CS5371
Theory of Computation

Lecture 22: Complexity VII
(More NP-complete Problems)

Announcement

Final Exam:

- Date: Jan 12, 2007 (Friday)
- Time: 3:20pm - 6:10pm (F7F8F9)
- Scope: Chapter 1 to Chapter 7
- Tentative Format:
- 4-5 Long Questions: 85%
- Some True/False Questions: 15%

Objectives

- We shall continue to look at more NP-complete problems:

  Directed-HAMPATH, HAMPATH, SUBSET-SUM, PARTITION

- Some more if we have time today

Directed HAMPATH

Let G be a directed graph. A directed Hamiltonian path in G is a path that visits all the vertices of G once and only once.

Let D-HAMPATH be the language

$\{ \langle G,s,t \rangle \mid G$ has a directed Hamiltonian path which starts from s and ends at $t\}$

Theorem: D-HAMPATH is NP-complete.

D-HAMPATH is NP-complete (2)

Proof: First, D-HAMPATH is in NP (easy to show). Then, we show it is NP-complete by reduction from 3SAT.

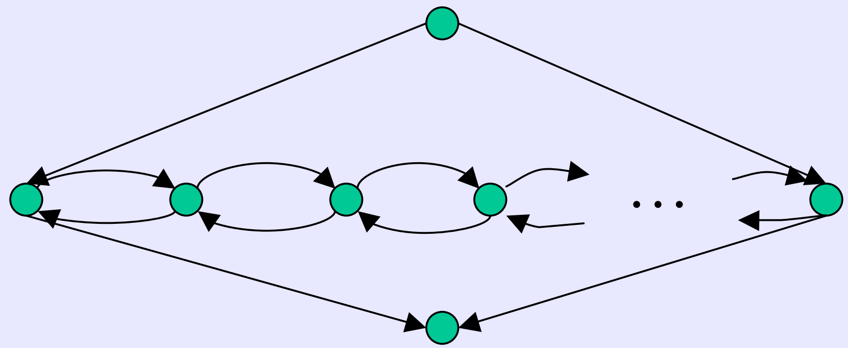
To determine if $\langle F \rangle$ is in 3SAT, we shall construct G with two special vertices s and t such that

$$\langle F \rangle \in \text{3SAT} \Leftrightarrow \langle G,s,t \rangle \in \text{D-HAMPATH}$$

D-HAMPATH is NP-complete (3)

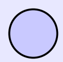
Let x_1, x_2, ..., x_j be the variables in the 3cnf-formula F. For each variable x_i, we create a 'diamond-shaped structure' that contains a horizontal row of nodes:

