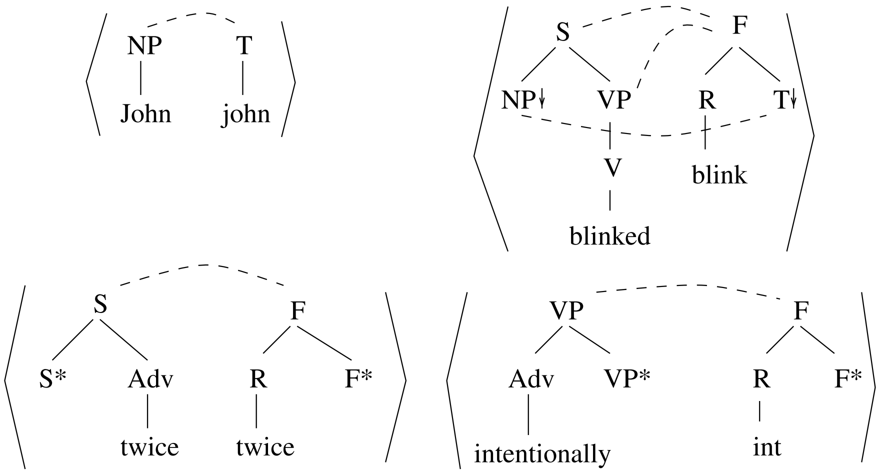

Abbildung 14: Die Elementarbäume für *John, blink,twice* und *intentionally*

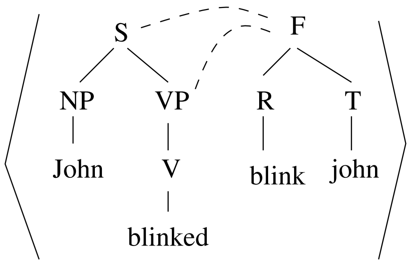

Abbildung 15: Der abgeleitete Baum für *John blinked*

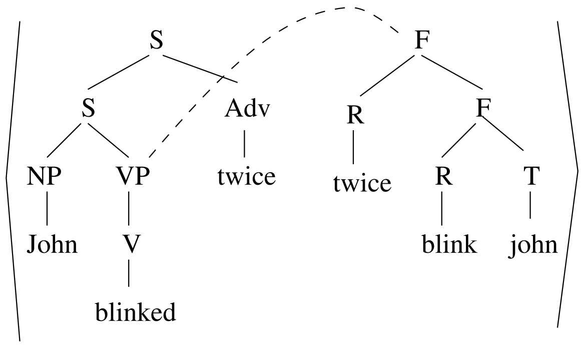

Abbildung 16: Der abgeleitete Baum für *John blinked twice*

4.3 Die Definition von STAGs nach S. Shieber

Die leicht geänderte Definition löst die oben angesprochenen Probleme: STAGs sind damit genauso mächtig wie normale TAGs und die Reihenfolge

