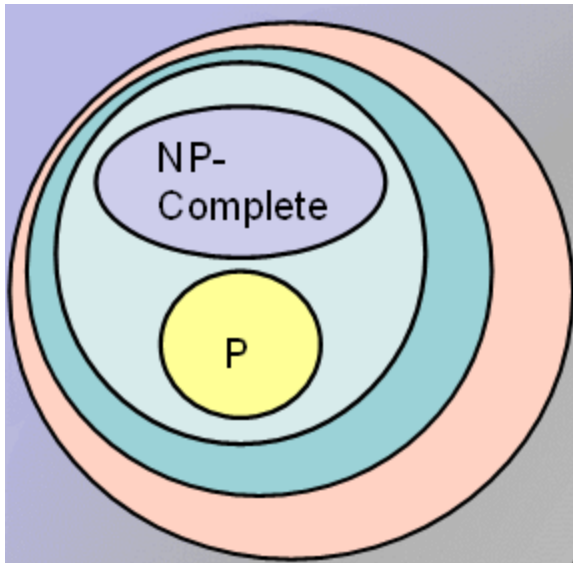


Lecture29



CSE 417

Algorithms and Complexity

Autumn 2024

Lecture 29

NP-Completeness and Beyond

Announcements

- Exam practice problems on course homepage
- Final Exam: Monday, December 9, 8:30 AM

Mon, Dec 2	NP-Completeness
Wed, Dec 4	NP-Completeness
Fri, Dec 6	NP-Completeness and Beyond
Mon, Dec 9	Final Exam

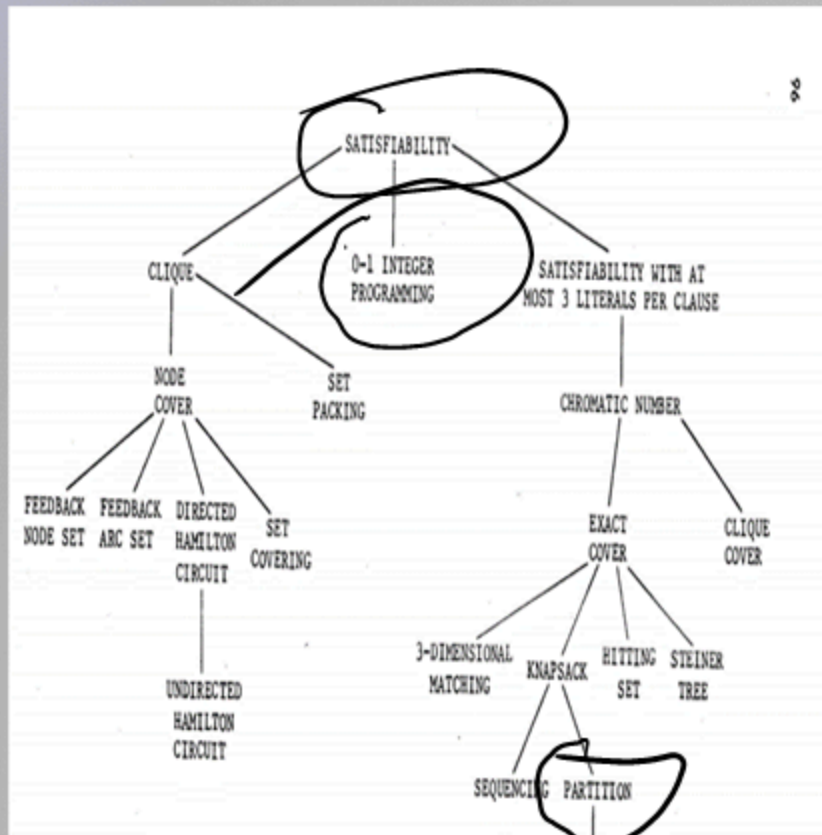
- This is my last lecture

NP-Completeness Proofs

- Prove that problem X is NP-Complete
 - Show that X is in NP (usually easy)
 - Pick a known NP complete problem Y
 - Show $Y <_p X$

