BMBF

Verbmobil
Verbundvorhaben

Inkrementelle Erstellung von Worthypothesengraphen

Thilo Pfau
Günther Ruske

Technische Universität München

Vm **Report 107**
März 1996

März 1996

Thilo Pfau
Günther Ruske

Forschungsgruppe Sprachverarbeitung
Lehrstuhl für Mensch-Maschine-Kommunikation
Technische Universität München
Arcisstr.21
80290 München

Tel.: (089) 2105 - 8563
e-mail: ruske@e-technik.tu-muenchen.de

Gehört zum Antragsabschnitt: TP3 Spracherkennung und Sprecheradaption

Das diesem Bericht zugrundeliegende Forschungsvorhaben wurde mit Mitteln des Bundesministeriums für Forschung und Technologie unter dem Förderkennzeichen 01 IV 102 C/6 gefördert. Die Verantwortung für den Inhalt dieser Arbeit liegt bei den Autoren.

Inhaltsverzeichnis

1 Einführung

Im Rahmen des Verbmobil-Projekts wird vom Modul „Spracherkenner“ erwartet, daß eine Liste von möglichen erkannten Wörtern ausgegeben wird. Diese Liste wird als Worthypothesengraph abgelegt und enthält alternative Erkennungsergebnisse für Wörter mit alternativen Segmentierungen. Im Gegensatz zur akustischen 1-best Satzerkennung bietet sich in diesem Fall für die nachfolgende linguistische Verarbeitung die Möglichkeit, unter Einbeziehung höherer Wissensquellen (Syntax, Semantik, Dialog) die Satzerkennung entscheidend zu verbessern.

Die höheren Stufen verwerten im allgemeinen mehr Information, als der beste - nur aufgrund der akustischen Ähnlichkeit - erkannte Satz verkörpert. Vielmehr benötigen die nachfolgenden Stufen eine Repräsentation eines Teils des Suchraumes. Der Suchprozeß trifft also noch keine Entscheidung über die gesprochene Wortfolge, sondern bietet den nachfolgenden Stufen mehrere Alternativen an. Verschiedene auf Basis der bisherigen Wissensquellen sinnvoll erscheinende Satzhypothesen werden dabei als Eingangsdaten für die weiteren Stufen benötigt. Die N besten zur Merkmalsvektorfolge passenden Sätze - in einer anhand der Wissensquellen bewerteten Abfolge (Ranking) - werden den nachfolgenden Stufen angeboten.

Prinzipiell ist es einfach, die N besten Sätze zu bestimmen. Am Ende des Suchvorgangs werden anstelle eines Backtrackings der am besten endenden Hypothese die N besten endenden Hypothesen zurückverfolgt und in der Reihenfolge ihrer Scores ausgegeben. Dies ist in der Praxis aber nicht zufriedenstellend. Viele Hypothesen unterscheiden sich nur in einer verschiedenen Segmentierung derselben Wortfolge, d.h. die Wörter beginnen und enden zu verschiedenen Zeitpunkten. Eine auf diese Art erstellte Liste von Sätzen enthält damit sehr viel redundante Information. Außerdem kann diese Wortliste nicht inkrementell erstellt werden. Die nachfolgenden Stufen können deshalb ebenfalls nicht inkrementell arbeiten.

Ideal wäre die Einbindung der Information der höheren Stufen direkt in die integrierte, inkrementell arbeitende Suche, so daß alle zur Verfügung stehenden Wissensquellen gleichzeitig ausgewertet werden können. Für diesen Ansatz des Problems gibt es aber zur Zeit noch keine befriedigenden Lösungsansätze.

Als suboptimale Lösung des Problems bietet sich die inkrementelle Erstellung der bewerteten Satzlisten während des Suchvorgangs an, die im folgenden ausführlich vorgestellt wird. Die „höheren“ Stufen können dann ebenfalls inkrementell arbeiten. Weiterhin erweist sich die Darstellung der Listen in Form von Worthypothesengraphen als sinnvoll.

Bei Worthypothesengraphen, auch „Wortlattices“ genannt, handelt es sich um Wortnetze. In diesen werden Wörter, die in verschiedenen Sätzen an ähnlichen Positionen vorkommen, nicht mehrfach geführt. Abbildung 1 zeigt ein Beispiel eines Worthypothesengraphen, der z.B. als sinnvollen Teilsatz die Wortfolge „ich wollte Sie fragen“ enthält.

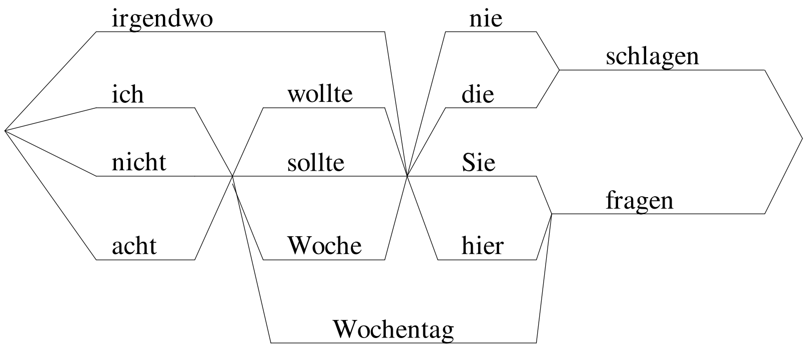


Abbildung 1: Beispiel für einen Worthypothesengraphen

Grundsätzlich können Worthypothesen aber auch ohne höhere Verarbeitungsstufen sinnvoll eingesetzt werden. Sie enthalten mehr Informationen über den Suchvorgang als der am besten erkannte Satz. Die „Breite“ der Lattice ist abhängig von der Sicherheit der Entscheidung während des Suchvorgangs. Laufen viele Pfade parallel zueinander, d.h. gibt es viele Alternativen zu einem erkannten Wort, so konnte keine sichere Entscheidung über das gesprochene Wort getroffen werden. Sind dagegen nur wenige parallele Pfade in der Lattice enthalten, kann die getroffene Entscheidung als wesentlich sicherer angesehen werden.

In diesem Zusammenhang ist auch die Verzweigungsdichte des Graphen wichtig. d.h. in wie viele Pfade sich die Lattice nach einem Knoten verzweigt. Diese ist ebenfalls ein Indikator für die Sicherheit der getroffenen Entscheidung. Zudem ist die Anzahl der abgehenden und eintreffenden Kanten pro Knoten entscheidend für die Anzahl der verschiedenen Sätze, die auf Basis eines Worthypothesengraphen gebildet werden können. Aus derselben Anzahl von Wörtern können je nach Verzweigungsdichte der enthaltenen Knoten unterschiedlich viele verschiedene Sätze gebildet werden.

Für eine sinnvolle Auswertung durch die nachfolgenden Stufen muß der Graph bestimmten Anforderungen genügen.

Da als gültige erkannte Sätze nur Wortfolgen angesehen werden, entlang derer man vom Anfangsknoten des Graphen bis an dessen Endknoten gelangen kann, muß der Graph lückenlos sein. Das bedeutet, daß er keine Hypothesen enthalten darf, die nicht bis zum gemeinsamen Endknoten verfolgt werden können. Der Graph darf also keine „toten" oder „offenen" Enden enthalten. Ebenso müssen alle Hypothesen vom Satzanfangsknoten aus erreichbar sein. Hypothesen, die erst später beginnen, also „offene" Anfänge darstellen, sind nicht akzeptabel.