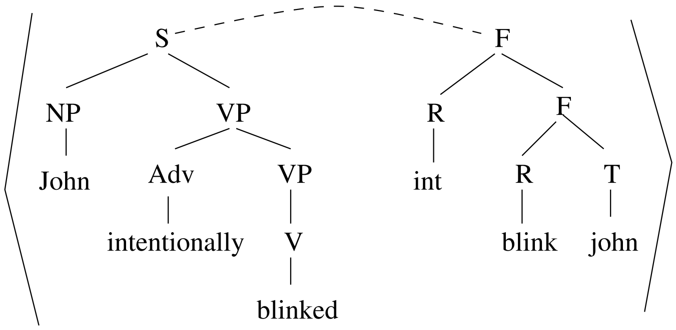

Abbildung 17: Der abgeleitete Baum für *John intentionally blinked*

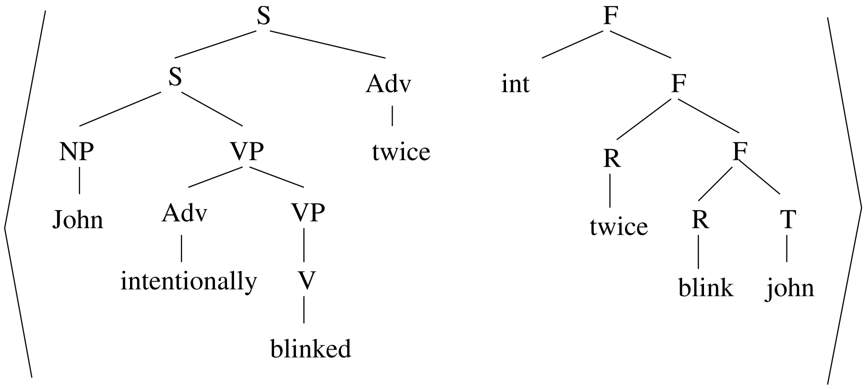

Abbildung 18: Eine Möglichkeit für *John intentionally blinked twice*

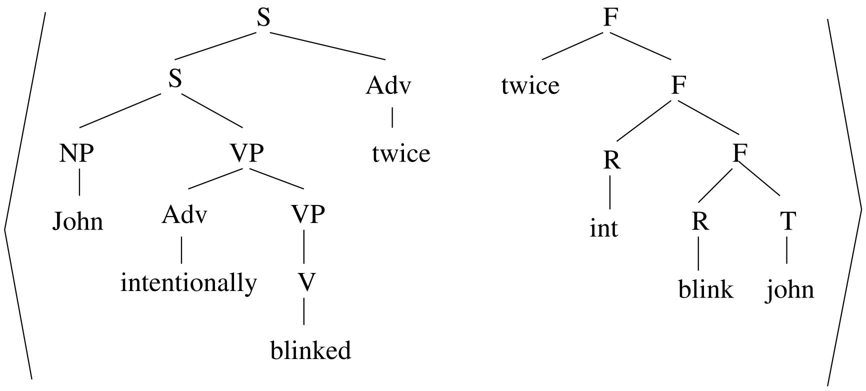

Abbildung 19: Die andere Möglichkeit für *John intentionally blinked twice*

der Adjunktionen spielt keine Rolle mehr. Erreicht wird dies zum einem, in dem der Ableitungsbaum nicht mehr von oben nach unten durchlaufen wird um den synchronen Baum zu erzeugen, sondern von unten nach oben. Desweiteren werden die Adjunktionen an einem Knoten im Ableitungsbaum gleichzeitig durchgeführt. Damit können sich die Constraints bei Adjunktionen nicht mehr gegenseitig überschreiben wie es in dem Beispiel weiter oben passiert ist (siehe Abbildungen 11 bis 13). Die Forderung nach gleichzeitiger Adjunktion verbietet natürlich mehrere Adjunktionen an einer Stelle auszuführen. Es kann zwar weiterhin Links geben, die an der gleichen Stelle enden, aber sie dürfen nicht mehr beide in einem Baum durch Adjunktion benutzt werden.

Nach diesen erklärenden Sätzen nun die formale Definition. G bezeichnet die STAG-Grammatik, G_L ist die TAG-Grammatik, die entsteht, wenn nur die linken Bäume von den Baumpaaren genommen wird, G_R analog dazu nur die rechten. Bei Ableitungsbäumen bezeichnet $parent(n)$ den Vaterknoten von n, $tree(n)$ bezeichnet den Elementarbaum, der durch den Knoten n eingefügt wird. $addr(n)$ steht für die Stelle an der $tree(n)$ in $tree(parent(n))$ eingefügt wird.

1. D_L ist ein Ableitungsbaum unter Verwendung von G_L
2. D_R ist ein Ableitungsbaum unter Verwendung von G_R
3. D_L und D_R sind isomorph, d.h. es gibt eine eineindeutige Funktion f, die die Knoten von D_L so auf Knoten aus D_R abbildet, daß die Dominanz erhalten bleibt $(f(n_l) = n_r \Rightarrow f(parent(n_l)) = parent(n_r))$
4. Links müssen die entsprechenden Stellen, an denen adjungiert wird, verbinden, d.h. wenn $f(n_l) = n_r$ dann muß es ein Baumpaar geben mit $< tree(n_l), tree(n_r), \frown_n >$ in G. Darüberhinaus muß es, wenn n_l einen Mutterknoten hat, auch ein Baumpaar
$< tree(parent(n_l)), tree(parent(n_r)), \frown_P >$
geben, wobei gilt: $addr(n_l) \frown_P addr(n_r)$