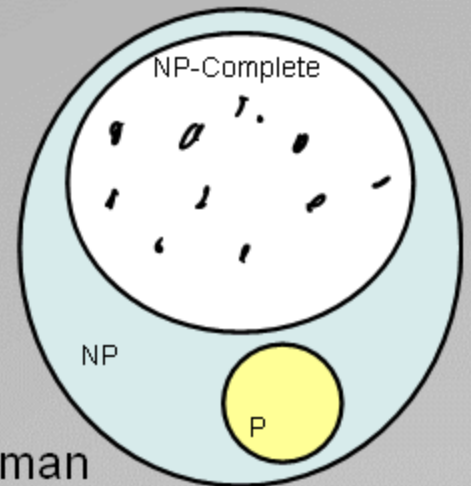


Reducibility Among Combinatorial Problems



Populating the NP-Completeness Universe

- Circuit Sat $<_P$ 3-SAT
- 3-SAT $<_P$ Independent Set
- 3-SAT $<_P$ Vertex Cover
- Independent Set $<_P$ Clique
- 3-SAT $<_P$ Hamiltonian Circuit
- Hamiltonian Circuit $<_P$ Traveling Salesman
- 3-SAT $<_P$ Integer Linear Programming
- 3-SAT $<_P$ Graph Coloring
- 3-SAT $<_P$ Subset Sum
- Subset Sum $<_P$ Scheduling with Release times and deadlines



Coping with NP-Completeness

- Approximation Algorithms
- Exact solution via Branch and Bound
- Local Search

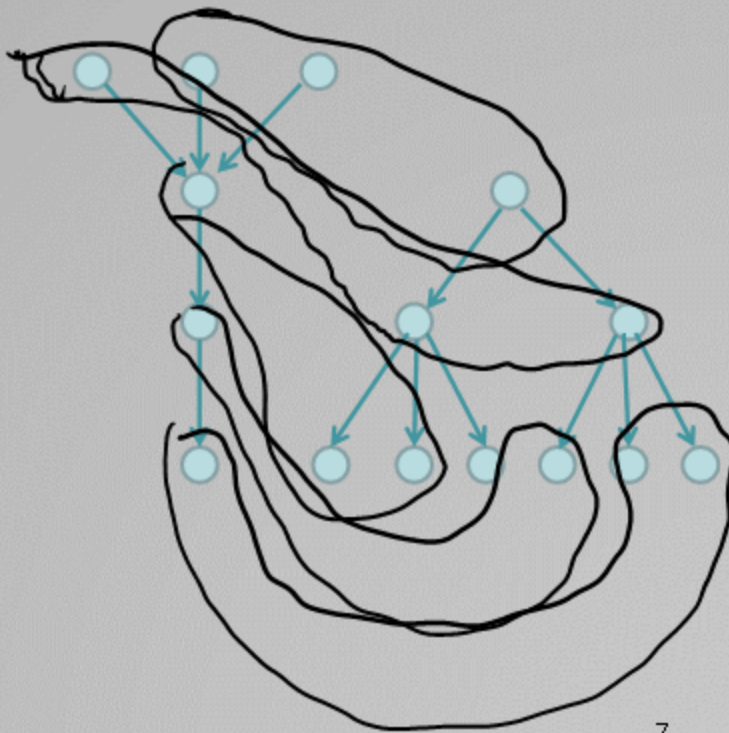


I can't find an efficient algorithm, but neither can all these famous people.

6

Multiprocessor Scheduling

- Unit execution tasks
- Precedence graph
- K-Processors
- Polynomial time for $k=2$
- Open for $k = \text{constant}$
- NP-complete if k is part of the problem



7

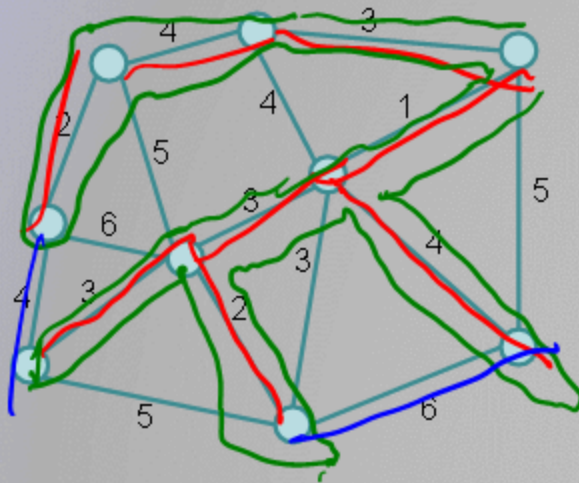
Highest level first is 2-Optimal

Choose k items on the highest level

Claim: number of rounds is at least twice the optimal.