Äußerst kritisch für die nachfolgenden Stufen ist das Vorhandensein von Zyklen im Graphen, da die „höheren" Stufen alle aus dem Graphen zu bildenden Sätze auswerten. Sind Zyklen in einem Graphen enthalten, können aber unendlich viele Sätze aus einem Worthypothesengraphen gebildet werden. Die Entstehung von Zyklen in einem Worthypothesengraphen muß deshalb unter allen Umständen vermieden werden.

1.1 Bewertung von Worthypothesengraphen

Soll aus einem Worthypothesengraphen ohne die nachfolgenden „höheren" Stufen ein Ergebnis gewonnen werden, so sind andere Beurteilungskriterien nötig als bei der Bewertung des besten Satzes.

Bei der Ausgabe des besten Satzes wird ein DP-Matching von vorgegebenem korrektem und erkanntem Satz durchgeführt, um die Worterkennungsrate zu bestimmen. Liegt das Ergebnis dagegen in Form einer Lattice vor, ist ein anderes Vorgehen notwendig. Zur Bestimmung der Worterkennungsrate ist nun ein DP-Matching für jeden möglichen Weg durch den Graphen durchzuführen. Die beste dabei erreichte Erkennungsrate wird als Ergebnis angesehen.

Nun ist es aber leicht einzusehen, daß je größer die Lattice wird (je größer N gewählt wird), es umso wahrscheinlicher ist, daß der korrekte Satz in der Lattice enthalten ist, die Qualität der Erkennung aber deshalb nicht unbedingt besser sein muß. Der korrekte Satz kann z.B. in einer umfangreichen Lattice enthalten sein, oder er kann in einer kleinen Lattice enthalten sein. Beide Alternativen liefern eine Worterkennungsrate von hundert Prozent und werden gleich bewertet, wenn nur die Worterkennungsrate betrachtet wird. Die Worterkennungsrate reicht also als Kriterium für die Erkennung nicht mehr aus, da es wenig sinnvoll erscheint, eine riesige Lattice zu erzeugen, die dann mit großer Sicherheit den korrekten Satz enthält. Mit steigender Latticegröße sinkt gleichzeitig die Wahrscheinlichkeit, daß die linguistische Verarbeitung einen „auf dem Grund der Lattice" liegenden Satz noch als bestes Endergebnis findet, da viele andere Sätze gültig und wahrscheinlicher sind.

Bei Verwendung der Worthypothesengraphen ohne „höhere“ Verarbeitungsstufen ist das Ziel , eine möglichst kleine Lattice zu erzeugen, die eine hohe Worterkennungsrate liefert. Es fehlt noch ein Kriterium, um die Größe einer Lattice zu beurteilen. Hierfür bietet sich die Worthypothesendichte (WHD) an. Diese ist definiert als der Quotient aus Anzahl der Wörter in der Lattice und Anzahl der Wörter im korrekten Satz.

$$\mathrm{WHD} = \frac{\mathrm{Anzahl\_W\ddot{o}rter\_in\_Lattice}}{\mathrm{Anzahl\_W\ddot{o}rter\_in\_korrektem\_Satz}}$$

Die Worthypothesendichte ist aber nur eingeschränkt als Beurteilungskriterium geeignet, da nur die Anzahl der Worthypothesen eingeht, nicht aber die Verzweigungsdichte des Graphen. Ein geeignetes Maß sollte sowohl die Anzahl der enthaltenen Worthypothesen wie auch die Verzweigungsdichte berücksichtigen.

Als zusätzliches Maß zur Berücksichtigung der Verzweigungsdichte des Graphen wird die Perplexität (PP) verwendet. Diese ist definiert als der Quotient aus der Anzahl der Kanten und der Anzahl der Knoten im Graphen.

$$\mathrm{PP} = \frac{\mathrm{Anzahl\_der\_Kanten}}{\mathrm{Anzahl\_der\_Knoten}}$$

Bei Einsatz der Worthypothesengraphen in einem System mit linguistischen Verarbeitungsstufen werden je nach verwendeten „höheren“ Stufen an die erzeugten Graphen unterschiedliche Anforderungen gestellt.

Trifft die eingesetzte „höhere“ Stufe eine harte Entscheidung über die untersuchte Satzhypothese (wahr oder falsch), ist ein breiter Hypothesengraph durchaus sinnvoll. Hypothesen, die aufgrund der bisherigen Wissensquellen als sehr gut angesehen wurden, können sich z.B. als grammatikalisch falsch erweisen. Andere, bisher als schlecht bewertete Hypothesen dagegen können grammatikalisch richtig sein und so als beste Hypothese aus der „höheren“ Stufe hervorgehen. Eine „höhere“ Stufe, die harte Entscheidungen trifft kann das Ranking der Hypothesen nachhaltig beeinflussen.

Ist die nachfolgende Stufe eher als weich anzusehen, d.h. verändert sie die bisher verteilten Scores nur in geringem Maße, so ist die Erstellung vieler Hypothesen nicht sinnvoll. Die Bewertung durch die „höheren“ Stufen verändert die Reihenfolge der Sätze in diesem Fall nur unwesentlich. Akustisch schlecht bewertete Satzhypothesen haben wenig Chancen, durch die nachfolgende Bewertung in der Hierarchie weit aufzusteigen.

2 Erzeugung von Worthypothesengraphen

2.1 Ziele

Beim Einsatz des Spracherkennungssystems im Online-Betrieb werden „höhere“ Verarbeitungsstufen verwendet, die das akustische Ranking nur wenig beeinflussen können. Der Online-Betrieb macht es unmöglich, eine harte grammatikalische Entscheidung (grammatikalisch korrekt oder nicht korrekt) zu treffen, da im Online-Betrieb häufig spontansprachliche Äußerungen vorkommen, die im Sinne einer strengen Grammatik als falsch angesehen werden müssen, aber in der Umgangssprache als richtig akzeptiert werden.

Das Ziel ist es deshalb, auf Basis der bisherigen Wissensquellen einen Worthypothesengraphen zu erzeugen, der eine möglichst kleine Worthypothesendichte aufweist und gleichzeitig eine hohe Erkennungsrate. Aufgrund des sehr großen Suchraumes enthalten aber die von der Suche erzeugten Satzhypothesen sehr viel redundante Information. Eine sinnvolle Reduktion der Hypothesenanzahl ist daher notwendig.

Natürlich kann die Hypothesenanzahl durch die Parameter des Histogramm-Prunings reguliert werden. Eine Veränderung der Pruningparameter hat aber nicht nur einen Einfluß auf die von der Suche erzeugten Worthypothesen. Der gesamte Suchraum wird durch diese Parameter nachhaltig beeinflußt. Eine Veränderung der Pruningparameter ist auch allein schon deshalb nicht sinnvoll, da der mit Hilfe des Backtrackings ermittelte beste Satz durch diese Parameter stark verändert werden kann.

Ein weiterer, zusätzlicher Pruningmechanismus auf Wortebene scheint deshalb sinnvoll. Dieser soll die Anzahl der Worthypothesen, die von der Suche an die Schnittstelle zum Worthypothesengraphen übermittlelt werden reduzieren, ohne dabei den Suchraum zu beeinflussen. Es werden also nicht alle von der Suche erzeugten Worthypothesen an das Modul zur Latticeerzeugung übermittelt.