x_i



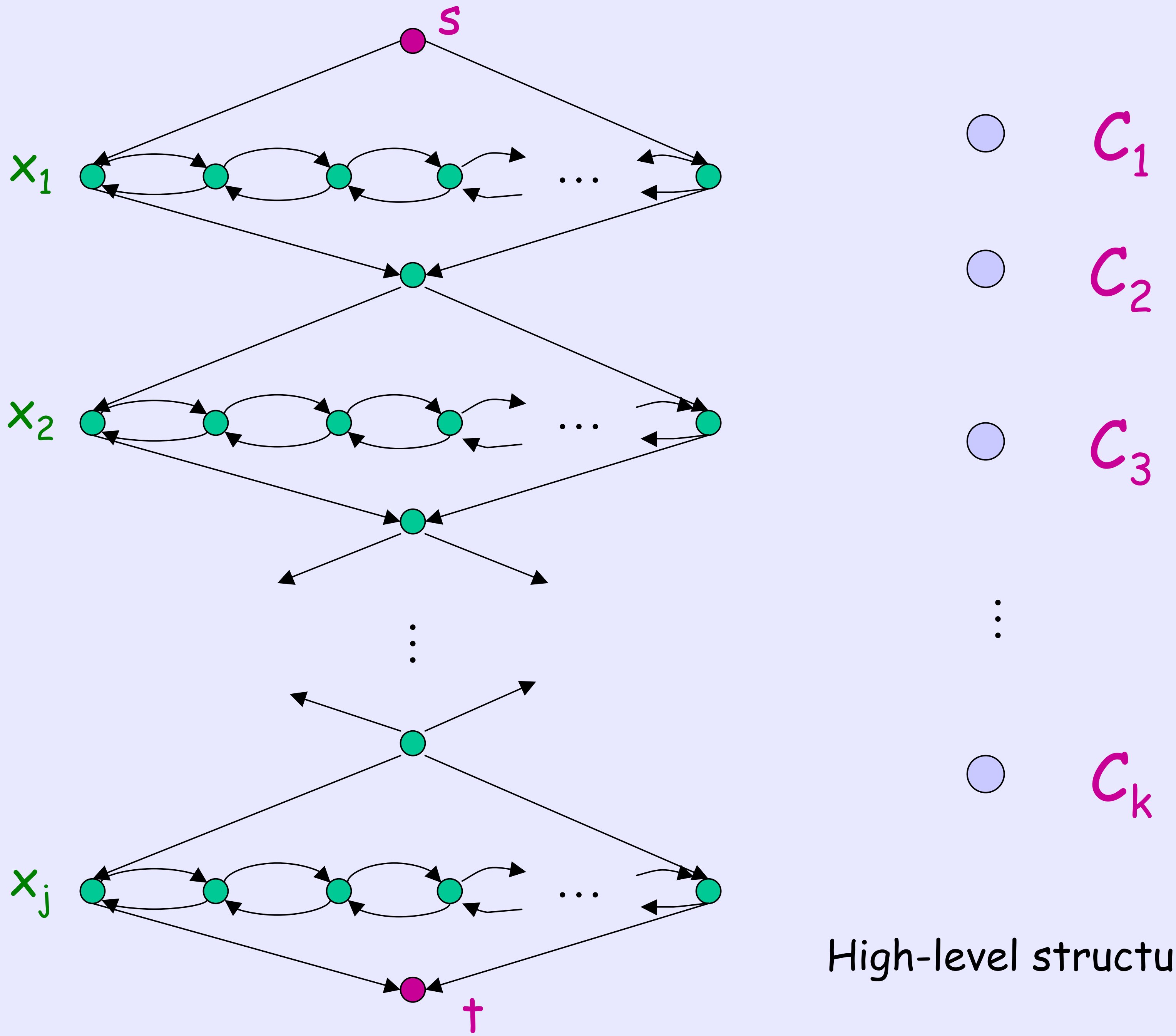
D-HAMPATH is NP-complete (4)

Let C_1, C_2, …, C_k be the clauses in F. For each clause C_m, we create a node:



C_m

The figure in the next slide shows the global structure of G. It shows all the elements of G and their relationship, except the relationship of the variables to the clauses that contains them



High-level structure of G

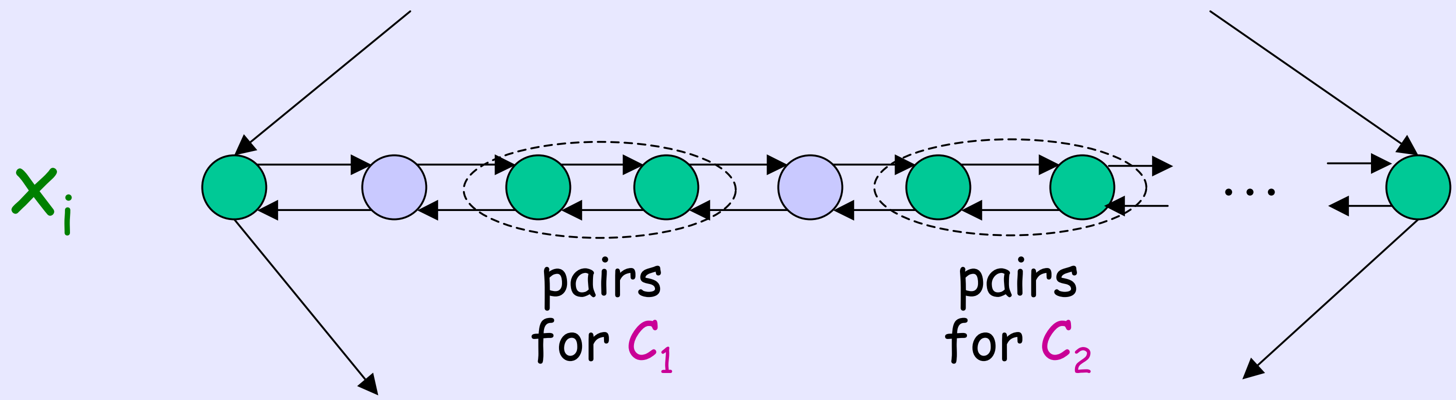
Diamond Structure

Each diamond structure contains a horizontal row of nodes connected by edges running in both directions.