Christofides TSP Algorithm

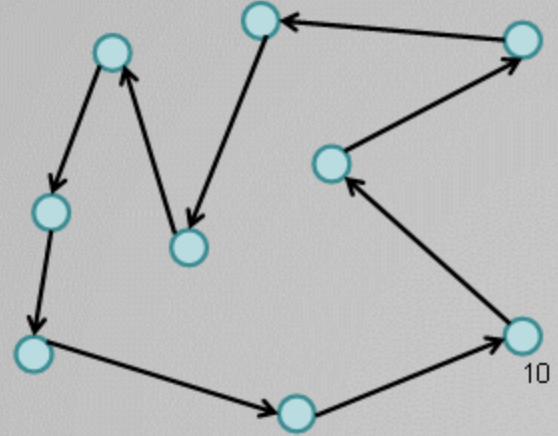
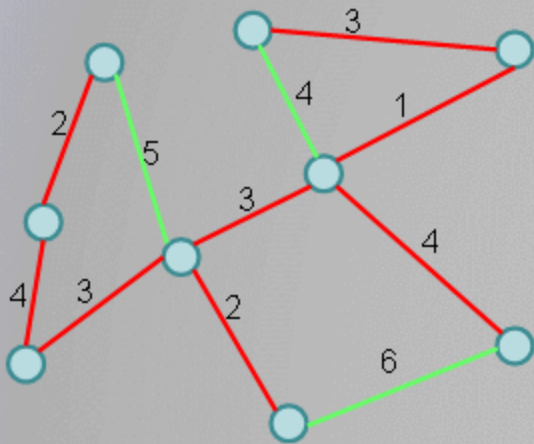
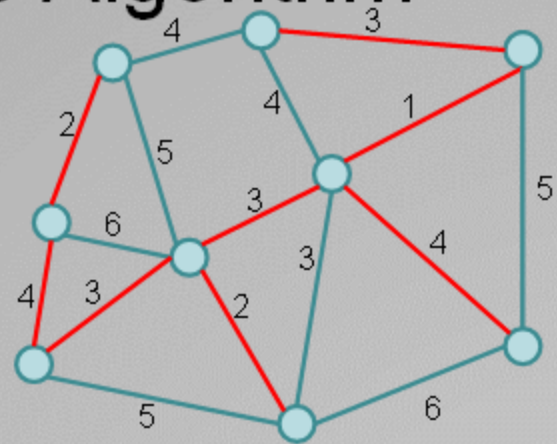
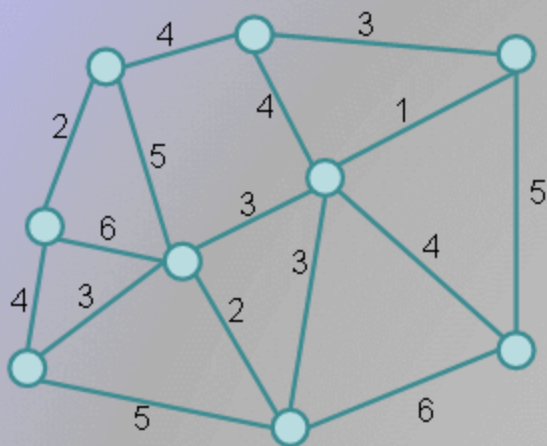
- Undirected graph satisfying triangle inequality



1. Find MST
2. Add additional edges so that all vertices have even degree
3. Build Eulerian Tour

3/2 Approximation

Christofides Algorithm



Branch and Bound

- Brute force search – tree of all possible solutions
- Branch and bound – compute a lower bound on all possible extensions
 - Prune sub-trees that cannot be better than optimal