Ein grundsätzliches Ziel besteht außerdem darin, die während der Suche entlang der Pfade aufgebauten Wort-Bigramm-Bindungen bei der Lattice-Erzeugung aufrecht zu erhalten. Der Worthypothesengraph soll eine möglichst genaue Repräsentation des Suchraumes darstellen. Dabei ist die Beibehaltung der Wort-Vorwortfolgen der verfolgten Pfade von großer Wichtigkeit.

2.2 Nichtinkrementelle Erstellung von Worthypothesengraphen

Der Aufbau der Lattice erfolgt hier anhand des Backtrackingarrays, das während der Suche aufgebaut wurde. Dieses enthält in der bisher realisierten Version des Erkenners nur Informationen über den besten Satz. Für jedes Wort wird nur der beste Vorgänger gespeichert, d.h. anhand der besten im letzten Zeitschritt der Suche endenden Hypothese kann - beginnend beim letzten Wort des Satzes - der beste erkannte Satz Wort für Wort zurückverfolgt werden.

Für die Erzeugung eines Worthypothesengraphen anhand des Backtrackingarrays müßte dieses Array pro Worthypothese mehrere mögliche Vorgänger enthalten. Für diese verschiedenen Vorgänger müßten ebenfalls die Scores enthalten sein. Die Scores sind dabei die vom Satzanfang bis ans Ende des betreffenden Wortes aufakkumulierten Scores. Hierbei gehen die Emissionen der Zustände, die Interzustands-Scores, die Inter-HMM-Scores, sowie die Wortübergangsstrafen ein. Problematisch bei der Erstellung der Worthypothesengraphen auf diese Art ist aber grundsätzlich, daß kein inkrementelles Vorgehen möglich ist. Das Backtrackingarray liegt erst nach dem letzten Suchschritt komplett vor.

Allerdings bietet ein derart erstellter Worthypothesengraph auch Vorteile. Die aus dem Backtrackingarray bestimmbaren Sätze werden aufgrund einer globalen Maximumentscheidung am Ende der Suche ausgewählt. Es können grundsätzlich die N-besten während der Suche erstellten Hypothesen ermittelt werden. Das Backtrackingarray repräsentiert exakt einen Teil des Suchraumes. Da die Erstellung der Worthypothesengraphen sich aber ebenfalls in das Konzept des inkrementellen Vorgehens einfügen soll, scheidet diese, mit vergleichbar geringem Aufwand zu realisierende Alternative aus.

2.3 Inkrementelle Erstellung von Worthypothesengraphen

Die Worthypothesengraphen können auch anhand von lokalen Entscheidungen während des Suchvorgangs erstellt werden, was aber zwangsläufig zu einem suboptimalen Verfahren führt.

In vielen Suchschritten erfolgen Wortübergänge in den aktuell verfolgten Suchpfaden. Für alle Pfade, die solche Wortübergänge im aktuellen Schritt enthalten, können die endenden Worthypothesen mit ihren jeweiligen Vorgängerwörtern zum Aufbau eines Worthypothesengraphen verwendet werden.

Das Grundproblem hierbei ist, daß die Optimalität des Vorgehens verloren geht, da jetzt Entscheidungen über die Aufnahme von Hypothesen in den Graphen aufgrund von lokal erreichten Scores getroffen werden müssen. Es ist also nicht gewährleistet, daß die N besten Sätze in der Lattice enthalten sind, da lokal mit guten Scores bewertete Hypothesen gegen Ende schlechte Scores erhalten können und somit nicht mehr zu den N besten Hypothesen am Ende gehören. Im Extremfall können lokal mit guten Scores versehene Hypothesen durch das Histogramm-Pruning enden, bevor das Satzende erreicht ist. Dadurch entstehen sogenannte „tote“ oder „offene“ Enden im Graphen. Diese stellen keine sinnvoll auszuwertende Information dar, da nur Wortfolgen, die vom Satzanfang bis zum Satzende durchgehend verfolgt werden können, einer späteren Bewertung unterzogen werden. Grundsätzlich stellt sich das Problem als eine Reduktion eines Baumes, auf einen geschlossenen Graphen dar.

In Abbildung 2 ist die Reduktion des Suchraums auf einen geschlossenen Graphen skizziert. Der Suchraum wurde hier vereinfacht ohne Rekombinationen und auf Basis von ganzen Wörtern dargestellt. In der Realität erfolgt schon während der Suche eine Rekombination und der Suchraum basiert auf Phonemen.

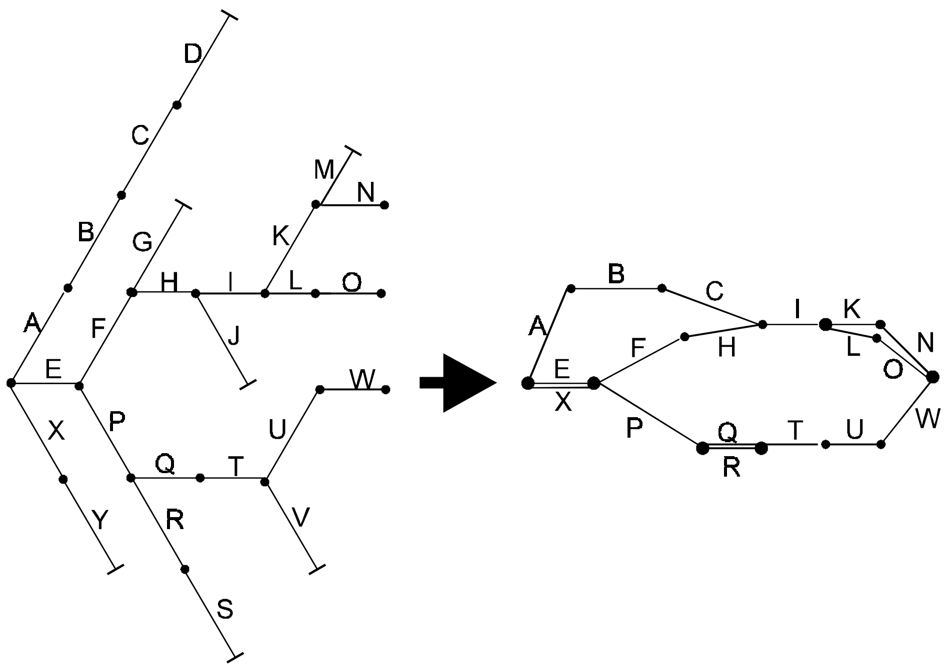


Abbildung 2: Suchbaum ⇒ geschlossener Graph

Die Wörter D, G, J, M, S, V und Y stellen hier „tote“ Enden dar, die durch das Histogramm-Pruning entstanden sind. Die Reduktion des Suchraumes auf den

geschlossenen Graphen erfolgte unter Zuhilfenahme der Methoden, die im Worthypothesengraphen-Modul implementiert sind und nachfolgend erläutert werden.
Der verfolgte Lösungsansatz bei der inkrementellen Erstellung der Worthypothesengraphen ist dem Ansatz des Phonem-Lookahead ähnlich. Entscheidungen über die Aufnahme einer Hypothese in die Lattice werden nicht aufgrund der vom aktuellen Suchschritt emittierten Wortfolgehypothesen (Wort und zugehöriges Vorwort) getroffen, sondern es erfolgt eine Pufferung dieser Hypothesen, und die Entscheidung wird für Hypothesen früherer Suchschritte getroffen, für die schon eine gewisse „Zukunft“ im Puffer enthalten ist. Für die Entscheidungsfindung kommt folgende Strategie zum Einsatz. Die Pufferung der Hypothesen erfolgt in einem Ringpuffer, der mindestens Teilhypothesen der Länge K+1 enthält. Teilhypothesen der Länge K+1 sind Wortfolgen der Länge K+1, also ein Wort und K weitere Folgewörter.