Der abgeleitete Baum setzt sich dann folgendermaßen zusammen:

$T(D) = tree(n)[T(D_1)/t_1, \ldots, T(D_k)/t_k]$,

wobei D ein Baum mit dem Wurzelknoten n und den Tochterbäumen $D_1 \ldots D_2$ ist. $I[A_1/t_1, \ldots, A_k/t_k]$ meint hier die gleichzeitige Adjunktion der Bäume A_1 bis A_k an den Stellen t_1 bis t_k in I.

Wie bereits erwähnt wurde, können nun auch keine multiplen Adjunktionen in den rechten Bäumen vorkommen. Deshalb kann der rechte Baum direkt anhand des Ableitungsbaums für den linken Baum generiert werden. Somit kann problemlos ein normaler TAG-Parser verwendet werden und der Transfer bereitet keine Probleme mehr.

Allerdings ist die Einschränkung, daß keine multiplen Adjunktionen vorkommen dürfen, eine recht harte. Diese kann aber, wie auf der vorhergehenden Seite schon beschrieben wurde, relativ einfach beseitigt werden.

S. Shieber weist am Ende seines Artikels noch auf Probleme hin, die dadurch auftreten, daß die Ableitungsbäume für den linken und rechten Baum isomorph sein müssen. Dieses Problem ist sowohl bei der ersten Definition von STAG als auch bei der neuen Definition gegeben (ein Sachverhalt der in seinem Artikel nicht so richtig deutlich wird). Er macht dabei keine konkreten Lösungsvorschläge, sondern verweist nur etwas vage auf Kontrollgrammatiken und auf das Shake-and-Bake-Übersetzungsverfahren, das später noch vorgestellt wird. Mit konkreten Löungsvorschlägen können allerdings Karin Harbusch und Peter Poller aufwarten, die im nächsten Abschnitt vorgestellt werden.

5 Nicht isomorphe STAGs (NIS-TAGs)

Wenn vermieden werden soll, daß die Bäume in den Baumpaaren mehrere Wörter abdecken müssen, reichen normale STAGs zum Übersetzen nicht mehr aus. Die Ableitungsbäume für Ausgangs- und Zielsatz sind dann nicht mehr isomorph, was bei STAGs gefordert wird. Beispiele hierfür sind many-to-one-mappings (z.B. *on espère que* und *hopefully*), Dominanzverschiebungen (z.B. *Le docteur lui soigne les dents* und *The doctor treats his teeth*) und Dominanzumkehrungen (z.B. *Jean monte la rue en courant* und *John runs up the street*).

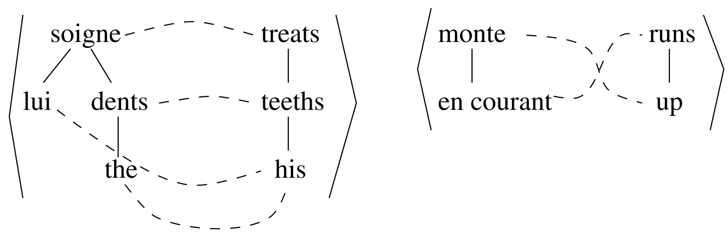

Abbildung 20: nicht isomorphe Ableitungsbäume

5.1 Grundlagen

Die Idee liegt nun darin, die Baumpaare, die mehrere Wörter abdecken, in ihre Elementarbäume zu zerlegen und die Links dann über Baumpaargrenzen hinaus reichen zu lassen, d.h. zwei Knoten, die nicht im selben Baumpaar liegen, können miteinander verbunden werden. Benutzt werden können diese

Links erst dann, wenn beide Baumpaare in dem beim Parsen aufgebauten Baum vorkommen. Ein Beispiel für die Verwendung von dynamischen Links ist *Le docteur lui soigne les dents* und *The doctor treats his teeth*, das in Abbildung 21 dargestellt ist. Zum Vergleich ist in Abbildung 22 noch dargestellt, wie das Beispiel aussehen könnte, wenn ein Baumpaar mehrere Worte abdecken darf.

