Dynamic Programming

- (Weighted) Independent Set in Graphs
- Weighted Independent Sets in Path
- Dynamic Programming Formulation
- Implementation and Backtracking

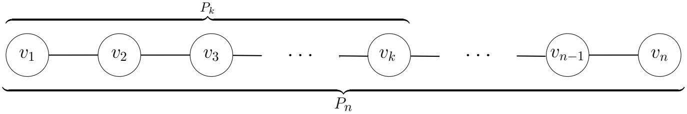
Imdad ullah Khan

Max weight independent set in path graph

Input: A node weighted path graph $P = (V, E)$, $w : V \to \mathbb{R}^+$
Output: An independent set of P of maximum weight

Notation: Think of P as a sequence of vertices $v_1, v_2, \ldots, v_n$

- P_k: The (sub)path induced by $v_1, v_2, \ldots, v_k$



- OPT-SET(k): An optimal independent set in P_k
- OPT-VAL(k): Value of an optimal independent set in P_k

Our goal is to find OPT-SET(n) and OPT-VAL(n)

Max weight independent set in path graph

Argue about structure of the solution (though we can't compute it)

See how solution is composed of that to smaller subproblems

- Either v_n is part of the optimal solution, OPT-SET(n)
  - w_n is counted in OPT-VAL(n)
  - Node v_{n-1} is not part of the OPT-SET(n)
  - Analyze solution to P_{n-2} $(v_1, v_2, \ldots, v_{n-2})$
- Or v_n is not part of the optimal solution, OPT-SET(n)
  - w_n is not counted in OPT-VAL(n)
  - Node v_{n-1} may or may not be part of OPT-SET(n)
  - Analyze solution to P_{n-1} $(v_1, v_2, \ldots, v_{n-1})$
- Analyze solution to P_{n-2} and P_{n-1}?

Max weight independent set in path graph

Analyze solution to P_{n-2} and P_{n-1}?

- $v_n \notin \text{OPT-SET}(n)$
  - $\text{OPT-SET}(n)$ is an independent set in P_{n-1}
  - $\text{OPT-SET}(n)$ is a maximum WIS in P_{n-1}
    - Let $A \subset P_{n-1}$ be an independent set with $wt(A) > \text{OPT-VAL}(n)$
    - A is also an independent set in P_n
    - It contradicts optimality of $\text{OPT-SET}(n)$
- $v_n \in \text{OPT-SET}(n)$
  - $v_{n-1} \notin \text{OPT-SET}(n)$
  - $\text{OPT-SET}(n) \setminus \{v_n\}$ is an independent set in P_{n-2}
  - $\text{OPT-SET}(n) \setminus \{v_n\}$ is a maximum WIS in P_{n-2}
    - Let $A \subset P_{n-2}$ be an independent set with $wt(A) > \text{OPT-VAL}(n) - w_n$
    - $A \cup \{v_n\}$ is an independent set in P_n
    - It contradicts optimality of $\text{OPT-SET}(n)$

A max WIS in P_n is $\begin{cases} \text{either a max WIS in } P_{n-1} \\ \text{or it is } v_n \text{ union with a max WIS in } P_{n-2} \end{cases}$

Max weight independent set in path graph

Optimal Substructure Property: "an optimal solution can be constructed from optimal solutions of subproblems"

A max WIS in P_n is $\begin{cases} \text{either a max WIS in } P_{n-1} \\ \text{or it is } v_n \text{ union with a max WIS in } P_{n-2} \end{cases}$

$$\text{OPT-VAL}(n) = max \begin{cases} \text{OPT-VAL}(n-2) + w_n & \text{if } v_n \in \text{OPT-SET}(n) \\ \text{OPT-VAL}(n-1) & \text{if } v_n \notin \text{OPT-SET}(n) \end{cases}$$

In general,

$$\text{OPT-VAL}(k) = \max \begin{cases} \text{OPT-VAL}(k-2) + w_k & \text{if } v_k \in \text{OPT-SET}(k) \\ \text{OPT-VAL}(k-1) & \text{if } v_k \notin \text{OPT-SET}(k) \end{cases}$$

We don't know bases cases and which branch to take

▷ i.e. the if condition cannot be evaluated

Max weight independent set in path graph

Recursion

- If $v_n \in \textsc{opt-set}(n)$, then recursively find max WIS in P_{n-2}
- If $v_n \notin \textsc{opt-set}(n)$, then recursively find max WIS in P_{n-1}

We don't know bases cases and which branch to take

$$\textsc{opt-val}(k) = \max \begin{cases} w_1 & \text{if } k = 1 \\ \max\{w_1, w_2\} & \text{if } k = 2 \\ \textsc{opt-val}(k-2) + w_k & \text{if } v_k \in \textsc{opt-set}(k) \\ \textsc{opt-val}(k-1) & \text{if } v_k \notin \textsc{opt-set}(k) \end{cases}$$

Try both branches and select the bigger one

Max weight independent set in path graph

$$\textsc{opt-val}(k) = \max \begin{cases} w_1 & \text{if } k = 1 \\ \max\{w_1, w_2\} & \text{if } k = 2 \\ \textsc{opt-val}(k-2) + w_k & \text{if } v_k \in \textsc{opt-set}(k) \\ \textsc{opt-val}(k-1) & \text{if } v_k \notin \textsc{opt-set}(k) \end{cases}$$

Algorithm Recursive $\textsc{opt-val}(n)$

function $\textsc{opt-val}(k)$ ▷ implements the above recurrence
  if $k = 1$ **then**
    return w_1
  else if $k = 2$ **then**
    return $\max\{w_1, w_2\}$
  else
    return $\max\{\textsc{opt-val}(k-1), \textsc{opt-val}(k-2) + w_k\}$