Branch and Bound for TSP

- Enumerate all possible paths
- Lower bound, Current path cost plus MST of remaining points
- Euclidean TSP
 - Points on the plane with Euclidean Distance
 - Sample data set: State Capitals

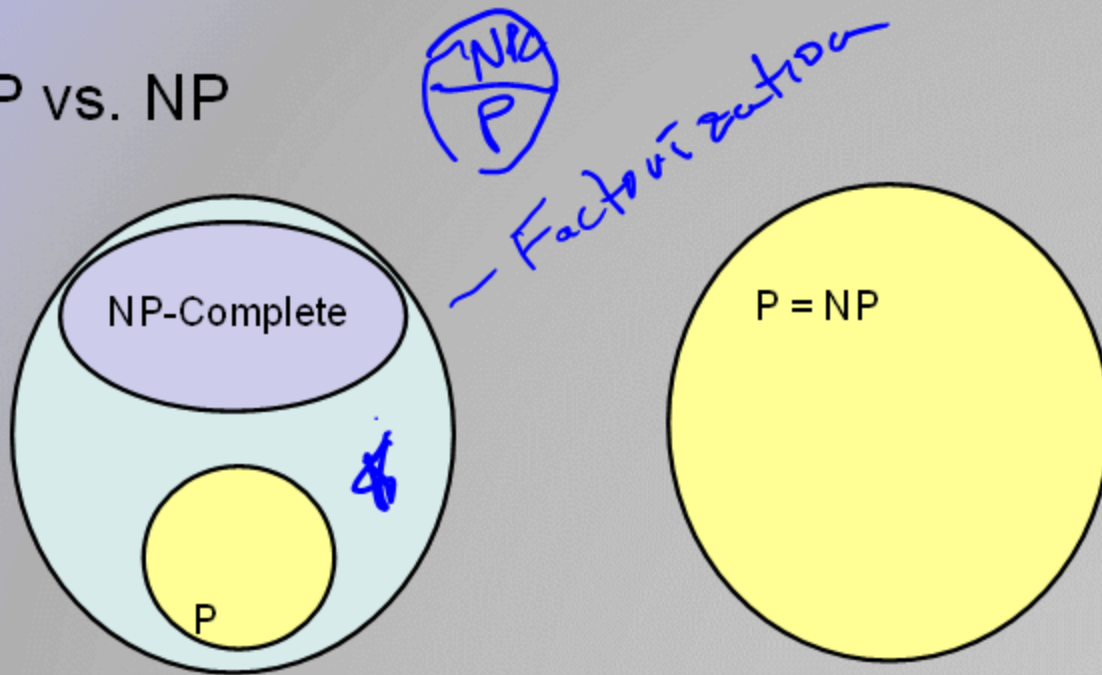


Local Optimization

- Improve an optimization problem by local improvement
 - Neighborhood structure on solutions
 - Travelling Salesman 2-Opt (or K-Opt)
 - Independent Set Local Replacement

What we don't know

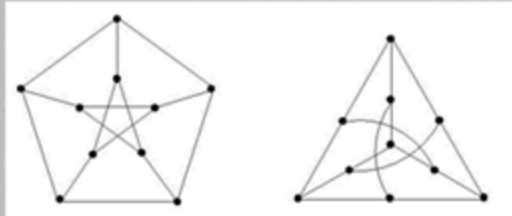
- P vs. NP



If $P \neq NP$, is there anything in between

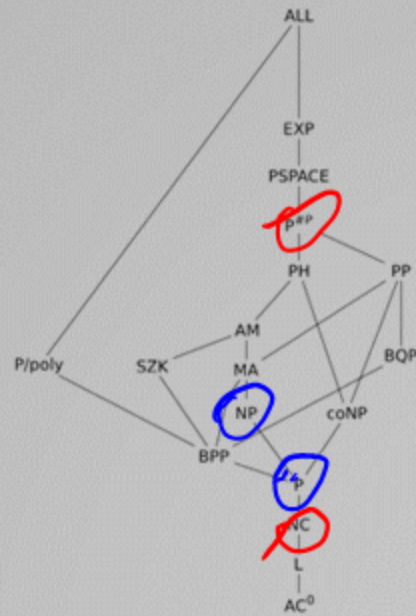
- Yes, Ladner [1975]
- Problems not known to be in P or NP Complete
 - Factorization
 - Discrete Log
 - Graph Isomorphism