There are 3k+1 nodes in each row (in addition to the two nodes on the end that belongs to the diamond)

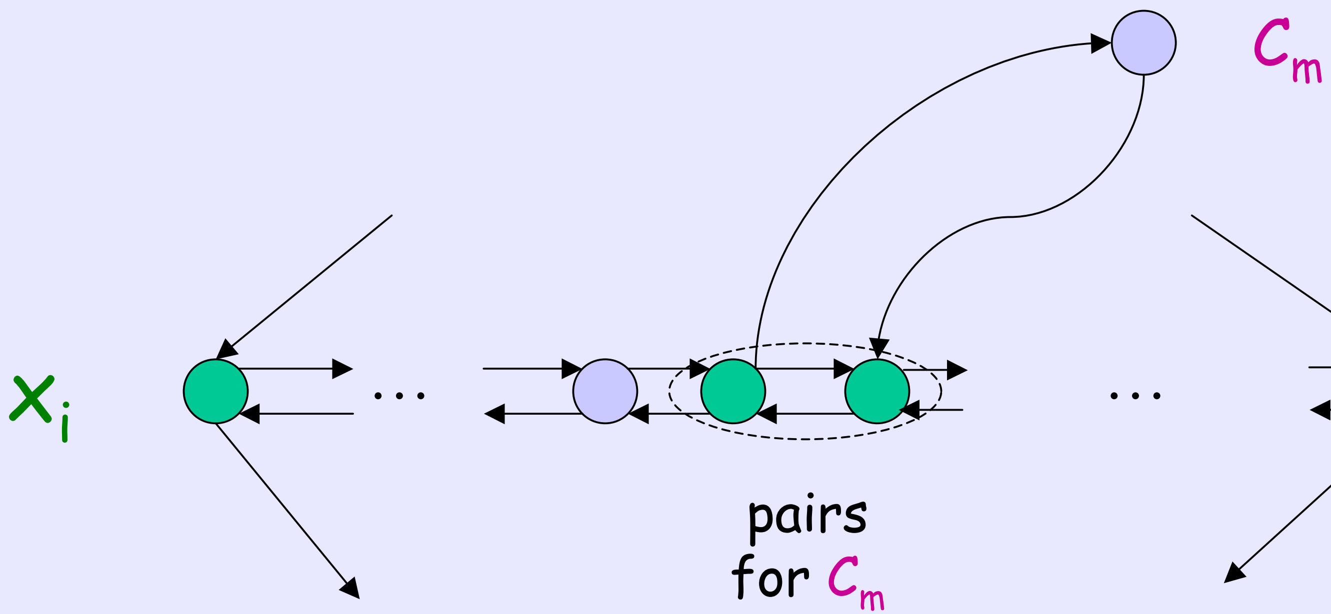
Diamond Structure

The nodes in the horizontal row are grouped into adjacent pairs, one for each clause, with extra separator nodes next to the pair:



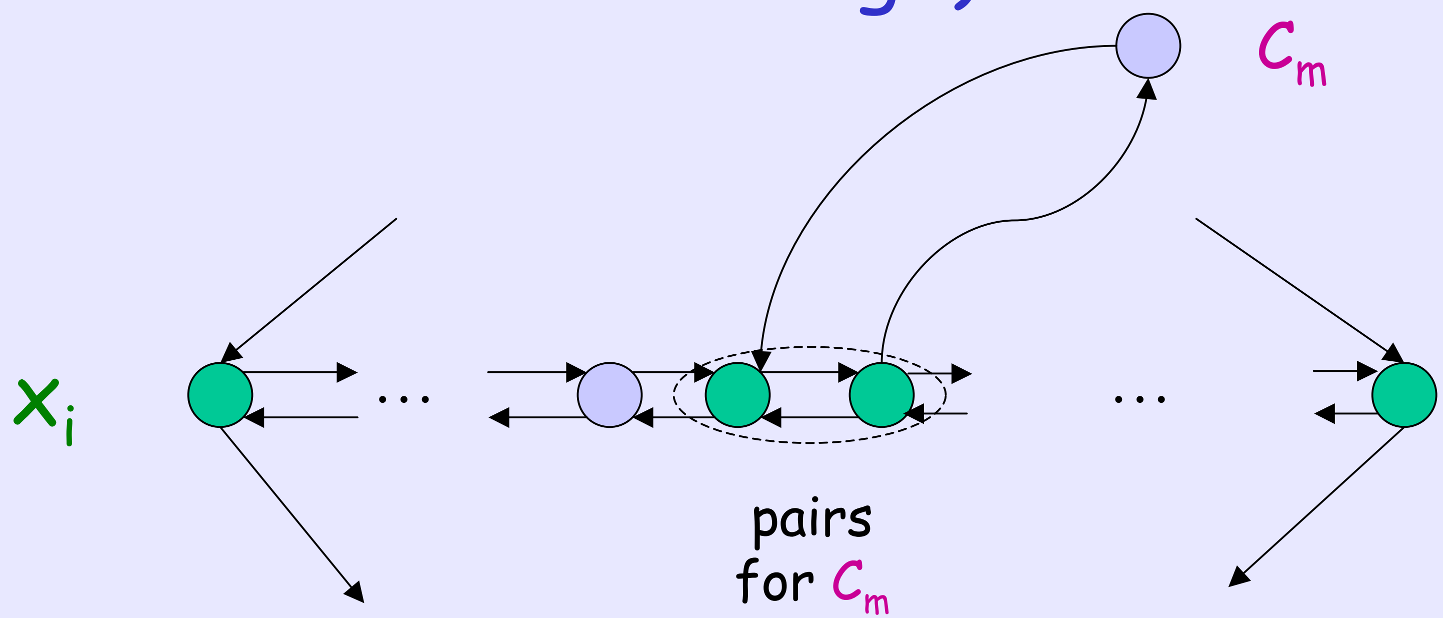
Connection with Clauses

If variable x_i appears in clause C_m, we add the following two edges from the m^{th} pair in the x_i's diamond structure to node C_m



Connection with Clauses (2)

If $\neg x_i$ appears in clause C_m, we add the following two edges from the m^{th} pair in the x_i's diamond structure to node C_m (Note the direction change)



D-HAMPATH is NP-complete (5)

After we add all the edges corresponding to each occurrence of x_i or $\neg x_i$ in each clause, the construction of G is finished

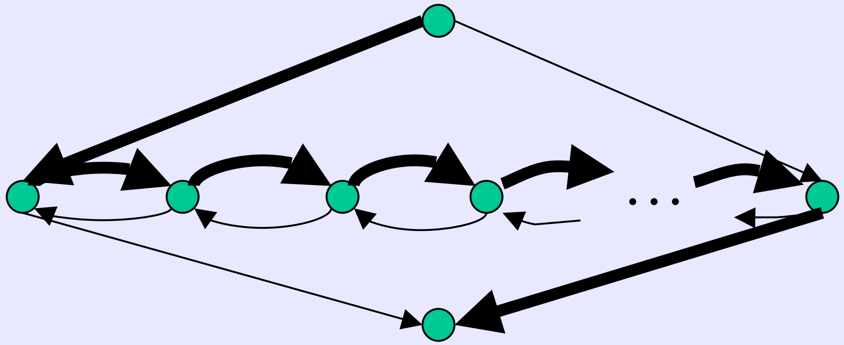
Now, we claim that this is our desired reduction.

(➔) Suppose that F is satisfiable. To demonstrate an Hamiltonian path from s to t in G, we first ignore the clause nodes.

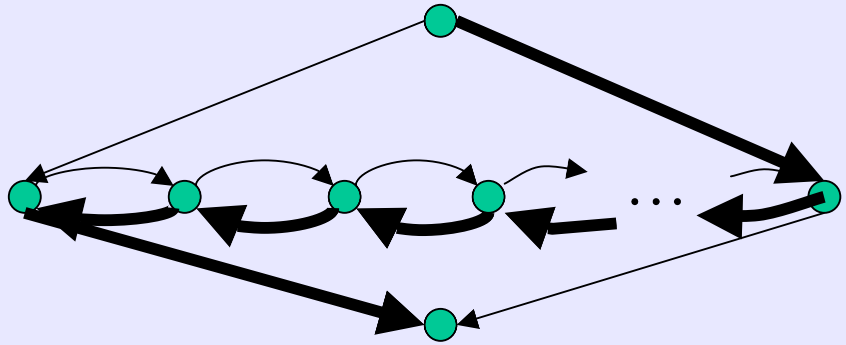
D-HAMPATH is NP-complete (6)

The path begins at s, goes through each diamond in turn, and ends up at t.

