NP-COMPLETE REDUCTIONS

Admin

Guest lecture on Thursday
Assignment 11 out today (last one!)
Review next Tuesday

P problems

P = problems with a polynomial runtime solution
Also, called “tractable” problems
(Basically, all of the problems in this class)

NP problems

NP is the set of problems that can be verified in polynomial time
A problem can be verified in polynomial time if you can check that a given solution is correct in polynomial time

(NP is an abbreviation for non-deterministic polynomial time)
Big-O allowed us to group algorithms by run-time.

Today, we’re talking about sets of problems grouped by how easy they are to solve.

**P** = problems with a polynomial runtime solution (tractable)

---

**Reduction function**

Given two problems $P_1$ and $P_2$, a reduction function $f(x)$, is a function that transforms a problem instance $x$ of type $P_1$ to a problem instance of type $P_2$ such that a solution to $x$ exists for $P_1$ iff a solution for $f(x)$ exists for $P_2$.

---

**Reduction function**

Allows us to solve $P_1$ problems if we have a solver for $P_2$. 

---
Reduction function

Most of the time we'll worry about yes/no questions (aka, decision problems)

if we have more complicated answers we often just have to do a little work to the solution to the problem of $P_2$ to get the answer

Reduction function: Example

$P_1 =$ Bipartite matching
$P_2 =$ Network flow

Reduction function ($f$): Given any bipartite matching problem turn it into a network flow problem

What is $f$ and what is $f'$?
A problem is NP-complete if:
1. It can be verified in polynomial time (i.e., in NP).
2. Any NP-complete problem can be reduced to the problem in polynomial time (is NP-hard).

Why are NP-complete problems interesting?

Hamiltonian cycle
Given an undirected graph G=(V, E), a Hamiltonian cycle is a cycle that visits every vertex V exactly once.

Longest path
Given a graph G with nonnegative edge weights does a simple path exist from s to t with weight at least g?
3D matching

Bipartite matching: given two sets of things and pair constraints, find a matching between the sets.

3D matching: given three sets of things and triplet constraints, find a matching between the sets.

Figure from Dasgupta et. al 2008

3-SAT

A boolean formula is in n-conjunctive normal form (n-CNF) if:
- it is expressed as an AND of clauses
- where each clause is an OR of no more than n variables

\((a \lor \neg a \lor \neg b) \land (c \lor b \lor d) \land (\neg a \lor c \lor \neg d)\)

3-SAT: Given a 3-CNF boolean formula, is it satisfiable?

SAT

Given a boolean formula of n boolean variables joined by m connectives (AND, OR or NOT) is there a setting of the variables such that the boolean formula evaluate to true?

\((a \land b) \lor (\neg a \land \neg b)\)  

\(((\neg (b \lor c) \land a) \lor (a \land b \land c)) \land c \land \neg b\)

CLIQUE

A clique in an undirected graph \(G = (V, E)\) is a subset \(V' \subseteq V\) of vertices that are fully connected, i.e. every vertex in \(V'\) is connected to every other vertex in \(V'\).

CLIQUE problem: Does \(G\) contain a clique of size \(k\)?

Is there a clique of size 4 in this graph?
Proving NP-completeness

Given a problem NEW to show it is NP-Complete

1. Show that NEW is in NP
   a. Provide a verifier
   b. Show that the verifier runs in polynomial time
2. Show that all NP-complete problems are reducible to NEW in polynomial time
   a. Describe a reduction function f from a known NP-Complete problem to NEW
   b. Show that f runs in polynomial time
   c. Show that a solution exists to the NP-Complete problem IFF a solution exists to the NEW problem generated by f

HALF-CLIQUE

Given a graph G, does the graph contain a clique containing exactly half the vertices?

Is HALF-CLIQUE an NP-complete problem?

Is Half-Clique NP-Complete?

1. Show that NEW is in NP
   a. Provide a verifier
   b. Show that the verifier runs in polynomial time
2. Show that all NP-complete problems are reducible to NEW in polynomial time
   a. Describe a reduction function f from a known NP-Complete problem to NEW
   b. Show that f runs in polynomial time
   c. Show that a solution exists to the NP-Complete problem IFF a solution exists to the NEW problem generated by f

Given a graph G, does the graph contain a clique containing exactly half the vertices?
HALF-CLIQUE

Show that HALF-CLIQUE is in NP

a. Provide a verifier
b. Show that the verifier runs in polynomial time

Verifier: A solution consists of the set of vertices in V'
- check that |V'| = |V|/2
- for all pairs of u, v ∈ V'
  * there exists an edge (u, v) ∈ E