Solve $g^k = b$ over a finite group



Complexity Theory

- Computational requirements to recognize languages
- Models of Computation
- Resources
- Hierarchies
- [//complexityzoo.net](http://complexityzoo.net)



Time complexity

- P: (Deterministic) Polynomial Time
- NP: Non-deterministic Polynomial Time
- EXP: Exponential Time

Space Complexity

- Amount of Space (Exclusive of Input)
- L: Logspace, problems that can be solved in $O(\log n)$ space for input of size n
 - Related to Parallel Complexity
- PSPACE, problems that can be required in a polynomial amount of space

Other types of computation

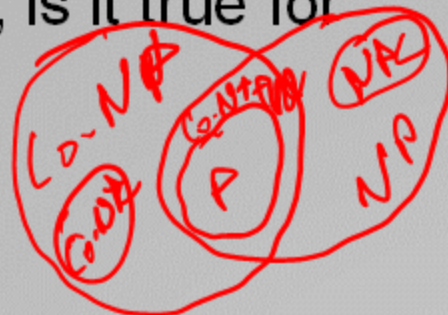
- **Randomization**
 - Can you solve problems faster with a random number generator?
- **Quantum Computers**
 - Can you solve problems faster if you have quantum bits which allow superposition?
 - Probably yes: Shor's Integer Factoring algorithm

So what is beyond NP?



NP vs. Co-NP

- Given a Boolean formula, is it true for some choice of inputs



- Given a Boolean formula, is it true for all choices of inputs

Problems beyond NP

- **Exact TSP**, Given a graph with edge lengths and an integer K , does the minimum tour have length K
- **Minimum circuit**, Given a circuit C , is it true that there is no smaller circuit that computes the same function as C

Polynomial Hierarchy

- Level 1

– $\exists X_1 \Phi(X_1)$, $\forall X_1 \Phi(X_1)$

- Level 2

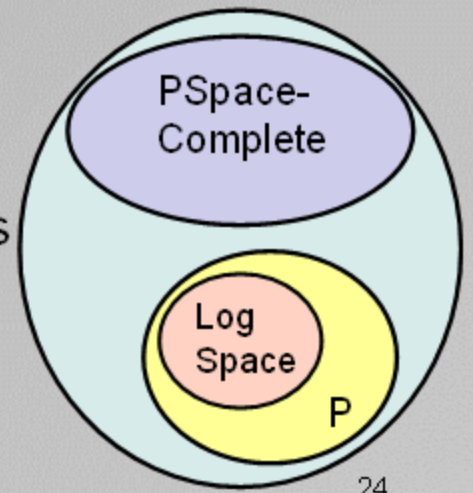
– $\forall X_1 \exists X_2 \Phi(X_1, X_2)$, $\exists X_1 \forall X_2 \Phi(X_1, X_2)$