Bei der Aufnahme in den Puffer kommt zunächst eine zusätzliche Pruningschwelle auf Wortebene zum Einsatz. Alle Scores derjenigen Pfade, die im aktuellen Suchschritt einen Wortübergang enthalten, werden verglichen. Die Wortfolgehypothesen (Wort und Vorgänger) dieser Pfade, die eine im Vergleich zum besten dieser Scores festgelegte Schwelle überschreiten, können prinzipiell in den Puffer aufgenommen werden.

An dieser Stelle kommt ein weiterer Mechanismus zur Reduktion der im Suchraum enthaltenen Redundanz zum Einsatz. Viele Wortfolgehypothesen unterscheiden sich nur durch unterschiedliche Segmentierungen derselben Wort-Vorwort-Folge, d.h. die Anfänge und Enden der Hypothesen fallen auf unterschiedliche Anfangs- und Endzeitpunkte. Liegen die Anfangs- und Endzeitpunkte einer in den Puffer aufzunehmenden Wortfolgehypothese in einem vordefinierten Bereich um die Anfangs- und Endzeitpunkte einer schon im Puffer befindlichen identischen Wortfolgehypothese (gleiches Wort-Vorwort-Paar), so wird nur diejenige Hypothese in den Puffer aufgenommen, die den besseren Score aufweist. Ist die neue Hypothese die „bessere“, so wird die alte Hypothese durch diese ersetzt. Anderenfalls wird die neue Hypothese nicht in den Puffer aufgenommen.

Bei der Aufnahme in den Puffer werden Knotennummern vergeben. Dafür wird für neu aufzunehmende Hypothesen im Puffer nach passenden Vorgängern gesucht. Der Wortanfang des aufzunehmenden Wortes bekommt dieselbe Knotennummer wie das Wortende desjenigen Wortes, an das der Wortanfang angehängt wird. Für die Wortenden der aufzunehmenden Wörter werden vorläufige Knotennummern vergeben. Da Wörter, die später enden auch später in den Puffer aufgenommen werden, erhalten diese auch höhere

Endknotennummern. Die Knotennummern repräsentieren also die Chronologie der Aufnahme in den Puffer.

Die Aufnahme in den Puffer gestaltet sich aufwendig, da mehrere Fälle zu unterscheiden sind:

- Für ein aufzunehmendes Wort können mehrere passende Vorgängerwörter gefunden werden. Passende Vorgänger sind dabei Wörter, die die Wort-Bigramm-Bedingung der aufzunehmenden Hypothese erfüllen und in einem definierten Bereich (Frameradius R) um den Anfangszeitpunkt der Hypothese enden. Werden mehrere passende Vorgänger gefunden, so werden die Enden dieser Vorgängerwörter auf einen gemeinsamen Knoten gelegt. Verwendet wird dabei, sofern keiner der Vorgänger schon Nachfolger im Puffer besitzt, der Endknoten desjenigen Vorgängers, der Teil des Pfades mit dem besten Score ist. Dieser Score wird zur weiteren Berechnung des Pfadscores verwendet (siehe Abbildung 3: Fall 1, Einhängen von Wort B(A) (Wort B mit Vorgänger A) in den Puffer).
- Besitzt einer der passenden Vorgänger aber schon Nachfolger im Puffer, werden alle Enden der Vorgänger auf den Endknoten des Vorgängers gelegt, der schon Nachfolger besitzt. Die neue Hypothese wird an diesen Knoten angehängt (siehe Abbildung 3: Fall 2) und der Pfadscore anhand des Scores dieses Vorgängers berechnet.
- Es kann aber auch vorkommen, daß für eine aufzunehmende Hypothese keine Vorgänger gefunden werden. Dies hat seine Ursache in der zusätzlichen Pruningschwelle auf Wortebene. Durch diese Pruningschwelle stimmen Suchraum und Pufferinhalt in dem im Puffer enthaltenen Bereich nicht exakt überein. Dies hat folgenden Grund. Ein Wort überschreitet die zusätzliche Pruningschwelle nicht und wird deshalb nicht in den Puffer aufgenommen. Im Suchraum wird der entsprechende Pfad aber weitergeführt, da die Histogramm-Pruningschwelle von dieser Hypothese in den folgenden Zeitschritten überschritten wird. Überschreitet nun aber ein Nachfolger dieses Wortes, das nicht in den Puffer aufgenommenen wurde, die zusätzliche Pruningschwelle und erfüllt auch alle anderen Bedingungen zur Aufnahme in den Puffer, so ist kein passender Vorgänger im Puffer enthalten. Um die Hypothese, die ja im Suchraum bis hierher „überlebt" hat, aber nicht ganz verwerfen zu müssen, wird nun die betreffende Hypothese unter Verletzung der Wort-Vorwort-Bedingung an ein anderes Vorgängerwort angehängt (siehe Abbildung 3: Fall 3). Dabei wird als Vorgängerwort das innerhalb des Frameradius endende Wort mit dem besten Score gewählt.