- Check for edge existence in O(V)
- O(V^2) checks
- O(V^3) overall, which is polynomial

HALF-CLIQUE

Show that all NP-complete problems are reducible to HALF-CLIQUE in polynomial time
- Describe a reduction function f from a known NP-Complete problem to HALF-CLIQUE
- Show that f runs in polynomial time
- Show that a solution exists to the NP-Complete problem iff a solution exists to the HALF-CLIQUE problem generated by f

Reduce CLIQUE to HALF-CLIQUE:

Given a problem instance of CLIQUE, turn it into a problem instance of HALF-CLIQUE

Three cases:
1. k = |V|/2
2. k < |V|/2
3. k > |V|/2

Reduce CLIQUE to HALF-CLIQUE:

Given an instance of CLIQUE, turn it into an instance of HALF-CLIQUE

It's already a half-clique problem

f(G, k)
1. if |V|/2 = k
   return G
2. elseif k < |V|/2
   return G plus (|V| - 2k) nodes which are fully connected and are connected to every node in V
3. else
   return G plus 2k - |V| nodes which have no edges
HALF-CLIQUE

Reduce CLIQUE to HALF-CLIQUE:
Given an instance of CLIQUE, turn it into an instance of HALF-CLIQUE

We’re looking for a clique that is smaller than half, so add an artificial clique to the graph and connect it up to all vertices

```python
f(G, k)
1. if [V]/2 = k
2. return G
3. else if k < [V]/2
4. return G plus ([V] - 2k) nodes which are fully connected and are connected to every node in V
5. else
6. return G plus 2k - [V] nodes which have no edges
```

Runtime: From the construction we can see that it is polynomial time

Hal-CLIQUE

Reduce CLIQUE to HALF-CLIQUE:
Given an instance of CLIQUE, turn it into an instance of HALF-CLIQUE

We’re looking for a clique that is bigger than half, so add vertices until k = |V|/2

```python
f(G, k)
1. if [V]/2 = k
2. return G
3. else if k < [V]/2
4. return G plus ([V] - 2k) nodes which are fully connected and are connected to every node in V
5. else
6. return G plus 2k - [V] nodes which have no edges
```

Reduction proof

Show that a solution exists to the NP-Complete problem IFF a solution exists to the NEW problem generated by f

- Assume we have an NP-Complete problem instance that has a solution, show that the NEW problem instance generated by f has a solution
- Assume we have a problem instance of NEW generated by f that has a solution, show that we can derive a solution to the NP-Complete problem instance
Reduction proof

Given a graph $G$ that has a CLIQUE of size $k$, show that $f(G,k)$ has a solution to HALF-CLIQUE

If $k = |V|/2$:
- The graph is unmodified
- $f(G,k)$ has a clique that is half the size

If $k < |V|/2$:
- We added a clique of $|V| - 2k$ fully connected nodes
- There are $|V| + |V| - 2k = 2(|V| - k)$ nodes in $f(G)$
- There is a clique in the original graph of size $k$
- Plus our added clique of $|V| - 2k$
- $k + |V| - 2k = |V| - k$, which is half the size of $f(G)$

Reduction proof

Given a graph $G$ that has a CLIQUE of size $k$, show that $f(G,k)$ has a solution to HALF-CLIQUE

If $k > |V|/2$:
- We added $2k - |V|$ unconnected vertices
- $f(G)$ contains $|V| + 2k - |V| = 2k$ vertices
- Since the original graph had a clique of size $k$ vertices, the new graph will have a half-clique

Key: $f(G)$ was constructed by your reduction function

Use a similar argument to what we used in the other direction
Independent-Set

Given a graph $G = (V, E)$ is there a subset $V' \subseteq V$ of vertices of size $|V'| = k$ that are independent, i.e. for any pair of vertices $u, v \in V'$ there exists no edge between any of these vertices.

Does the graph contain an independent set of size 5?

Independent-Set is NP-Complete

CLIQUE revisited

A clique in an undirected graph $G = (V, E)$ is a subset $V' \subseteq V$ of vertices that are fully connected, i.e. every vertex in $V'$ is connected to every other vertex in $V'$.

CLIQUE problem: Does $G$ contain a clique of size $k$?

Is CLIQUE NP-Complete?

