Generalizing to computing longest paths in DAGs: Directed Acyclic Graphs

- The Manhattan grid is a DAG
- Exemplify Topological Ordering: DAG for Dressing in the morning problem
Topological Ordering

- A numbering of vertices of the (directed edge) graph is called **topological ordering** of the DAG if every edge of the DAG connects a vertex with a smaller label to a vertex with a larger label.

- In other words, if vertices are positioned on a line in an increasing order of labels then all edges go from left to right.
Topological ordering

- 2 different topological orderings of the DAG
Heaviest Path in DAG Problem

- **Goal**: Find a heaviest path between two vertices in a weighted DAG

- **Input**: A weighted DAG $G$ with source and sink vertices

- **Output**: A heaviest path in $G$ from source to sink
Heaviest Path in DAG: Dynamic Programming

• Suppose vertex v has in degree k and predecessors \{u_1, u_2 \ldots u_k\}

• Heaviest path to v from source is:

\[ s_v = \max \left( s_{u_1} + \text{weight of edge from } u_1 \text{ to } v, s_{u_2} + \text{weight of edge from } u_2 \text{ to } v, \ldots, s_{u_k} + \text{weight of edge from } u_k \text{ to } v \right) \]

In General, compute by **increasing Topological Order**:

\[ s_v = \max_{u \in \text{Predecessors}(v)} (s_u + \text{weight of edge from } u \text{ to } v) \]
Computing the score for point $x$ is given by the recurrence relation:

$$ s_v = \max_{u \in \text{Predecessors}(v)} \left( s_u + \text{weight of vertex (u, v)} \right) $$

- **Predecessors** $(v)$ – set of vertices that have edges leading to $v$

**What is the running time for a DAG $G(V, E)$, where $V$ is the set of all vertices and $E$ is the set of all edges?**

**Answer:** $O(E + V)$ since each edge is evaluated once and each node traversed once.