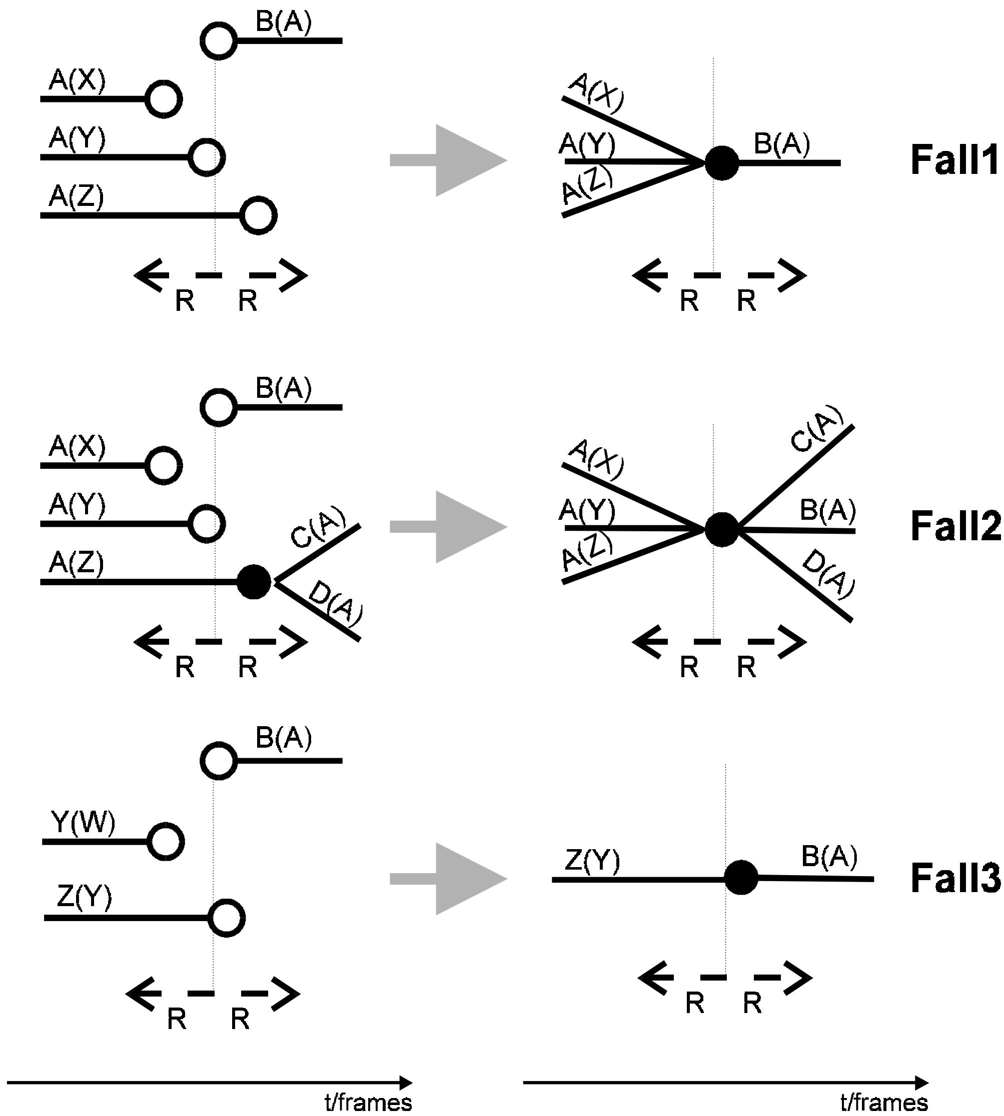


Abbildung 3: Aufnahme einer Hypothese in den Puffer

Bei der Aufnahme der Hypothesen in den Puffer wird auf diese Art neben einer Reduktion der Redundanz gleichzeitig eine Rekombination von Wortenden durchgeführt und damit eine Bedingung zur Erstellung eines geschlossenen Graphen erfüllt.

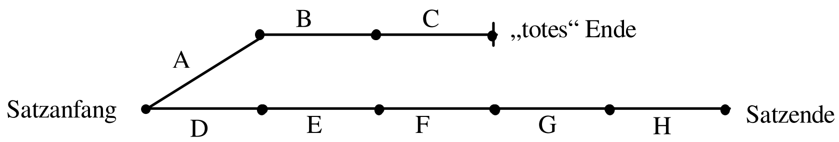
Nachdem nun der Vorgang der Aufnahme der Hypothesen in den Puffer beschrieben worden ist, wird jetzt näher auf die Aufnahme der Hypothesen in den Worthypothesengraphen eingegangen. Der Puffer sollte sinnvollerweise Wortfolgehypothesen der Länge K+1 enthalten. Ist der Puffer voll, müssen Hypothesen aus dem Puffer entfernt werden, um neu eintreffende Hypothesen aufnehmen zu können. Da es sich bei dem Puffer um einen FIFO handelt, werden die am längsten im Puffer befindlichen Hypothesen aus diesem entfernt. Dabei wird nun die Entscheidung getroffen, ob die Hypothese in den Graphen aufgenommen werden soll oder nicht.

Folgende Strategie kommt zum Einsatz:

- Hat das aus dem Puffer zu entfernende Wort eine Nachfolgerkette der Länge K aufzuweisen, wird es in die Lattice aufgenommen, da es sehr sicher erscheint, daß diese Hypothese bis zum Satzende weitergeführt wird.
- Hat ein Wort dagegen gar keine Nachfolger oder eine Nachfolgerkette deren Länge kleiner ist als K-1, so wird das Wort verworfen und nicht in den Graphen aufgenommen, da es wenig wahrscheinlich erscheint, daß diese Hypothese noch weiter in die Zukunft reicht.
- Eine Besonderheit stellen Wörter mit Nachfolgerketten der Länge K-1 dar. Bei richtig gewählter Puffergröße ist es als sicher anzusehen, daß alle auftretenden Nachfolgerketten bis mindestens zur Länge K zum Zeitpunkt der Entscheidung im Puffer enthalten sind. Angenommen es existiere eine Wortfolgekette der Länge K+1, also ein Wort und eine Nachfolgerkette der Länge K. Diese stelle ein „totes“ Ende im Suchraum dar, da keine weiteren Nachfolger das Pruning überstehen, das Satzende aber noch nicht erreicht ist. Soll nun das erste Wort dieser Kette aus dem Puffer entfernt werden, wird es in die Lattice aufgenommen, da es eine Nachfolgerkette der Länge K enthält. Soll daraufhin noch das darauf folgende Wort aus dem Puffer entnommen werden, so stellt man fest, daß es nur eine Nachfolgerkette der Länge K-1 aufweist. Eine Nichtaufnahme in die Lattice würde aber dazu führen, daß das erste, bereits ausgegebene Wort als totes Ende in der Lattice stehenbleibt. Dies muß vermieden werden. Deshalb werden Wörter, die Nachfolgerketten der Länge K-1 aufweisen, präventiv an andere Nachfolger angehängt, die Nachfolgerketten der Länge K aufweisen. Dabei wird die Wort-Bigramm-Bedingung verletzt, aber das Entstehen von „toten“ Enden in der Lattice vermieden.

Dieser Sachverhalt wird anhand von Abbildung 4 klarer. Das angeführte Beispiel verwendet den beschriebenen Mechanismus mit Nachfolgerketten der Länge K=2. Diese Länge wurde im Hinblick auf das Bigramm-Sprachmodell als sinnvoll angesehen und im Erkennungssystem eingesetzt. Die Wortkette bestehend aus den symbolischen Wörtern A, B und C bildet ein „totes“ Ende im Puffer. Wort A wird in die Lattice aufgenommen, da es eine Nachfolgerkette der Länge 2 (entspricht K) aufweist. Wort B hat nur eine Nachfolgerkette der Länge 1 (entspricht K-1) und wird deshalb an den Knoten zwischen den Wörtern E und F angehängt. Wort C wird nicht in die Lattice aufgenommen, da es weniger als K-1 Nachfolger hat.

Pufferinhalt:



resultierender Graph:

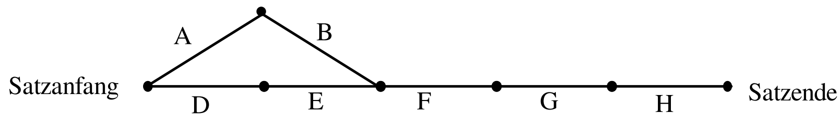


Abbildung 4: Präventives Einhängen (K=2)

Durch Rekombination beim Latticeaufbau und präventives Einhängen können vorher entfernte doppelte Kanten erneut entstehen. Um diese zu eliminieren, wird ein kleiner Ausgabepuffer eingeführt, der bei identischen Kanten (gleiches Wort zwischen gleichen Knoten) diejenige Kante mit dem besseren Score auswählt und die andere verwirft.

Am Ende des Satzes wird ein absoluter Endknoten eingeführt. Dieser wird auf den Endframe des letzten eingegangenen Wortes gelegt. Die restlichen im Puffer befindlichen Einträge werden wie oben behandelt und ausgegeben.