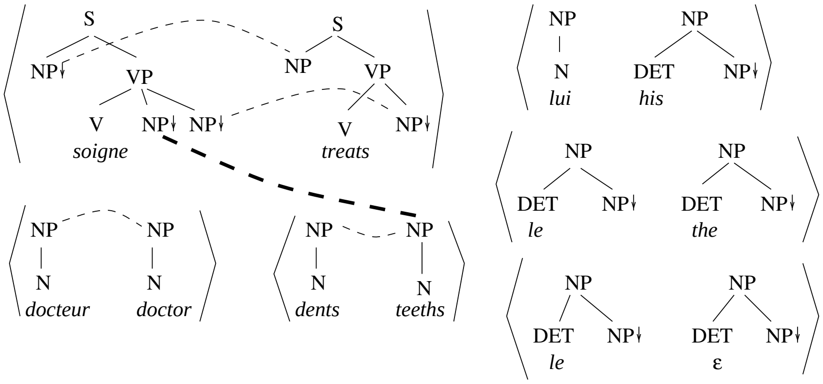

Abbildung 21: Die Elemtarbäume, die zum Übersetzen von *Le docteur lui soigne les dents* nach *The doctor treats his teeth* gebraucht werden, wobei der dynamische Link fett hervorgehoben ist.

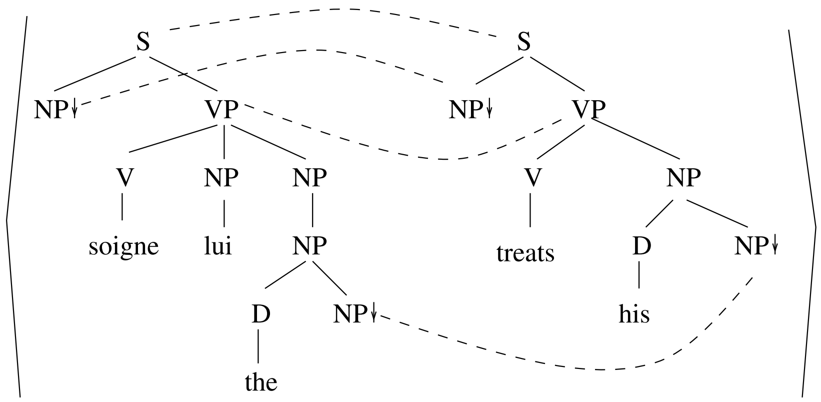

Abbildung 22: Das Beispiel von Abbildung 21 mit normalen STAGs formuliert

Mit dem Beispiel von Abbildung 21 kann nun die Funktionsweise der dynamischen Links erläutert werden. Der Baum für *docteur* kann ganz normal substituiert werden. Wenn der Baum für *dents* substituiert wird, geschieht dies auch noch ganz normal, nur wird anschließend der dynamische Link zu dem Baum für *teeth* aktiviert, da dieser jetzt innerhalb eines Paares abgeleiteter Bäume liegt. Damit kann nun auch der Baum für *lui* substituiert werden. Die synchrone Aktion dazu ist die Adjunktion von dem Baum für *his* in den Baum von *teeth*. Zu beachten ist, daß die Adjunktion von dem Baum für *les* in den Baum für *dents* ebenfalls eine synchrone Adjunktion im Englischen erfordert, obwohl es kein entsprechendes Wort im Englischen gibt. Deshalb wird ein ϵ – Knoten an den Baum für *teeths* adjungiert. Der Generierungsalgorithmus muß so etwas natürlich berücksichtigt. (Übringens fällt auf, daß hier die Adjunktionsreihenfolge wieder eine Rolle spielt, so kann der ϵ – Knoten oberhalb oder unterhalb von dem Baum für *the* landen – im diesem Fall hat das natürlich kein Einfluß aufs Ergebnis).

Zu beachten ist, daß die Elementarbäume in Abbildung 21, die nur für die Übersetzung von *Le docteur lui soigne les dents* gebraucht werden (z.B. das Baumpaar für *lui* und *le*), durch Contraints davon abgehalten werden, in einem anderen Kontext zu adjungieren.