1. Show that CLIQUE is in NP
   a. Provide a verifier
   b. Show that the verifier runs in polynomial time
2. Show that all NP-complete problems are reducible to CLIQUE in polynomial time
   a. Describe a reduction function $f$ from a known NP-Complete problem to CLIQUE
   b. Show that $f$ runs in polynomial time
   c. Show that a solution exists to the NP-Complete problem IFF a solution exists to the CLIQUE problem generated by $f$

Given a graph $G$, does the graph contain a clique of size $k$?
Given a graph $G = (V, E)$ is there a subset $V' \subseteq V$ of vertices of size $|V'| = k$ that are independent, i.e. for any pair of vertices $u, v \in V'$ there exists no edge between any of these vertices. Is there an independent set of size $k$?

Both are selecting vertices

Independent set wants vertices where NONE are connected

Clique wants vertices where ALL are connected

How can we convert a NONE problem to an ALL problem?

**Independent-Set to Clique**

Given a graph $G = (V, E)$, the complement of that graph $G' = (V, E)$ is the a graph constructed by remove all edges $E$ and including all edges not in $E$.

For example, for adjacency matrix this is flipping all of the bits.

$f(G)$

    return $G'$

**Reduction proof**

Show that a solution exists to the NP-Complete problem IFF a solution exists to the NEW problem generated by $f$

- Assume we have an Independent-Set problem instance that has a solution, show that the Clique problem instance generated by $f$ has a solution.

- Assume we have a problem instance of Clique generated by $f$ that has a solution, show that we can derive a solution to Independent-Set problem instance.

$f(G)$

    return $G'$
Proof

Given a graph $G$ that has an independent set of size $k$, show that $f(G)$ has a clique of size $k$.
- By definition, the independent set has no edges between any vertices.
- These will all be edges in $f(G)$ and therefore they will form a clique of size $k$.

Proof

Given $f(G)$ that has clique of size $k$, show that $G$ has an independent set of size $k$.
- By definition, the clique will have an edge between every vertex.
- None of these vertices will therefore be connected in $G$, so we have an independent set.

Independent-Set revisited

Given a graph $G = (V, E)$ is there a subset $V' \subseteq V$ of vertices of size $|V'| = k$ that are independent, i.e. for any pair of vertices $u, v \in V'$ there exists no edge between any of these vertices.

Is Independent-Set NP-Complete?

Independent-Set revisited

Given a graph $G = (V, E)$ is there a subset $V' \subseteq V$ of vertices of size $|V'| = k$ that are independent, i.e. for any pair of vertices $u, v \in V'$ there exists no edge between any of these vertices.

Reduce 3-SAT to Independent-Set
Given a 3-CNF formula, convert it into a graph

\[(a \lor \neg a \lor \neg b) \land (c \lor b \lor d) \land (\neg a \lor \neg c \lor \neg d)\]

For the boolean formula in 3-SAT to be satisfied, at least one of the literals in each clause must be true

In addition, we must make sure that we enforce a literal and its complement must not both be true.

To enforce that only one variable and its complement can be set, we connect each vertex representing \(x\) to each vertex representing its complement \(\neg x\).

**Proof**

Given a 3-SAT problem with \(k\) clauses and a valid truth assignment, show that \(f(3\text{-SAT})\) has an independent set of size \(k\). (Assume you know the solution to the 3-SAT problem and show how to get the solution to the independent set problem.)
Proof

Given a 3-SAT problem with $k$ clauses and a valid truth assignment, show that $f(3\text{-SAT})$ has an independent set of size $k$.

(Assume you know the solution to the 3-SAT problem and show how to get the solution to the independent set problem).

Since each clause is an OR of variables, at least one of the three must be true for the entire formula to be true. Therefore each 3-clique in the graph will have at least one node that can be selected.

Our known NP-Complete problems

We can reduce any of these problems to a new problem in an NP-completeness proof

- SAT, 3-SAT
- CLIQUE, HALF-CLIQUE
- INDEPENDENT-SET
- HAMILTONIAN-CYCLE
- TRAVELING-SALESMAN
- VERTEX-COVER
- SUBSET-SUM
Search vs. Exists

All the problems we’ve looked at asked decision questions:
- Is there a hamiltonian cycle?
- Does the graph have a clique of size k?
- Does the graph have an independent set of size k?
- ...

For many of the problems with a k in them, we really want to know what the largest/smallest one is:
- What is the largest clique in the graph?
- What is the shortest path that visits all the vertices exactly once?

Why don’t we care?

P vs. NP

The big question:

Someone finds a polynomial time solution to one of the NP-Complete problems
NP-Complete problems are somehow harder and distinct

Solving NP-Complete problems

http://www.tsp.gatech.edu/index.html