CSCI203

Week 12 – Lecture B

Dynamic Music

- For a computer scientist, playing music (piano or guitar) is easy. ☺
- We simply need to solve the following problem:
  - Given *music*, an ordered sequence of N notes; and *fingers*, a set of F possible fingers we can use:
  - Find the best sequence of f_i where f_i is the finger we use to play note $music_i$.
- To do this we need a transition function, d, where $d(p,f,q,g)$ is the difficulty of playing note p with finger f followed by note q with finger g.
- We will attempt to solve this problem using dynamic programming the (by now) familiar way.

Dynamic Programming VIII
Music made easy

A DP Formulation

- Sub Problems:
  - How best to play the rest of the music;
  - Suffixes of the form *music*$_{i\ldots}$
- Guess:
  - Which finger do I play note *i* with?
- Recursion:
  - ```
    DP(i) = min(
      for f in F
            DP(i+1) +d(note(i),f,note(i+1),?)
      rof )
    end DP
    ```
- We have a problem:
  - We have no knowledge of which finger we will use for the next note!
    ```

A DP reFormulation

- We need to expand our sub-problems:
  - How best to play *music*$_{i\ldots}$ becomes;
  - How best to play *music*$_{i\ldots}$ when we play *music*$_i$ with finger *f*.
- Our guess now becomes:
  - Which finger do we use to play note *i*+1
- Our recursive problem now looks like this:
  - ```
    DP(i,f)=min(
     for g in F
         DP(i+1,g)+d(music(i),f,music(i+1),g)
     rof )
    end DP
    ```
    ```

Onwards

- Now, we can solve the problem by memoizing DP and computing bottom up:
- ```
  for f in F
      DP(n,f)=0
  for i in n-1..1
      for f in F
          DP(i,f) = min (
              for g in F
                  DP(i+1,g)+d(i,f,i+1,g)
              rof )
      rof
  rof
  ```
- The overall solution is the minimum value of **`DP(1,f)`** over the *F* possible choices for the starting finger.
  ```

Analysis

- The number of sub-problems is $N \times F$.
- The time per sub-problem is $O(F)$.
- The overall problem is:
  - # sub-problems × time per sub-problem;
  - $N \times F \times O(F)$;
  - $O(N \times F^2)$.

Generalizing the Solution

- The solution we have developed is good for monophonic music:
  - Music in which a single note is played at a time.
- Most music is polyphonic:
  - Multiple notes are played at the same time.
- For a single instrument, the maximum number of simultaneous notes has an upper limit:
  - For piano, *F*, the number of fingers;
  - For guitar, *S*, the number of strings.
- The tune now consists of an ordered sequence of chords, sets of notes, which must be played together at each time step.

Generalizing the Solution

- With this extended problem the solution involves finding a sequence of fingerings:
- An allocation of specific fingers to specific notes at each interval.
- This can be represented by an ordered sequence where, for each finger in F, we record either the note in $chord_i$ it is playing or null.
- For a given chord, there are $O((F+1)^F)$ possible fingerings.
  - This is exponential in F but, at least for humans, F is a small integer.
- The dynamic programming solution now generalizes as follows:

Generalizing the Solution

- Sub-problems:
  - Given a fingering, *f*, for chord $music_i$, find the optimum sequence of fingerings to play the remainder of the tune, $music_{i\ldots}$
- Guess:
  - Which fingering do we choose to play the next chord, $music_{i+1}$?
- We also need to generalize the difficulty function, *d*, between successive chords.
- Apart from these changes, the solution is reached in essentially the same way.

Analysis

- The number of sub-problems is $N \times (F+1)^{F.}$
- The time per sub-problem is $O((F+1)^F)$.
- The overall problem is:
  - # sub-problems × time per sub-problem;
  - $N \times (F+1)^F \times O((F+1)^F)$;
  - $O(N \times (F+1)^{2F})$.
- Once again this solution is linear in the length of the tune, even though it is exponential in the, probably finite, number of fingers *F.*

Further Complications

- If we are playing a guitar, the same note may possibly be played in more than one way:
  - For example, G can be played:
    - On the open 4^{th} string;
    - On fret 4 of the 5^{th} string;
    - On fret 9 of the 6^{th} string.
- This adds more possibilities to the choice of fingering for a given chord.
- Now, for each finger we need to decide which note it is playing and where it is playing it.
- The fingering now has $O(((F+1)\text{x}S)^F)$ possible arrangements.
- Even with this complication the solution is still linear in N, the number of notes.

Further Extension

- As we increase the number of fingers (and strings) the problem becomes more complex.
- The final problem is left as an exercise for the student. ☺