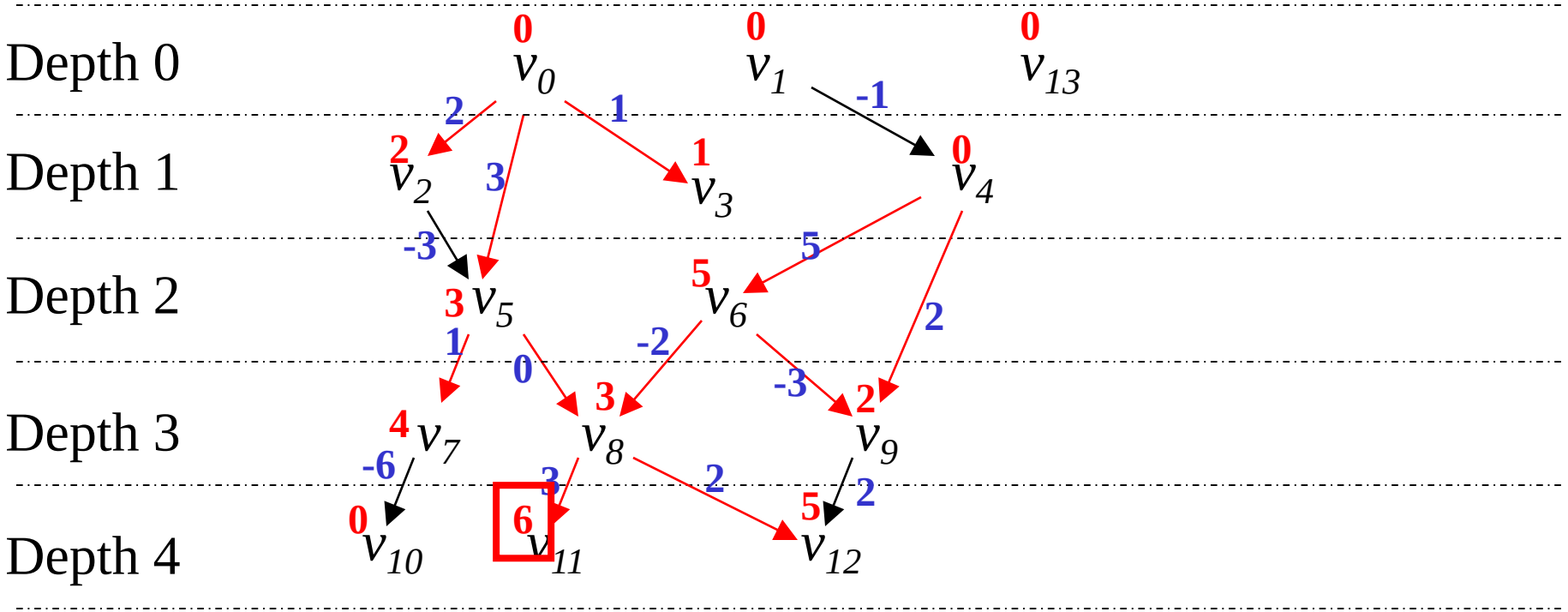
Today's Lecture

- (Dynamic programming)

- Weighted linked lists
  - Applications: Sequence graphs, "motif clusters", numerical data
  - Statistical issues
  - Finding multiple high-scoring paths/segments

Dynamic programming on WDAGs



Implementing Dynamic Programming in a Computer Program

- Storing entire graph has space complexity = $O(|V|+|E|)$
- If graph has regular structure, can often "create" and process vertices and edges on the fly, without storing in memory
  - cf. edit graph (to be defined later) for aligning sequences

Same dynamic programming approach can be used to find:

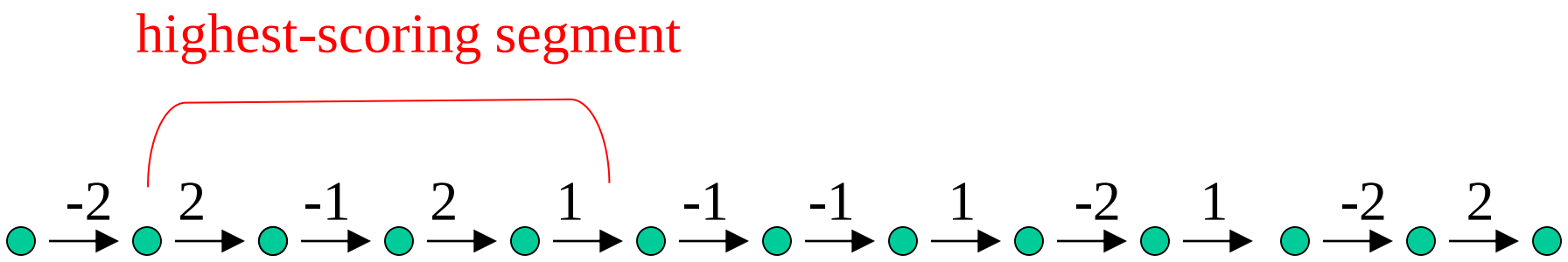
1. Highest product weight path (if weights are ≥ 0)
2. Highest weight path that
   - *starts* in particular subset V' of vertices,
     - don't consider paths that start outside V' :

       i.e. when computing $w(v)$, don't consider trivial path unless $v \in V'$
   - and/or *ends* in particular subset V''
     - only scan for the maximum $w(v)$ over V''
3. Sum of product weights of all paths ending at particular vertex
   - *sum* over all edges coming into v, instead of *maximizing*
   - this useful for probability calculations

- Will use the above variants later!

Weighted Linked Lists (WLLs)

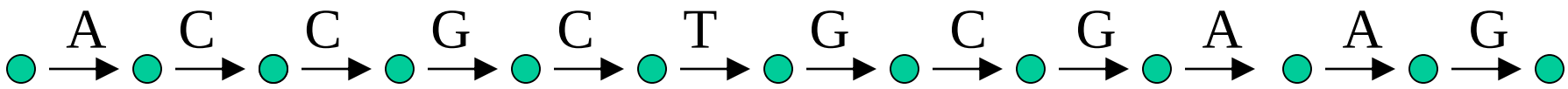
- *WLL* is linked list with weights on each edge
  - simplest kind of WDAG.
- Highest weight paths correspond to highest-scoring segments of WLL.



- Find these segments by dynamic programming
  - Much better than “brute force” algorithm!
- Beginning & end of best path determine path uniquely, so
  - traceback is unnecessary
  - single pass through list suffices to find best path.

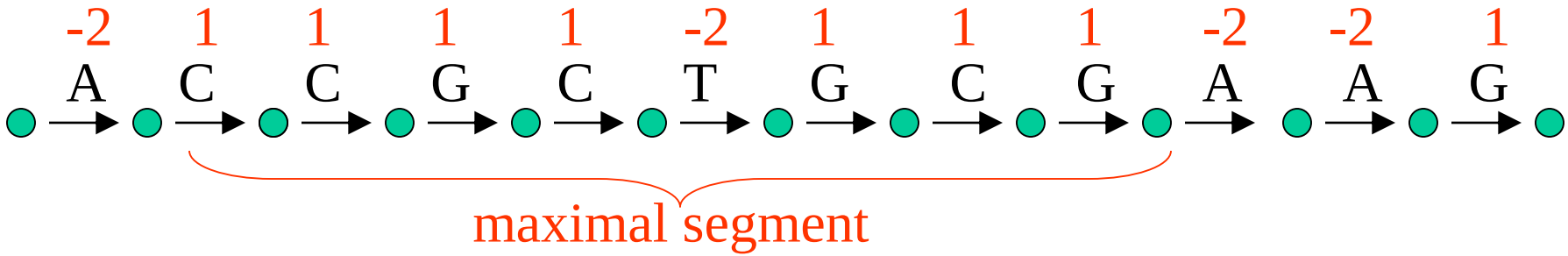
Applications to Sequences

- A *sequence graph* of a sequence is linked list whose edges are labelled by sequence residues (in order):
- e.g. graph for sequence ACCGCTGCGAAG is:



Weighted Sequence Graphs

- If attach weight to each residue, sequence graph becomes a WLL.