3 Darstellung des Worthypothesengraphen

Die Auswertung der Worthypothesengraphen zur Bestimmung von Worterkennungsrate und Worthypothesendichte erfolgt mit den Programmen „vmeval“ und „vmevadoc“. Diese stellen strenge Anforderungen an die Darstellung der Worthypothesengraphen. (siehe [1]).

Der Worthypothesengraph muß zusammenhängend sein, d.h. jede Hypothese muß entlang eines Pfades vom Anfangsknoten aus erreichbar sein. Es darf also keine „offenen Anfänge“ in der Lattice geben. Der Graph wird kantenweise geschrieben, wobei für jede Kante eine neue Zeile in folgendem ASCII-Format eingetragen und durch das <newline>-Symbol abgeschlossen wird. Eine Kante entspricht einer Worthypothese:

Kante := <A E wort score ta te [infostring]>

mit:

A:	logischer Anfangsknoten
E:	logischer Endknoten
wort:	Hypothese
score:	Bewertung der Hypothese
ta:	Anfangsframe
te:	Endframe
infostring:	zusätzliche Informationen

Alle Einträge müssen nach steigenden Anfangsknotennummern geordnet sein. Bei gleichen Anfangsknoten wird nach steigenden Endknotennummern sortiert. Es dürfen keine Knotennummern übersprungen werden. Der Graph beginnt mit der Anfangsknotennummer 1. Die Knotennumern („A" und „E") müssen eine zeitliche Zuordnung ermöglichen, d.h. kleinere Knotennumern müssen sich auf zeitlich frühere Ereignisse beziehen. Jeder Eintrag enthält zwingend die Einträge A, E, wort, score, ta und te. Die Angabe eines infostrings ist optional. Begrenzer in einer Zeile zwischen den einzelnen Einträgen sind Leerzeichen oder Tabulatorzeichen. Beginn und Ende des Graphen werden durch die Schlüsselwörter „BEGIN_LATTICE" und „END_LATTICE" gekennzeichnet, die jeweils in einer eigenen Zeile stehen müssen. Vor der Kennung „BEGIN_LATTICE" wird eine Kommentarzeile mit der Kennung der Sprachdatendatei des Turns eingefügt (Kommentarzeilen werden durch % eingeleitet), z.B. %TURN: /opt/vml-cd1/g071a/g071a001.a16

In der folgenden Abbildung ist ein Ausschnitt aus einem Worthypothesengraphen dargestellt. Dieser wurde auf Basis der Sprachdatei „g071a001.a16" erstellt. Der korrekte Satz lautet: „ja also für den eintägigen wenn wir den als erstes erledigen wollten quasi wäre mir ganz recht Montag der achte November."

```
%TURN: /opt/vml-cd1/g071a/g071a001.a16
BEGIN_LATTICE
1   2   ja          3068.611  0    34
2   3   auch        2982.239  34   65
3   4   #PAUSE#     1959.368  65   83
3   5   Hause       3904.182  65   107
4   5   Hause       2712.975  83   107
5   6   fr"uh       4971.819  107  149
6   7   den         2144.383  149  168
7   8   #NIB#       1294.641  168  184
7   8   #PAUSE#     1207.847  168  184
7   8   ja          1381.884  168  184
8   9   eint"agigen 8056.564  184  266
8   10  bei         7678.305  184  264
8   10  beim        7760.885  184  264
8   10  dein        7758.903  184  264
8   10  deine       7769.287  184  264
8   10  ein         8815.395  184  264
```

...

31	34	dachte	5046.016	678	718
32	35	achte	3494.295	691	718
33	36	denn	2264.747	707	722
34	38	noch	1364.073	718	730
35	38	noch	1348.950	718	730
35	40	November	3774.550	718	755
36	37	um	1189.574	722	729
36	41	um	3282.464	722	752
37	39	so	1154.407	729	738
38	39	so	932.200	730	738
38	42	am	4330.507	730	767
39	42	noch	3028.057	738	767
39	42	macht	3026.784	738	767
39	42	machen	2988.040	738	767
40	42	auch	1542.277	755	767
40	42	acht	1435.377	755	767
41	42	acht	1347.861	752	767

END_LATTICE

Abbildung 5: Ausschnitt aus einem Worthypothesengraphen

Probleme:
Die Anforderung der lückenlosen Nummernvergabe für die Knoten kann nicht erfüllt werden. (Im Beispiel wurde die erstellte Lattice nachbearbeitet!). Der inkrementelle Aufbau des Graphen mit den aufwendigen Pruning- und Rekombinationsmechanismen, sowie dem präventiven Einhängen und dem komplizierten Entscheidungsmechanismus zur Aufnahme der Hypothesen in die Lattice macht es unmöglich, schon während des Aufbaus der Lattice durchgehende Knotennummern zu vergeben. Auch die Sortierung nach Anfangs- und Endknotennummern ist nicht trivial. Durch die Vorgängersuche in bestimmten Framebereichen und das präventive Einhängen bei der Aufnahme in die Lattice kann die Sortierung durcheinandergeraten.

Abhilfe für beide Probleme schafft ein für den Offline-Betrieb akzeptables Verfahren, das nach Erzeugung der gesamten Lattice diese sortiert und eine lückenlose Knotennummerierung durchführt.

Sinnvoller ist es aber, zum Einsatz in einem inkrementellen System die „höheren“ Stufen dahingehend abzuändern, daß weder eine lückenlose Nummerierung noch eine Sortierung unbedingt notwendig sind.

4 Schnittstelle zwischen Suche und Lattice-Generierung

Zur Generierung eines Worthypothesengraphen werden Funktionen der Klasse **Clattice** verwendet**.** Diese ist in den Datei „jw_lattice.h“ und „jw_lattice.cc“ implementiert.

4.1 Die Schnittstellenklasse Clattice

Konstruktor *Clattice::Clattice*:
Clattice (Cword_list* word_list, long int ld_size, double lf_threshold, CRecoOut* resultout);

Die zu übergebenden Parameter haben folgende Bedeutung:

* word_list :	Zeiger auf die verwendete Wortliste,
ld_size :	Ringpuffergröße
lf_threshold:	Lattice-Pruningschwelle
* resultout:	Zeiger auf die Ausgabeklasse **CRecoOut**.

Mithilfe des Konstruktors wird ein Objekt vom Typ **Clattice** erzeugt. Da der Funktion zum Aufbau der Lattice nur die Wortnummern und nicht die Wörter selber übergeben werden, muß dem Lattice-Objekt ein Zeiger auf die verwendete Wortliste übergeben werden, damit aus den Wortnummern die Wörter bestimmt werden können. Die Ringpuffergröße gibt an, wieviele Hypothesen in den Puffer aufgenommen werden können. Diese Größe muß auf die Lattice-Pruningschwelle abgestimmt werden um zu gewährleisten, daß immer genügend lange Wortketten im Puffer enthalten sind. Mithilfe der Lattice-Pruningschwelle kann eingestellt werden, welchen Score die Hypothesen überschreiten müssen, um in den Puffer aufgenommen zu werden. Die Funktionen der Klasse **CRecoOut** (t_lat_out.h/.cc) sind für die Ergebnisausgabe zuständig.