To hit the horizontal nodes in a diamond, the path is either one of the following way:



Left-to-Right



Right-to-Left

D-HAMPATH is NP-complete (7)

If x_i is assigned TRUE in the satisfying assignment of F, we use Left-to-Right method to traverse the corresponding diamond. Otherwise, if x_i is assigned FALSE, we use the Right-to-Left method

So far, the path covers all the nodes in G except the clause nodes. We can easily include them by adding detours at the horizontal node.

D-HAMPATH is NP-complete (8)

In each clause, we select a literal that is assigned TRUE in the satisfying assignment of F.

Suppose we select x_i in clause C_m. Then, in our current path, the horizontal nodes in x_i are from Left-to-Right. Also, by our construction of edges in P. 11, we can see that we can easily detour at the m^{th} pair of horizontal nodes, and cover C_m.

D-HAMPATH is NP-complete (9)

Similarly, if we select $\neg x_i$ in clause C_m, then in our current path, the horizontal nodes in x_i are from Right-to-Left. Also, by our construction of edges in P. 12, we can see that we can easily detour at the m^{th} pair of horizontal nodes, and cover C_m.

Thus, if F is satisfiable, G has a Hamiltonian path from s to t.

D-HAMPATH is NP-complete (10)

(←) On the other direction, if G has a Hamiltonian path from s to t, we shall demonstrate a satisfying assignment for F.

Firstly, we take a look at the Hamiltonian path. If it is "normal"--- that is, visiting the diamonds in the order from top one to the bottom one (excluding the detours to clause nodes) --- we can obtain a satisfying assignment as follows: (next slide)

D-HAMPATH is NP-complete (11)

If the path is Left-to-Right in the diamond for x_i, we assign TRUE to x_i

If the path is Right-to-Left in the diamond for x_i, we assign FALSE to x_i

Now, because each clause node visited once in the Hamiltonian path, by determining how the detour is taken, we know that at least one literal in each clause is TRUE.

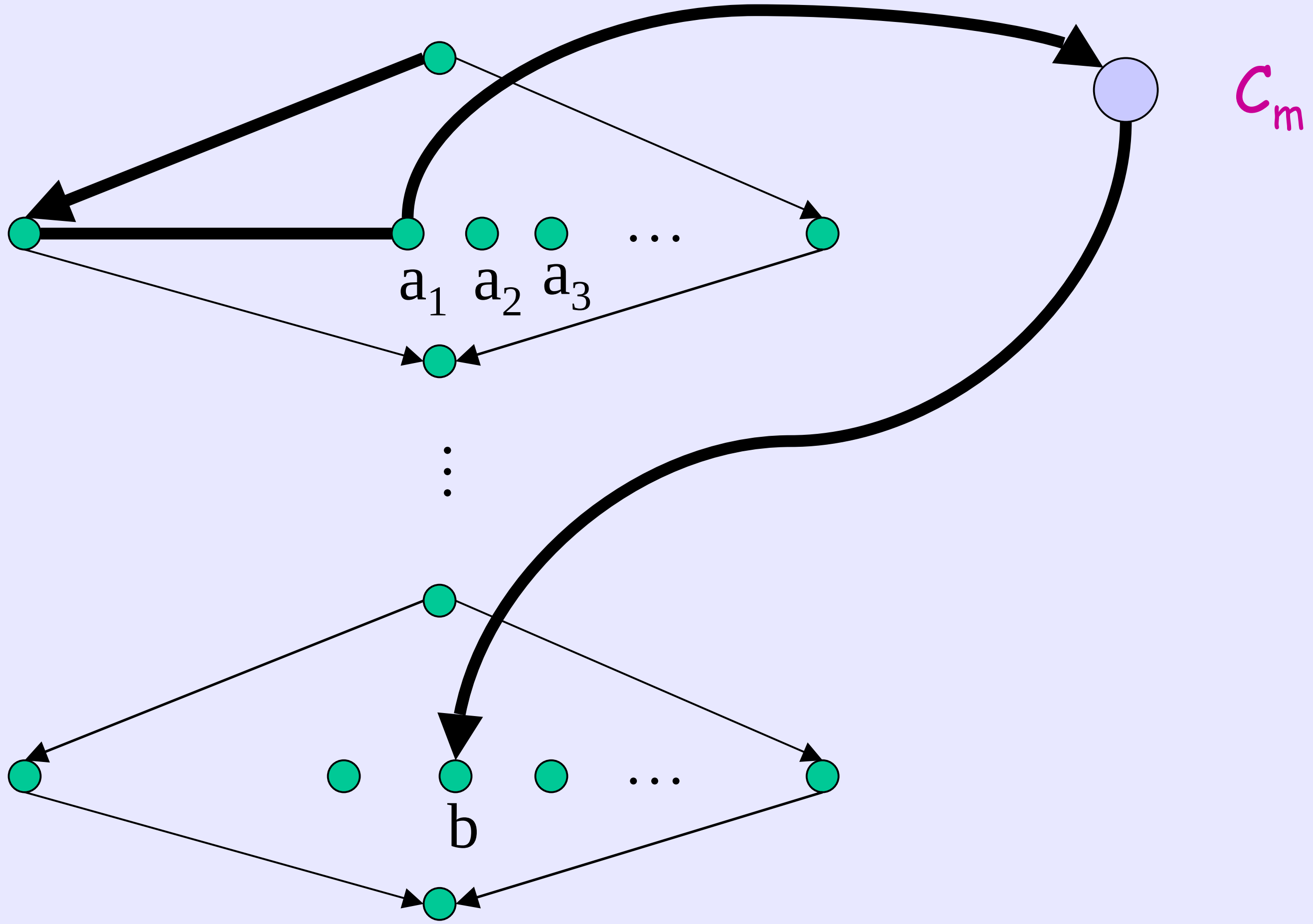
D-HAMPATH is NP-complete (12)