- Highest weight paths correspond to highest-scoring segments of sequence.
- Useful for identifying segments with “atypical composition”

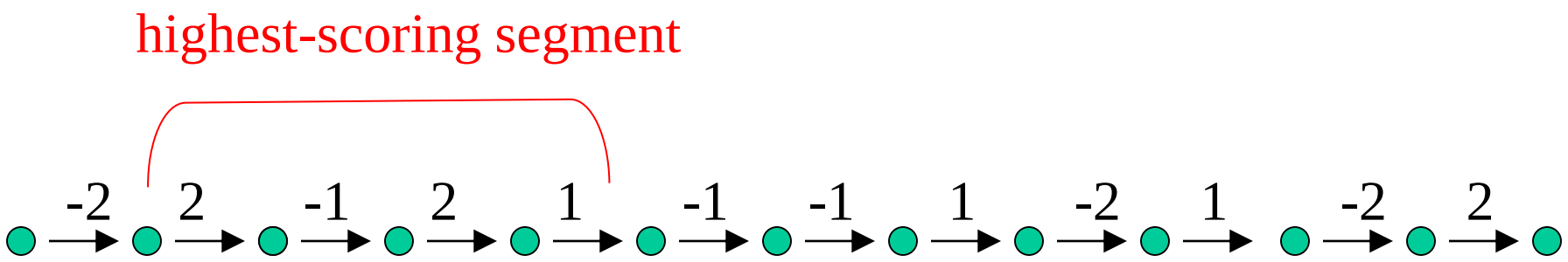
- For example:
  - Gives good way to find GC-rich regions in AT-rich thermophile genomes
    - generally correspond to RNA genes (Rob Klein & Sean Eddy)
  - AT-rich, purine-rich, pyrimidine-rich regions
  - Hydrophobic, acidic, or basic regions in protein sequences

- More broadly, can find regions enriched for sequence *motifs*:
  - CpG islands in mammalian genomes
    - positive weight (e.g. +17) to the first C of each CpG, and
    - negative weight (e.g. –1) to every other base

    (This approach was used in *Nature* human genome paper).
  - *horizontally transferred* regions
  - Regions rich in (known) transcription-factor motifs

Weighted Linked Lists (WLLs)

- *WLL* is linked list with weights on each edge
  - simplest kind of WDAG.
- Highest weight paths correspond to highest-scoring segments of WLL.



WLLs with non-sequence-based scores

- Can also assign scores to each genomic position based on other quantitative info:
  - Next-gen read frequency, e.g.
    - CNVs (Homework 3)
    - Hypersensitive sites
    - CHIP-seq
  - Other measurements?
- Attach scores to *columns* in sequence *alignments*

Crucial issues!

- What is best scoring system to detect the 'target regions'?
  - Short answer: $s(r) = \log(t_r / b_r)$ where
    - t_r , b_r are freqs of residue (or motif) r in target and background
    - (if unknown, can sometimes estimate iteratively)
- When is the score of a segment 'significant'?
  - (Karlin/Altschul) for s as above ($\log_e$), expected # segments of score $\geq S$ in random backgd seq of len N

$$\approx NK e^{-S}$$

for some constant K (not depending on S)

- Will revisit both issues later.

Finding *multiple* high-scoring segments

- In general, expect several regions of particular type in a given sequence – not just one!
- So want to find multiple high-weight paths in a WDAG
- But not interested in slight perturbations of previously found paths
- One strategy:
  - Find highest-weight path
  - 'Mask it' (remove its edges from graph)
  - Repeat above two steps until scores no longer 'interesting'

- Is there a more efficient algorithm not requiring repeated scans?
  - Ruzzo & Tompa solved for WLLs
  - ∃ solution for arbitrary WDAGs?