- Level 3

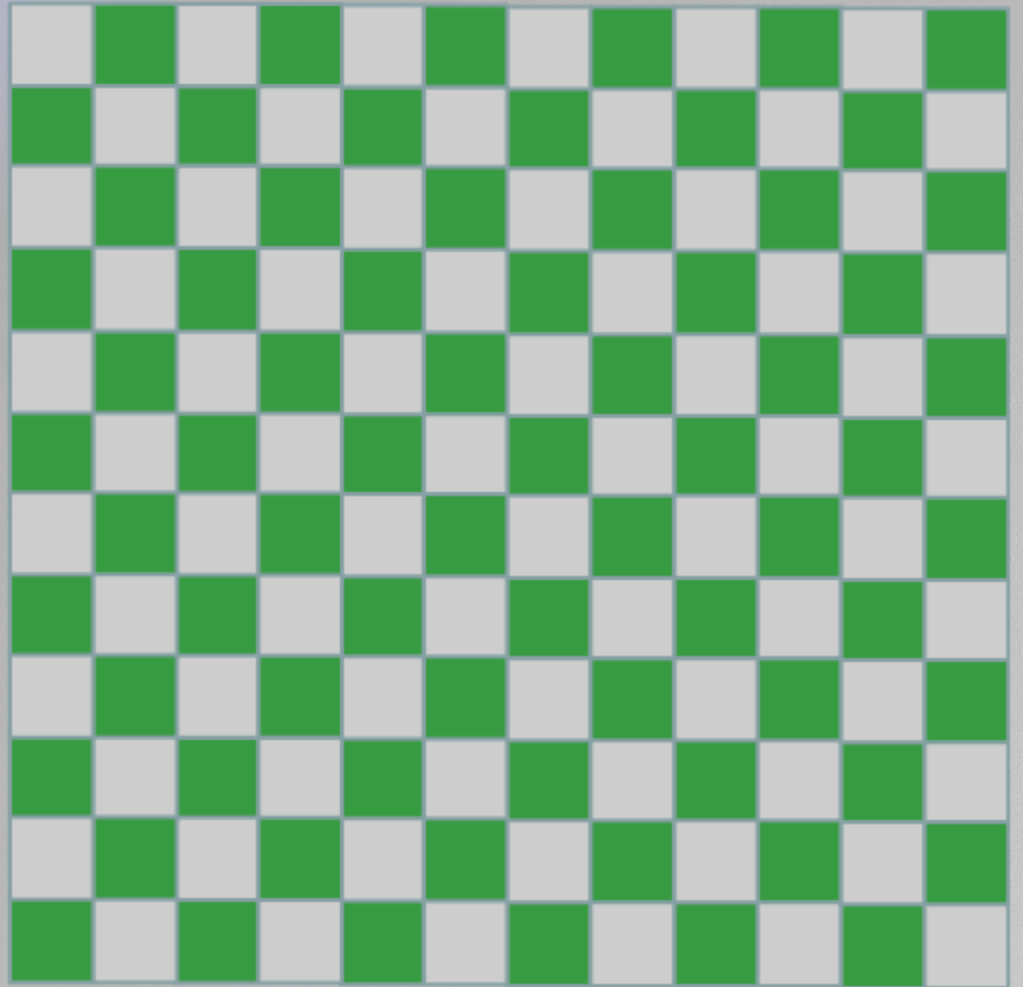
– $\forall X_1 \exists X_2 \forall X_3 \Phi(X_1, X_2, X_3)$, $\exists X_1 \forall X_2 \exists X_3 \Phi(X_1, X_2, X_3)$

Polynomial Space

- **Quantified Boolean Expressions**
 - $\exists X_1 \forall X_2 \exists X_3 \dots \exists X_{n-1} \forall X_n \Phi(X_1, X_2, X_3 \dots X_{n-1}, X_n)$
- **Space bounded games**
 - Competitive Facility Location Problem
 - N x N Chess
- **Counting problems**
 - The number of Hamiltonian Circuits



N X N Chess



Even Harder Problems

```
public int[] RecolorSwap(int[] coloring) {
    int k = maxColor(coloring);

    for (int v = 0; v < nVertices; v++) {
        if (coloring[v] == k) {
            int[] nbdColorCount = ColorCount(v, k, coloring);
            List<Edge> edges1 = vertices[v].Edges;

            foreach (Edge e1 in edges1) {
                int w = e1.Head;
                if (nbdColorCount[coloring[w]] == 1)
                    if (RecolorSwap(v, w, k, coloring))
                        break;
            }
        }
    }
    return coloring;
}
```

Is this code correct?

26

Halting Problem

- Given a program P that does not take any inputs, does P eventually exit?

Impossibility of solving the Halting Problem

Suppose $\text{Halt}(P)$ returns true if P halts, and false otherwise

Consider the program G :

What is $\text{Halt}(G)$?

$\text{Halt}(G) = \text{true} \Rightarrow \text{Halt}(G) = \text{false}$

```
bool G {  
    if (Halt(G)){  
        while (true) ;  
    }  
    else {  
        return true;  
    }  
}
```

Undecidable Problems

- The Halting Problem is undecidable
- Impossible problems are hard . . .