All remains to show is:

Hamiltonian path must be normal

Suppose on the contrary that it is not normal. Then, the Hamiltonian path must have entered a clause from one diamond, but left the clause to another diamond, as shown in next slide:

When Hamiltonian Path not Normal



D-HAMPATH is NP-complete (13)

The Hamiltonian path goes from a_1 to c but instead of returning to a_2, it goes to b in a different diamond.

If that occurs, either a_2 or a_3 must be a separator node

Case 1 [a_2 is a separator]: the only edges entering would be from a_1 or a_3.

Case 2 [a_3 is a separator]: a_1 and a_2 are in the same clause pair

D-HAMPATH is NP-complete (14)

In both cases, the path cannot contain a_2, because a_2 connects to at most three nodes: a_1 , a_3, and c (why?), but since a_1 and c has both been visited, a_2 cannot find two distinct nodes---one incoming neighbor, one outgoing neighbor---that connects it to the Hamiltonian path

Thus, all Hamiltonian path in G from s to t must be normal, and this implies that if such a path exisits, F is satisfiable

D-HAMPATH is NP-complete (15)

In conclusion, we have

$$\langle F \rangle \in \text{3SAT} \Leftrightarrow \langle G,s,t \rangle \in \text{D-HAMPATH}$$

As it is easy to see that the above reduction from 3SAT to D-HAMPATH takes only polynomial time, therefore D-HAMPATH is NP-complete

Undirected HAMPATH

Recall that HAMPATH be the language

$\{ \langle G,s,t \rangle \mid G$ has a (undirected) Hamiltonian path which starts from s and ends at t $\}$

Theorem: HAMPATH is NP-complete.

HAMPATH is NP-complete

We just give a sketch of the proof: First, it is easy to see that HAMPATH is in NP. To see why it is NP-complete, we reduce 3SAT to HAMPATH, using similar construction as we use in D-HAMPATH.

However, we are now dealing with undirected graphs. Instead of using directed edges in the reduction in D-HAMPATH before, we replace every node u in the previous graph …

HAMPATH is NP-complete (2)

… by 3 nodes u_{in}, u_{mid}, u_{out}, in the new graph.

A directed edge from u to v in the previous graph is now replaced by an undirected edge joining u_{out} and v_{in}

This completes the reduction, and similarly, we can show that this reduction works (try this as an exercise at home!)

Again, the reduction takes polynomial time
Thus, HAMPATH is NP-complete

How about HAM-CIRCUIT ?

Let HAM-CIRCUIT be the language

$\{ \langle G \rangle \mid G$ has a Hamiltonian circuit $\}$

Theorem: HAM-CIRCUIT is NP-complete.

HAMCIRCUIT is NP-complete (2)

Proof: First, HAMCIRCUIT is in NP (easy to show). Then, we show it is NP-complete by reduction from HAMPATH.

To determine if $\langle G,s,t \rangle$ is in HAMPATH, we construct G' by adding to G a new vertex v, and two edges $\{v,s\}$ and $\{v,t\}$. Then it is easy to see that (why??):

$$\langle G,s,t \rangle \in \text{HAMPATH} \Leftrightarrow \langle G' \rangle \in \text{HAMCIRCUIT}$$

SUBSET-SUM is NP-Complete

Let S be a set of positive integers.