Peter Poller hat seinen Algorithmus für STAGs so erweitert, daß er mit dynamischen Links umgehen kann. Allerdings hat er dann exponetielle Laufzeit. Praktische Erfahrungen bezüglich des Laufzeitverhaltens liegen noch nicht vor.

5.2 Mächtigkeit

Wie bei normalen STAGs, zeigt sich auch hier wieder, daß NIS-TAGs mächtiger sind als TAGs. Abbildung 22 zeigt ein Beispiel dafür, das stark an das Beispiel für STAGs erinnert (Abbildung 11). Auch hier wird von der Grammatik lediglich $a^n b^n c^n d^n e^n f^n g^n h^n$ erkannt, was keine normale TAG bewerkstelligen kann.

6 Vergleich mit dem Shake-and-Bake-Übersetzungsverfahren

Vom Prinzip her sehr ähnlich zu dem Übersetzungsverfahren mit STAGs ist das Shake-and-Bake-Verfahren, das von J. L. Beaven und P. Whitelock entwickelt wurde und in [Bea92] und [Whi92] ausführlich beschrieben wird. Im Gegensatz zu dem STAG-basierten Ansatz, werden hier keine geparsten Syntax-Bäume als Grundlage genommen, sondern die syntaktischen Beziehungen zwischen den Wörten. Syntaktische Beziehungen meint hier z.B. bei Verben, welches Subjekt, welche Objekte und was für Präpostionen es hat, bei Adverben ist angegeben zu welchem Verb es gehört usw. (siehe auch Abbildung 23 und 24).

Es gibt nun Übersetzungsregeln, die für ein oder mehrere Wörter gelten und die syntaktischen Beziehungen zwischen den einzelnen Wörtern ändern

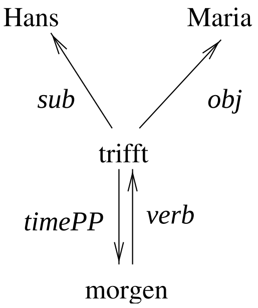

Abbildung 23: Syntaktische Beziehungen zwischen den Wörtern bei dem Satz *Hans trifft Maria morgen*

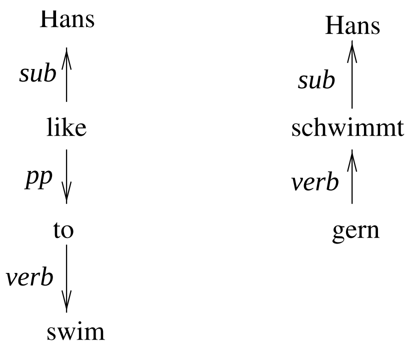

Abbildung 24: Syntaktische Beziehungen zwischen den Wörtern bei den Sätzen *Hans schwimmt gern* und *Hans like to swim*

können. Diese Regeln übernehmen somit die Funktion der Links in STAGs. Abbildung 25 zeigt eine solche Regel für das Wortpaar *gerne* $\leftrightarrow$ *like to*, wobei Abbildung 24 als Grundlage genommen wird. Die dicken Punkte in den Abbildungen repräsentieren Wörter, für die die syntaktischen Beziehungen gelten. Die gestrichelten Linien entsprechen den Regeln, die die Änderung der syntaktischen Beziehungen bewirken.

Der Vorteil von Shake-and-Bake gegenüber STAGs ist, daß die Übersetzungsregeln relativ unabhängig von den eingesetzten Parsegrammatiken sind. Damit können zum einen unabhängig von einander entwickelte Grammatiken (die auch von unterschiedlichem Typ sein können) zum Parsen des Ausgangs- und Zielsatzes verwendet werden. Anderseits sind Änderungen an den Grammatiken möglich, ohne daß die Übersetzungsregeln davon betroffen wären.