Für die Schnittstelle zwischen Suche und Lattice-Generierung sind nur Funktionen der Klasse **Clattice** wichtig.

Funktion *Clattice::makelattice*:
char makelattice(JW_LATTICE_bigramtype* bigram);

Der Funktion *makelattice* muß ein Parameter vom Typ **JW_LATTICE_bigramtype** übergeben werden. Dieser ist ein struct mit folgender Struktur:

```
typedef struct
 {
 short hd_windex;               // Wortnummer
 short hd_vwindex;              // Vorwortnummer
 double lf_accscore;            // akkumulierter Score
 long ld_aframe;                // Anfangsframe
 long ld_eframe;                // Endframe
 double lf_best_accscore;       // bester akkumulierter Score zu diesem Zeitpunkt
}JW_LATTICE_bigramtype;
```

Der Funktion *makelattice* müssen die während der Suche entstehenden Wort-Vorwort-Paare mit Angabe des Anfangs- und Endframes sowie dem bis zu diesem Zeitpunkt aufakkumulierten Score übergeben werden.

In der Funktion *makelattice* werden neben der Verwaltung des Ringpuffers Routinen zur Synonymbehandlung, Vorgängersuche und Verknüpfung sowie Ausgabe der einzelnen Zeile der Lattice ausgeführt

Funktion *Clattice::endlattice*:
int endlattice()
Diese Funktion wird am Ende der Suche aufgerufen, genau dann, wenn keine weiteren Worthypothesen mehr vorhanden sind. Mithilfe dieser Funktion wird ein absoluter Endknoten definiert, an dem alle Hypothesen enden müssen.

4.2 Benötigte Source-Files

ak_const.h:
    Konstanten für die Lattice-Generierung, z.B. Frameradius R
ak_nt.cc/.h:
    Funktionen zur Verwaltung der Knotentabelle
jw_lattice.cc/.h:
    Funktionen zum Aufbau der Lattice, Realisierung der Schnittstelle
t_lat_out.cc/.h:
    Funktionen zur Ergebnisausgabe in definiertes File
p_port.h:
    Ein-/Ausgabe-Konstanten
p_lex.h:
    Zu verwendende Wortliste

5 Ergebnisse

Wie in Abschnitt 2.1 deutlich gemacht wurde, ist es das Ziel, eine möglichst hohe Worterkennungsrate zu erreichen, während gleichzeitig die Worthypothesendichte (WHD) klein gehalten werden soll.

Die Breite des Worthypothesengraphen und damit letztlich auch die Worthypothesendichte hängt in erster Linie von der Wahl der Pruningschwelle auf Wortebene ab. Diese sogenannte Lattice-Pruningschwelle hat einen entscheidenden Einfluß auf die Breite der Lattice. Interessant ist der genaue Zusammenhang zwischen dieser Pruningschwelle und der Worterkennungsrate einerseits, sowie der Zusammenhang zwischen dieser Pruningschwelle und der Worthypothesendichte. Daraus ist ein Zusammenhang zwischen Worterkennungsrate und Worthypothesendichte ableitbar.

Klar ist aber auch, daß die Worterkennungsrate von der Pruningschwelle auf Zustandsebene, also dem Pruningfaktor, abhängig ist. Zu ermitteln ist also der Zusammenhang zwischen Pruningfaktor, Latticepruningschwelle und Worterkennungsrate sowie Worthypothesendichte.

Um quantitative Aussagen über die Worterkennungsrate machen zu können, genügt es aber nicht einige wenige Sätze zur Erkennung zu verwenden. Es muß ein ausreichend großes, repräsentatives Testset untersucht werden. Sprachdaten von mehreren Sprechern sind erforderlich. Die Durchführung von Erkennungsläufen mit mehreren hundert Testsätzen und verschiedenen Kombinationen von Pruningfaktor und Latticepruningschwelle ist noch nicht abgeschlossen.

Der Wahl von Pruningfaktor, Latticepruningschwelle und Puffergröße kommt eine besondere Bedeutung zu. Bei kleinen Pruningfaktoren, also großem Suchraum, muß entweder die Latticepruningschwelle besonders streng oder der Puffer entsprechend größer gewählt werden, um zu gewährleisten, daß immer noch Nachfolgerketten der Länge K (hier: K=2) im Puffer enthalten sind. Bei großem Suchraum werden logischerweise mehr Wortübergänge und somit mehr Wortfolghypothesen pro Suchschritt entstehen und an die Schnittstelle zum Lattice-Erzeugungsmodul gelangen. Die Einstellung der Parameter läßt sich komfortabel dadurch realisieren, daß während der Lattice-Erzeugung Statistiken über auftretende Ereignisse geführt werden. Solche Ereignisse sind z.B. das Nichtfinden eines „richtigen“, also der Wort-Bigramm-Bindung entsprechenden, Vorgängers, das Nichtfinden irgendeines Vorgängers, das Prunen von ankommenden Hypothesen und das Finden von Hypothesen ohne

Nachfolgerkette der Länge K. Anhand der Häufigkeit dieser Ereignisse kann darauf geschlossen werden, ob der Puffer groß genug gewählt worden ist.
Werden z.B. viele Hypothesen ohne Nachfolgerkette der Länge K gefunden, so deutet dies auf eine zu kleine Puffergröße für die gewählten Pruningparameter hin. Anhand der Anzahl der geprunten Hypothesen kann eine Aussage darüber getroffen werden, ob durch das zusätzliche Pruning zu viel Information verlorengeht und die zusätzliche Pruningschwelle vielleicht anders gewählt werden sollte.

Die Ermittlung von günstigen Kombinationen aus den verschiedenen Pruningschwellen und der Puffergröße muß ebenfalls über Testläufe über viele Testsätze erfolgen, da die Suchräume verschiedener Sätze stark untereinander variieren. Dieser Prozeß ist noch nicht abgeschlossen. Im Rahmen dieses Berichts wird eine kurze Untersuchung über die Abhängigkeit der Worthypothesendichte (WHD), der Perplexität (PP) und der Worterkennungsrate (WE) von der Wahl der Lattice-Pruningschwelle und der Puffergröße durchgeführt. Hierbei werden die Pruningparameter des Suchraumes konstant gehalten, um deren Einfluß ganz von den Auswirkungen der Lattice-Parameter zu trennen.

Testläufe mit zehn definierten Sprachdatenfiles bilden die Basis für diese Untersuchungen. Die im Offline-Betrieb vorgenommenen Testläufe können wegen der kleinen Anzahl an Sätzen nicht als repräsentativ angesehen werden. Aus den gewonnenen Daten können aber Trends abgelesen werden.

Die Testläufe lieferten folgende Ergebnisse in Bezug auf Worthypothesendichte, Perplexität und Worterkennungsrate. Dargestellt sind die zu untersuchenden Größen in Abhängigkeit von der Lattice-Pruningschwelle und der Puffergröße. Die Puffergröße wurde dabei so eingestellt, daß im Puffer immer genügend lange Hypothesen enthalten waren. Die Perplexität wurde, um eine sinnvolle Darstellung aller drei Größen in einem Diagramm zu ermöglichen, vorab mit dem Faktor 10 multipliziert.