Let SUBSET-SUM be the language

$\{ \langle S,k \rangle \mid S$ has a subset whose sum is $k \}$

Theorem: SUBSET-SUM is NP-complete.

SUBSET-SUM is NP-complete (2)

Proof: First, SUBSET-SUM is in NP (easy to show). Then, we show it is NP-complete by reduction from 3SAT.

Let F be a Boolean formula in 3cnf-form. Let $x_1,x_2,\ldots,x_j$ be its variables and let $C_1,C_2,\ldots,C_k$ be its clauses. We transform F into a set S of $2j+2k$ (very large) numbers, with each number having $j+k$ digits as follows: (next slide)

SUBSET-SUM is NP-complete (3)

For each variable x_i, we create two numbers y_i and z_i, such that their ith leftmost digit is set to one. Also, if x_i appears in clause C_m, the $(k-m)$th rightmost digit of y_i is set to one. If $\neg x_i$ appears in clause C_m, the $(k-m)$th rightmost digit of z_i is set to one. The remaining digits are all set to zero. E.g.,

Constructing the numbers in S

	1	2	3	$\cdots$	j	C_1	C_2	$\cdots$	C_k
y_1	1	0	0	$\cdots$	0	1	0		
z_1	1	0	0	$\cdots$	0	0	0		
y_2	0	1	0	$\cdots$	0	0	1		
z_2	0	1	0	$\cdots$	0	1	0		
$\vdots$									
y_j	0	0	0	$\cdots$	1	0	1		
z_j	0	0	0	$\cdots$	1	0	0		

Assume $C_1 = (x_1 \vee \neg x_2 \vee x_3)$ and $C_2 = (x_2 \vee \neg x_3 \vee x_j)$

SUBSET-SUM is NP-complete (4)

In addition, S contains one pair of number, g_m and h_m for each clause C_m, such that these two numbers are equal, with only the (k-m)th rightmost digit set to one, and all other digits set to zero.

Let t = 111…1 333…3 [j 1s followed by k 3s] be the target number. We shall show that F is satisfiable if and only if a subset of numbers in S adds up to t.

SUBSET-SUM is NP-complete (5)

(➔) Suppose F is satisfiable. We select y_i if x_i is assigned TRUE, and select z_i otherwise. assigned FALSE. If we add up the numbers we have selected so far,

1. The leftmost j digits will match those of t (why?)

2. Each of the rightmost digit will be between 1 and 3 (why?)

SUBSET-SUM is NP-complete (6)

Now, we further select g_i and h_i so that the sum of the rightmost digit adds up to 3, thus hitting the target.

(←) On the other hand, suppose a subset of S adds up to t. It implies exactly one of the y_i or z_i is in this subset (why?). By setting x_i to TRUE when y_i is in the subset, and FALSE if z_i is in the subset, F will be satisfied. The reason is that…

SUBSET-SUM is NP-complete (7)

… since each column for C_m sums up to 3, at least 1 is contributed by some y_i or z_i in the subset (why?). If it is by y_i in the subset, it means (i) the clause C_m contains x_i, and (ii) we have assigned x_i to TRUE, so that C_m is satisfied. Similarly, if it is by z_i, it means (i) the clause C_m contains $\neg x_i$, and (ii) we have assigned x_i to FALSE, so that C_m is satisfied. Thus, F is satisfied.

SUBSET-SUM is NP-complete (8)

Now, we have shown that

$$\langle F \rangle \in \text{3SAT} \Leftrightarrow \langle S, t \rangle \in \text{SUBSET-SUM}$$

Also, it is easy to check that the above reduction takes polynomial time (in terms of the length of F).

Thus, SUBSET-SUM is NP-complete.

PARTITION is NP-Complete

Let S be a set of positive integers.

Let PARTITION be the language

{ ⟨S⟩ | S can be partitioned into two groups such that the sum in each group is the same }

Theorem: PARTITION is NP-complete.

PARTITION is NP-complete (2)

Proof: First, PARTITION is in NP (easy to show). Then, we show it is NP-complete by reduction from SUBSET-SUM.

To determine if S,k is in SUBSET-SUM, let X = sum of values in S. We construct S' by adding the two numbers $2X-k$ and $X+k$ to S. Then it is easy to see that (why??):

$$\langle S,k\rangle \in \text{SUBSET-SUM} \Leftrightarrow \langle S'\rangle \in \text{PARTITION}$$

Brain Teaser 1: HITTING SET

Let C be a collection of subsets of S. A set of S' is called a hitting set for C if every subset of C has at least one element in S'.