Schwierig beim Shake-and-Bake Verfahren ist dagegen die Generierung des Zielsatzes, da keine Baumstruktur mehr vorhanden ist, die einfach durchlaufen werden könnte. Die Autoren schlagen deshalb einen Generierungsalgo-

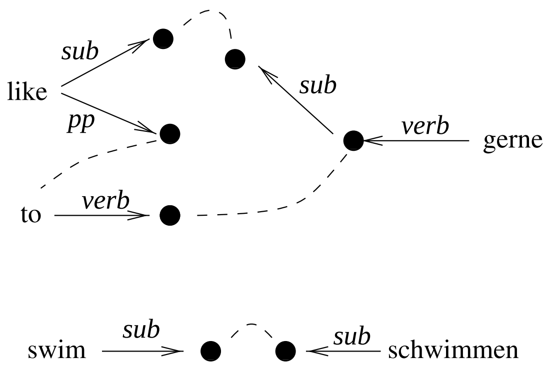

Abbildung 25: Übersetzungsregel für das Wortpaar *gerne* und *like to*

rithmus vor, der im wesentlichen ein modifizierter Shift-Reduce-Parser ist. (Dieser gab dem Verfahren auch seinen Namen.) Dabei wird in trial-and-error-Manier versucht, Wörter bzw. bereits kombinierte Wörter mit Hilfe der Grammatik für die Zielsprache miteinander zu kombinieren. Wenn alle Wörter miteinander kombiniert werden konnten, kann der Satz ausgegeben werden.

7 Fazit

Meines Erachtens ist die Übersetzung mit STAGs, ein Ansatz, den es weiter zu verfolgen lohnt. Der Aufwand für die Implementierung ist recht gering, Idiome können relativ problemlos verarbeitet werden und dadurch, daß direkt die synaktischen Strukturen verwendet werden, ist das Verfahren auch sehr flexibel und schnell an neue Bedürfnisse anpassbar. Außerdem können unter Verwendung von Entwicklungsumgebungen (auch Frameworks genannt) die Baumpaare für „normale“ Fälle leicht automatisch erstellt werden, aber für Sonderfälle können die Baumpaare zur Not auch von Hand erstellt werden.

Solche Frameworks zur automatischen Generierung von Baumpaaren sind allerdings auch bitter nötig (vorallem für die unzähligen Idiome, die in den Sprachen vorkommen), da die Menge der zu erstellenden Bäume einfach zu groß ist. Es müssen ja alle syntaktische Variationen, wie z.B. Aussagesatz, Fragesatz usw. mit den Bäumen abgedeckt werden. Die Erstellung der Links ist zudem sehr fehleranfällig und sollte nur in Ausnahmefällen von Hand erledigt werden.

Außerdem bedarf es noch einer Kompontente, die von den verschiedenen Übersetzungsmöglichkeiten eines Satzes, die die STAG-Komponente liefert, einen auswählt. Dazu ist eine Art semantischer und pragmatischer Analyse des Textes notwendig. Eine im Vergleich zu der hier angesprochen Aufgabe des Transfers ungleich schwierigere Aufgabe.

Literatur

[ASJ90] Anne Abeillé, Yves Schabes, and Aravind K. Joshi. Using lexicalized tags for machine translation. In *Proceedings of the International Conference on Computational Linguistics (COLING '90)*, Helsinki, Finland, 1990.

[Bea92] J.L. Beaven. Shake-and-bake translation. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING'92)*, pages 603–609, 1992.

[HP96a] Karin Harbusch and Peter Poller. Non–isomorphic synchronous tags. Technical report, DFKI GmbH, 1996.

[HP96b] Karin Harbusch and Peter Poller. Structural translation with synchronous tree adjoining grammars in verbmobil. Technical Report 184, DFKI GmbH, 1996.

[Pri97] Gilles Prigent. Synchronous tree adjoining grammars. In `http://www.cis.upenn.edu/~xtag/stag/`, 1997.

[Shi94] Stuart Shieber. Restricting the weak generative capacity of synchronous tree–adjoing grammars. *Computational Intelligence*, 10(4):371–385, November 1994.

[SS90] Yves Schabes and Stuart Shieber. Synchronous tree adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*, Helsinki, 1990.

[Whi92] Peter Whitelock. Shake-and-bake translation. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING'92)*, pages 784–790, 1992.