Graph Theory

Graph $G = (V, E)$

- $V$: finite set of vertices/nodes
- $E$: finite set of edges

Undirected graphs:

- $f^U: E \rightarrow \binom{V}{2}$: unordered pairs of vertices in $V$

- $V = \{1, 2, 3, 4\}$

- $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$

- $f^U(e_1) = \{1, 3\}$
- $f^U(e_2) = \{2, 3\}$
- $f^U(e_4) = \{2, 3\}$
\[ \left| V \right| = n, \quad \left| E \right| = m \]

**Directed:**

\[ f : E \rightarrow \text{ordered pairs of vertices in } V \times V \]

\[ f_D(e_1) = (2,3) \]

\[ f_D(e_2) = (3,2) \]
**Def:** A (directed) path is a sequence of vertices and edges:

\[ v_1 e_1 v_2 e_2 v_3 \ldots \]

- undirected: \( e \rightarrow \{u, v\} \)
- directed: \( e \rightarrow (u, v) \)

- Can add restriction that no edge is repeated in the sequence.

- Can add restriction that no vertex is repeated in the sequence.
Complexity of algorithms

Algorithm: Program/code.

elementary\[+,-,/,\star\]
operations
comparisons

Running time of an algorithm on any input is \# number of elementary operations performed during the execution of the algorithm on this input.

Class of optimization problems.
Scheduling

V tasks
m machines
C[i] if parameter

Numerical parameters:

Linear Regression: $\mathbb{R}^n \times \mathbb{R}^d \rightarrow \mathbb{R}$

Dimension (\# features) $\rightarrow$ \# data pts.

$y_1, \ldots, y_d$

Numerical parameters

"Binary encoding": \# bits needed to encode the numerical data.

Parameterization of my class of problem:

$\mathcal{I}$ is the set of all instances
in my class.

\[ R : \mathbb{F} \rightarrow (2, \ldots, 2^p) \]

Given any algorithm \( A \) for the problem, we get a function

\[ R : \mathbb{F} \rightarrow \mathbb{N} \]

running time of the algorithm on that instance.

\[ F(2_1, \ldots, 2_n) = \text{maximum over all instances with parameters } 2_1 \ldots 2_n \]
of running time of A.

A is "better" than B if

\[ F_A \leq F_B + 2^i, \ldots, 2^n. \]

Polynomial time algorithm

\[ F^A \] is a polynomial function

of the parameters.

\[ m^2 \]

\[ f^A \leq m^3 n^2 \]

\[ O(f(2, \ldots, 2p)) \]

\[ C f(2, \ldots, 2p) \]

Strongly poly algorithm

\[ F_A \] is a poly. for \( f \) only the dimensional parameters.