Es wurden Untersuchungen mit mittelgroßem und großem Suchraum durchgeführt. Zur Einstellung des Suchraumes wird der Pruningfaktor (PF) sowie die Histogrammpruning-Parameter minimale Zustandsanzahl (min) und maximale Zustandsanzahl (max) verwendet. (siehe [2])

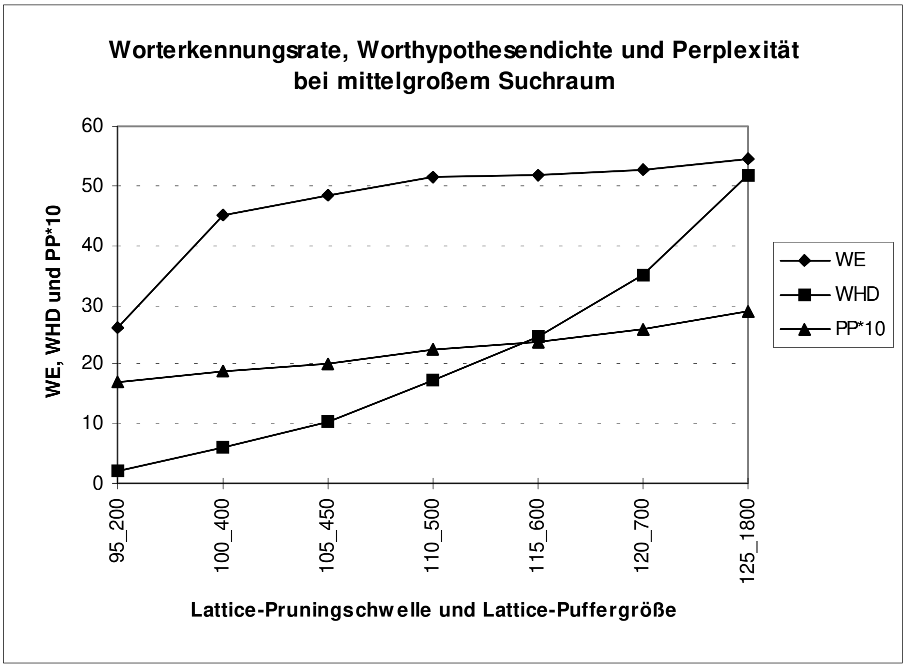


Abbildung 6: Worthypothesendichte, Perplexität und Worterkennungsrate bei mittelgroßem Suchraum (PF -60, min 5, max 10000)

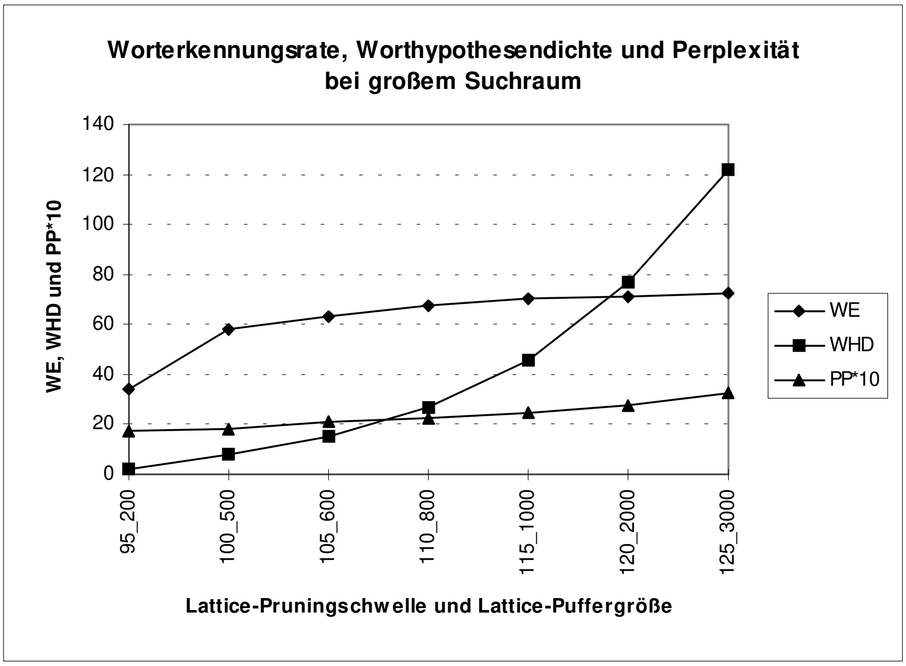


Abbildung 7: Worthypothesendichte, Perplexität und Worterkennungsrate bei großem Suchraum (PF -70, min 5, max 20000)

Allgemein weisen die Kurven für beide Suchraumgrößen die gleichen Charakteristika auf:

- Die Worterkennungsrate nimmt mit steigender Hypothesenanzahl zu (weniger strenges Pruning auf Wortebene, d.h. steigende Werte der Lattice-Pruningschwelle) und geht schließlich in Sättigung.
- Die Perplexität nimmt mit steigender Hypothesenanzahl ungefähr linear mit der Lattice-Pruningschwelle zu.
- Die Worthypothesendichte nimmt mit steigender Hypothesenanzahl stärker als linear in Bezug auf die Lattice-Pruningschwelle zu.

Unterschiede ergeben sich in Bezug auf die Worterkennungsrate bei unterschiedlicher Suchraumgröße. Bei größerem Suchraum ergibt sich eine höhere Worterkennungsrate, wobei die hier erzielten Erkennungsraten nicht als repräsentativ angesehen werden dürfen, da sie anhand eines Testlaufs mit 10 Sprachdatenfiles ermittelt wurden. Die Worterkennungsraten im Lattice-Modus liegen deutlich über den im 1-best Modus erzielten Worterkennungsraten bei gleichen Suchraumgrößen und bei Verwendung der selben Sprachdatenfiles.

Worterkennungsraten im 1-best Modus:

Suchraumgröße	**Worterkennungsrate**
PF -60, min 5, max 10000	44,8%
PF -70, min 5, max 20000	50,7%

Beim Vergleich der im Lattice-Modus und im 1-best-Modus erzielten Worterkennungsraten zeigt sich, daß bei sehr streng gewähltem zusätzlichen Lattice-Pruning die Worterkennungsraten unter die Worterkennungsraten im 1-best-Modus fallen. Dies muß durch sinnvolle Einstellung der Lattice-Pruningschwelle verhindert werden, da es ein grundsätzliches Ziel ist, daß der im 1-best-Modus als am besten erkannte Satz auch in der Lattice enthalten ist; diese Einstellung läßt sich am besten experimentell ermitteln. Bei geringer Latticetiefe und inkrementeller Erzeugung der Lattice kann aber grundsätzlich nicht garantiert werden, daß die 1-best-Erkennung tatsächlich immer enthalten ist.

Für eine Offline-Evaluierung läßt sich das Verfahren aber leicht modifizieren, indem die 1-best-Erkennung nachträglich in die Lattice mit aufgenommen wird und damit einem Offline-Verfahren gleichwertig ist.

6 Literatur

[1] E.Nöth, B.Plannerer, *Schnittstellendefiniton für den Worthypothesen-graphen*, Verbmobil-Memo-2-94, Universität Erlangen-Nürnberg und TU München, Dezember 1993

[2] T.Pfau, *Entwurf und Implementierung eines onlinefähigen automatischen Spracherkennungssystems*, Diplomarbeit am Lehrstuhl für Mensch-Maschine-Kommunikation, TU München, Februar 1996