Outline

- data structures for MST and Dijkstra’s
  - union-find with log \( n \) analysis (finishing up)
  - amortized analysis: potential function method
  - Fibonacci heaps

Recall: amortized analysis

- amortized analysis:
  - each operation has an amortized cost
  - any sequence of operations has cost bounded by sum of amortized costs

1. Fibonacci heap amortized vs. binary heap
  - \textsc{extract-min} \( O(\log n) \) vs. \( O(\log n) \)
  - \textsc{insert}, \textsc{decrease-key} \( O(1) \) vs. \( O(\log n) \)

Potential function method

- \( n \) operations on initial data structure \( D_0 \)
  - \( D_i \) after i-th operation
- potential function \( \Phi(D_i) \) (real number)
- amortized cost of i-th operation w.r.t \( \Phi \):
  \[ c_i = c_i + \Phi(D_i) - \Phi(D_{i-1}) \]

Potential function method

- Key observation:
  \[ \sum_{i=1}^{n} c_i = \sum_{i=1}^{n} c_i + \Phi(D_n) - \Phi(D_0) \]
  - sum of amortized costs is an upper bound on sum of actual costs, provided:
    \[ \Phi(D_n) - \Phi(D_0) \geq 0 \]
  - will typically ensure \( \Phi(D_n) - \Phi(D_0) \geq 0 \) for all \( i \)
Potential function method

• Example: binary counter $C$ on $k$ bits
  – single operation: INCREMENT
    
    \[
    \begin{array}{c}
    1010100011101011011111101111111011000000 \\
    \rightarrow \quad 1010010001110111101000000
    \end{array}
    \]
  
  – worst case cost?

Potential function method

• Example: binary counter $C$ on $k$ bits
  – single operation: INCREMENT
    
    \[
    \begin{array}{c}
    1010100011101011011111101111111011000000 \\
    \rightarrow \quad 1010010001110111101000000
    \end{array}
    \]

• Potential function: $\phi(C) = \# \text{ of ones}$

• Consider $i$-th operation:
  – actual cost $c_i = (\# \text{ of ones set to zero}) + 1$
  – $\Delta \phi = (\phi(C_{i-1}) - t_i + 1) - \phi(C_{i-1}) = 1 - t_i$
  – so amortized cost $c_i = 2$

Potential function method

• Example: binary counter $C$ on $k$ bits
  – single operation: INCREMENT
    
    \[
    \begin{array}{c}
    1010100011101011011111101111111011000000 \\
    \rightarrow \quad 1010010001110111101000000
    \end{array}
    \]

• Starting with 0 counter on $k$ bits:
  – $\phi(C_0) = 0$, $\phi(C_i) \geq 0$ for all $i$
  – so total cost of $n$ operations is $2n$

• Starting with arbitrary value on $k$ bits:
  \[
  \sum_{i=1}^{n} c_i = \sum_{i=1}^{n} \phi(C_i) + \phi(C_0) \leq 2n + k
  \]

Fibonacci heaps

data structure

Kevin Wayne’s slides
based on CLRS text