A Parsing Strategy in ARCOS-G

Schamai Safra, TIK, ETH Zurich

COST 249 meeting of 12/13 Feb 1998

1 Introduction

In the ARCOS-G project experiments are conducted in a new approach to Continuous Speech Recognition where basic elements are detected by (traditional) statistical means like HMMs or ANNs, but then, putting the resulting basic element hypothesis together into the desired solution is done by means of chart parsing according to linguistic rules, e.g. lexi-con, grammar, pronunciation rules etc. A more detailed elaboration on the chart parsing approach was given in my contribution “Chart parsing in Continuous Speech Recognition” at the Kosice meeting, 1996.

The basic element detector has to deliver many more hypotheses than are expected in the final solution because if it would exercise too harsh a selection, the risk of losing correct elements prematurely would be too high. On the other hand, having many concurring basic elements allows for innumerable basic element combinations as possible global recognition solutions. Therefore, an exhaustive search through the search space is prohibitive. A search strategy is needed so only a small portion of search space has to be examined with a high probability of finding there the desired solution.

In the following – after a short recapitulation of how basic element detection is combined with chart parsing in the ARCOS-G project – experiments and results with a simple search strategy approach are described.

2 Chartparsing on Speech elements

2.1 “Non-Segmental” Basic Elements

Using segmental hypotheses delivered by a phonetic decoder directly as terminal edges of a chart does not allow to fully exploit the “find once – use many times” paradigm of chart parsing.

Suppose that segmental hypotheses coming from the following phoneme lattice were to be used:

```
[ a: r n s ]
[a: n t s]
```

Then the chart – if considering all suggested boundary hypotheses – would contain many superfluous paths, with many paths corresponding to the same phoneme sequence:

Instead, in our approach, “similar” segments are merged into one basic element which is represented in the chart as a (labelled) node, and each such basic unit is assigned an ‘inherent’ score corresponding to the “average” of merged segments’ scores. Adjacency hypotheses connect nodes probable to be neighbours. These so-called “joins” are represented in the chart as edges. The joins carry a score of their own, being a correction term that take into account the optimal boundary within the context of the two basic elements joined together. The chart initialized with just the basic element hypotheses looks then as follows:

Jean-Pierre Martens and his crew from ELIS (Gent University) were so kind to train one of their phonoacoustic decoders
with our material (in German) and to supply us with the segmental hypotheses produced by that decoder when applied on our test material. In our experiment presented further on, we use these phonetic unit lattices as input to our chart parser.

The following graphics shows an example of such an input lattice. The segments labelled with [a] are represented as black boxes. Indeed, on both places where an [a] occurs in the signal, many [a]-hypotheses are generated that only slightly differ in segment boundaries.

After applying our algorithm to merge similar segments into basic units, we end with about half as many basic units as the original segments. Doing an exhaustive parse on these (merged) basic elements, we end up with about 35% as many chart edges than if we took every segment to be one basic element, and the time for that exhaustive parse is also reduced to about 25%.

2.2 Scoring of Constituents

In order to easily integrate scoring into the chart parsing mechanism, we established the following simple scoring scheme:

- Depending on what probabilities or other scores your phonoacoustic decoder delivers, recalculate from them a length dependent additive cost. E.g. if the probability $p_f(c)$ of frame $f$ to be of class $c$ is given, then $- \sum_f \log p_f(c)$ would be such a cost.
- Each constituent edge has its own length and cost, both being additive, and thus a “score vector” can be formed: $\text{score} = (\text{length, cost})$
- combination of edges is then done by simple vector addition:

\[
\text{score}_{\text{comb}} = \text{score}_{\text{part}_1} + \text{score}_{\text{join}} + \text{score}_{\text{part}_2}
\]

. The score of the “join” consists of the length correction (bridging a small gap or overlap) and a cost correction which takes into account the optimal segment boundary between the two edges to be joined.

- Other quantities can be calculated from the score vector whenever needed, e.g. the average cost per length as a simple length-independent measure.

