Search strategies
We can think of the generation task as a search problem: how do we find the output candidate that best satisfies the constraint ranking?

Depth-first searching

Benefit: you don’t have to remember too many forks in the road at once.

Breadth-first searching
Dynamic programming

Tesar (1995)
Tesar presents an example of how dynamic programming can choose the optimal candidate without considering an infinite set: basic syllabification.

Background
• Assumes containment: segments aren’t actually deleted or inserted.

  Deletion (“underparsing”) = failing to attach segment to any prosodic structure
  Violates PARSE

  Insertion (“overparsing”) = blank prosodic structure (featural values are “filled in” by the “phonetic component”)
  Violates FILL

• Context-free grammar for syllable structure:

  S => e | oO | nN
  O => nN
  N => e | dD | oO | nN
  D => e | oO | nN

  o = onset, n = nucleus, d = coda

  We could add, for the terminals:

  o => C
  n => V
  d => C
Candidates are derived by constructing a tree for the input string of Vs and Cs. At any point in the derivation, we can be in one of four states: the nonterminal character is S, O, N, or D.

Tree construction is guided by Operations Set (p. 13, p. 25), which tells you what your options for the next step are.

*Algorithm*
Let’s step through how the algorithm would work for CCV (Tesar shows VC).

1. Set cell \([S, i_0]\) to blank: no structure, no violations
2. For each row \(i_0\) (proceeding from left to right)
3. { For each row X
4. { Fill \([X, i_j]\) with the result of the underparsing operation for X
5. For each parsing operation for X
6. { If the result of applying the operation is more harmonic than what’s currently in the cell, replace it.
7. } Until no cell entries change
8. { For each cell \([X, i_0]\)
9. { For each overparsing operation for X
10. { If the result of applying the operation is more harmonic than what’s currently in the cell, replace it.
11. }
12. }
13. }

Maximum number of passes through \#7 is three, because of the *cycle*:
. . . is same as . . , but with three extra violations of Fill.

*Complexity*
The maximum number of steps for each column is equal to:

\[
(1+\text{Under}_S)+(1+\text{Under}_O)+(1+\text{Under}_N)+(1+\text{Under}_D)+3(\text{Over}_S+\text{Over}_O+\text{Over}_N+\text{Over}_D) \\
= (4+\text{Under}_S+\text{Under}_O+\text{Under}_N+\text{Under}_D)+3(\text{Over}_S+\text{Over}_O+\text{Over}_N+\text{Over}_D)
\]

This is a constant (how big depends on the number of operations) so the maximum number of steps is “linear in the size of the input” (the length of the input string times a constant number). The maximum amount of cells to keep in memory is 8 (you can ignore row \(i_j\) once you’re at \(i_{j+2}\), because none of the operations require looking back more than one column).
**Simplifying assumptions**

**Locality**: The most nonlocal constraint is ONSET, which requires looking back from a nucleus to the previous position to see if it’s an onset. Can you think of some constraints that are more nonlocal?

**Cycles**: assumes that a whole blank syllable will never be inserted. What if you had a language requiring minimally disyllabic words?

**Research suggestions based on Tesar**
(for all of these, an actual computer implementation and a proof of correctness would be nice)

- Include feet in the prosodic structure, and do stress assignment.
- Remove the assumption that nuclei must be vowels, and add constraints like *P/X and *M/X.
- Allow complex onsets and complex codas to be considered (though constraints may rule them out).
- Under what circumstances is the locality assumption not met? Can the procedure still be made to work?
- Metathesis? Coalescence?

**Next time: Mapping output to input**

**For next time**

- **Exercise**: give a DPT (just like the one on Tesar’s page 18, showing structure, constraint violations, and which cell it comes from) for the sequence CCVV, using the ranking
  \[ \text{ONSET} \gg \text{NOCODA} \gg \text{FILL}^{\text{ONS}} \gg \text{PARSE} \gg \text{FILL}^{\text{NUC}}. \]
  Also give a standard tableau (the winner should be the same in both!!). You probably want to do the tableau first so that you can tell if something’s going wrong in your DPT.
- No reading. Catch up from last week if you need to.