MAX CUT is NP-Complete\textsuperscript{a}

- We will reduce \textsc{naeSat} to \textsc{max cut}.
- Given an instance $\phi$ of 3\textsc{sat} with $m$ clauses, we shall construct a graph $G = (V, E)$ and a goal $K$ such that:
  - There is a cut of size at least $K$ if and only if $\phi$ is \textsc{naeSat}satisfiable.
- Our graph will have multiple edges between two nodes.
  - Each such edge contributes one to the cut if its nodes are separated.

\textsuperscript{a}Garey, Johnson, and Stockmeyer (1976).

The Proof

- Suppose $\phi$'s $m$ clauses are $C_1, C_2, \ldots, C_m$.
- The boolean variables are $x_1, x_2, \ldots, x_n$.
- $G$ has $2n$ nodes: $x_1, x_2, \ldots, x_n, \neg x_1, \neg x_2, \ldots, \neg x_n$.
- Each clause with 3 distinct literals makes a triangle in $G$.
- For each clause with two identical literals, there are two parallel edges between the two distinct literals.
- No need to consider clauses with one literal (why?).
- For each variable $x_i$, add $n_i$ copies of the edge $[x_i, \neg x_i]$, where $n_i$ is the number of occurrences of $x_i$ and $\neg x_i$ in $\phi$.

The Proof (continued)

- Set $K = 5m$.
- Suppose there is a cut $(S, V - S)$ of size $5m$ or more.
- A clause (a triangle or two parallel edges) contributes at most 2 to a cut no matter how you split it.
- Suppose both $x_i$ and $\neg x_i$ are on the same side of the cut.
- Then they together contribute at most $2n_i$ edges to the cut as they appear in at most $n_i$ different clauses.
The Proof (continued)

- Changing the side of a literal contributing at most $n_i$ to the cut does not decrease the size of the cut.
- Hence we assume variables are separated from their negations.
- The total number of edges in the cut that join opposite literals is $\sum n_i = 3m$.
  - The total number of literals is $3m$.
- The remaining $2m$ edges in the cut must come from the $m$ triangles or parallel edges that correspond to the clauses.
- As each can contribute at most 2 to the cut, all are split.
- A split clause means at least one of its literals is true and at least one false.
- The other direction is left as an exercise.
MAX BISECTION

- MAX CUT becomes MAX BISECTION if we require that \(|S| = |V - S|\).
- It has many applications, especially in VLSI layout.
- Sometimes imposing additional restrictions makes a problem easier.
  - SAT to 2SAT.
- Other times, it makes the problem as hard or harder.
  - MIN CUT TO BISECTION WIDTH.
  - LINEAR PROGRAMMING TO INTEGER PROGRAMMING.

The Proof (concluded)

- Every cut \((S, V - S)\) of \(G = (V, E)\) can be made into a bisection by appropriately allocating the new nodes between \(S\) and \(V - S\).
- Hence each cut of \(G\) can be made a cut of \(G'\) of the same size, and vice versa.
BISECTION WIDTH

- **BISECTION WIDTH** is like **MAX BISECTION** except that it asks if there is a bisection of size *at most* $K$ (sort of MIN BISECTION).
- Unlike **MIN CUT**, **BISECTION WIDTH** remains NP-complete.
  - A graph $G = (V, E)$, where $|V| = 2n$, has a bisection of size $K$ if and only if the complement of $G$ has a bisection of size $n^2 - K$.
  - So $G$ has a bisection of size $\geq K$ if and only if its complement has a bisection of size $\leq n^2 - K$.

HAMILTONIAN PATH is NP-Complete\(^a\)

**Theorem 41** Given an undirected graph, the question whether it has a Hamiltonian path is NP-complete.

- Reduce 3SAT to HAMILTONIAN PATH.
- We skip the messy proof in the text.

\(^a\)Karp (1972).

TSP (D) is NP-Complete

**Corollary 42** TSP (D) is NP-complete.

- Consider a graph $G$ with $n$ nodes.
- Define $d_{ij} = 1$ if $[i, j] \in G$ and $d_{ij} = 2$ if $[i, j] \notin G$.
- Set the budget $B = n + 1$.
- If $G$ has no Hamiltonian paths, then every tour on the new graph must contain at least two edges with weight 2.
- The total cost is then at least $(n - 2) + 2 \cdot 2 = n + 2$.
- There is a tour of length $B$ or less if and only if $G$ has a Hamiltonian path.
3-COLORING is NP-Complete<sup>a</sup>

- We will reduce NAESAT to 3-COLORING.
- We are given a set of clauses $C_1, C_2, \ldots, C_m$ each with 3 literals.
- The boolean variables are $x_1, x_2, \ldots, x_n$.
- We shall construct a graph $G$ such that it can be colored with colors $\{0,1,2\}$ if and only if all the clauses can be NAESAT satisfied.

<sup>a</sup>Karp (1972).

---

Graph Coloring

- $k$-COLORING asks if the nodes of a graph can be colored with $\leq k$ colors such that no two adjacent nodes have the same color.
- 2-COLORING is in $P$ (why?).
- But 3-COLORING is NP-complete.

---

The Proof (continued)

- Every variable $x_i$ is involved in a triangle $[a, x_i, \neg x_i]$ with a common node $a$.
- Each clause $C_i = (c_{i1} \lor c_{i2} \lor c_{i3})$ is also represented by a triangle $[c_{i1}, c_{i2}, c_{i3}]$.
- There is an edge between $c_{ij}$ and the node that represents the $j$th literal of $C_i$. 
The Proof (concluded)

Suppose the clauses are NAE-satisfiable.

- Color node $a$ with color 2.
- Color the nodes representing literals by their truth values (color 0 for false and color 1 for true).
- For each clause triangle:
  - Pick any two literals with opposite truth values and color the corresponding nodes with 0 if the literal is true and 1 if it is false.
  - Color the remaining node with color 2.

The Proof (continued)

Suppose the graph is 3-colorable.

- Assume without loss of generality that node $a$ takes the color 2, $x_i$ takes the color 1, and $\neg x_i$ takes the color 0.
- A triangle must use all 3 colors.
- The clause triangle cannot be linked to nodes with all 1s or all 0s; otherwise, it cannot be colored with 3 colors.
- Treat 1 as true and 0 as false (it is consistent).
- Treat 2 as either true or false; it does not matter.
- As each clause triangle contains one color 1 and one color 0, the clauses are NAE-satisfiable.