2.3 Stack Decoding (A*-Algorithm)

We wanted to test a parsing strategy that would speed up the finding of the correct result, i.e. if we had to stop the parsing before an exhaustive search could be accomplished, we would have an increased probability of the correct result already being among the edges found so far.

In chart parsing, the strategy can be regarded as the algorithm with which an agenda entry is selected to be processed next, or in other words, which of several edges should be tried to be expanded next.

The Stack-Decoding algorithm selects the one edge with the lowest estimated global cost, where the estimated global cost is the cost of the edge itself plus the estimated cost of the rest of the utterance.

With this approach, edges are not regarded by themselves but their contribution as part of the whole solution is estimated, and thus edges can be compared even if they cover different segments of the signal.

Estimating the global cost is the real challenge of the stack decoding approach. (In general one can say that an optimistic estimate of the rest of the utterance leads to a better solution but saves less searching effort and vice versa.) We decided on using the simplest possible global cost estimate, namely a length proportional cost for the rest of the utterance, with some constant global cost density(CD), e.g. the noise model cost over the whole utterance divided by the utterance length:
3 Experiments

For our experiments we used our own “TELNUMBERS” Corpus with the following specifications:

- 60 Speakers
- 15 Phone numbers from every speaker, spoken in groups of up to 999. (e.g. “312-4-15”)
- recorded over different phone sets.

As acoustic-phonetic decoders we used a standard HTK recognizer (run in N-best mode to generate a lattice) and the ELIS/Gent segment classifier mentioned before (here, only a very short training could be accomplished to date and we had also fewer test utterances).

3.1 Morpheme Recognition Rate

In the first experiment, we measured the “basic recognition power” of the systems in that we put a morpheme hypothesis at any possible starting point, and thus let the parser detect each morpheme independently. Of course, the inter-morpheme coarticulation model of ARCOS was not helpful here.

The experiment resulted in 36% found morphemes with the HTK units and 29% with the ELIS units, the latter result being worse most probably because only preliminary training on a small subset of the data was done, here.

These very low rates which sound unusual for traditional systems with similar tasks can be explained by the fact that the basic element detection works with no lexicon or language model of any kind and that nothing was undertaken to bridge missing basic elements. It is a peculiarity of chart parsing that with no counter-measure one missing element spoils the whole parse.

3.2 Number of Actions to solution

The second experiment was conducted to validate the usefulness of the stack decoding strategy. For that end, the number of actions (agenda entries processed) that it took to find the correct solution was measured. This number for the stack decoding strategy was then compared with the same number when applying a trivial strategy. (E.g. FIFO)

As with a morpheme recognition rate as low as around 30%, a full parse was impossible in almost all test utterances, we had to “cheat” a bit in that the basic element lattices were “minimally enriched” to make a full parse possible. For that purpose, the actual lattice was compared to the result of a forced alignment with variants. (The comparison was done with a dynamic programming approach.) Resulting from that comparison was a minimal set of new segments that was added to the original lattice.

Two parameters were also varied to optimize the strategy:

- A score threshold for phoneme hypotheses was used to mimic variable productivity of the phonoacoustic decoders
- The cost density estimates were varied, making the length of a segment hypothesis more or less influencial.

With the best configuration, the stack decoding strategy needed 20% less actions than the trivial strategy.

4 Conclusions

- Stack Decoding as we tried it is significantly but not overwhelmingly better than the trivial strategy. Probably a more elaborate cost estimate could do a better job.

- Recognition gaps seem to be the crucial problem when applying chart parsing. Here are some ideas how to overcome it:
  - There is no reason why to keep the phonoacoustic decoder completely ignorant. In the HMM-decoder, for instance, using a recognition network that models a morpheme sequence grammar would ensure that only correct morphemes are found and thus certainly increase the probability of finding all the necessary phones while keeping the number of produced elements constant.
  - The decoder could be told to locally produce more hypotheses around a gap, but of course, the gap has to be detected, first.
  - It is also possible to modify the workings of the parser, e.g. by introducing “glue”-constituents in the grammar, that could bridge decoding gaps.