Let HITTING-SET be the language

$\{ \langle C,k \rangle \mid C$ is a collection of subsets with a hitting set of size $k \}$

Theorem: HITTING-SET is NP-complete.

Brain Teaser 2: SUBGRAPH ISOMORPHISM

We say two graph $H=(V,E)$ and $H'=(V',E')$ are isomorphic if there exists a one-to-one function f: $V' \rightarrow V$ such that

$$\{ u,v \} \text{ in } E \text{ if and only if } \{ f(u),f(v) \} \text{ in } E'$$

Let SUBGRAPH-ISO be the language

$\{ \langle G,H \rangle \mid G$ has a subgraph isomorphic to $H\}$

Theorem: SUBGRAPH-ISO is NP-complete.

SUBGRAPH-ISO is NP-complete

Proof: First, SUBGRAPH-ISO is in NP (easy to show). Then, we show it is NP-complete by reduction from CLIQUE.

Given G,k, we construct G',H' as follows:

Set G' = G. Set H' = k-clique. Then it is easy to see that:

$\langle G,k \rangle \in$ CLIQUE $\Leftrightarrow$ $\langle G',H' \rangle \in$ SUBGRAPH-ISO

Brain Teaser 3: BOUNDED-DEG SPANTREE

A spanning tree of a graph $G=(V,E)$ is a tree containing every vertex in G, and whose edges are from E. A degree-k spanning tree is a spanning tree such that degree of each internal node is at most k.

Let Bounded-Deg-ST be the language

$\{ \langle G,k \rangle \mid G$ has a degree-k spanning tree $\}$

Theorem: Bounded-Deg-ST is NP-complete.

Bounded-Deg-ST is NP-complete

Proof: First, Bounded-Deg-ST is in NP (easy to show). Then, we show it is NP-complete by reduction from HAMPATH.

Hint: What is a degree-2 spanning tree?

Bounded-Deg-ST is NP-complete (2)

A degree-2 spanning tree is a Hamiltonian path in the graph!!!

Now, given a graph G, we can transform G into G' by adding two nodes, u and v, and two edges, $\{u,s\}$ and $\{v,t\}$. Then, we can see

$\langle G,s,t\rangle \in$ HAMPATH $\Leftrightarrow \langle G',2\rangle \in$ Bounded-Deg-ST

Thus, Bounded-Deg-ST is NP-complete (why?)

Brain Teaser 4: KNAPSACK

Let S be a set of items, each item x in S has a positive integral value v(x) and a positive integral weight w(x).

Let KNAPSACK be the language

{ $\langle$S,b,k$\rangle$ | a subset of items in S of total weight at most b, but whose total value is at least k }

Theorem: KNAPSACK is NP-complete.

KNAPSACK is NP-complete

Proof: First, KNAPSACK is in NP (easy to show). Then, we show it is NP-complete by reduction from PARTITION.

Let $S = \{s_1, s_2, \ldots, s_j\}$ be a set of +ve integers. We want to construct S', b, and k such that

$$\langle S \rangle \in \text{PARTITION} \Leftrightarrow \langle S', b, k \rangle \in \text{KNAPSACK}$$

The construction of S' is as follows: (next slide)

KNAPSACK is NP-complete (2)

For each s_i in S, we create x_i in S' such that $w(x_i) = v(x_i) = s_i$. Also, we set $b = k = Y/2$, where $Y = s_1+s_2+\ldots+s_j$

Now, if S has a partition, then a subset of numbers in S adds up to $Y/2$. By choosing the items of S' that corresponds to this subset, the total weight $\leq b$ and the total value $\geq k$ (why?) Thus, $\langle S \rangle \in$ PARTITION

➔ $\langle S',b,k \rangle \in$ KNAPSACK

KNAPSACK is NP-complete (3)

On the other hand, if a subset of items of S' have total weight $\leq b$ and total value $\geq k$, then the sum of the corresponding s_i in S will be at most b and at least k (why?). Since $b = k = Y/2$, we have the sum of those items in $S = Y/2$. Thus, $\langle S', b, k \rangle \in$ KNAPSACK ➔ $\langle S \rangle \in$ PARTITION

As the reduction can be done in polynomial time, KNAPSACK is NP-complete.

What we have learnt

- The class P, and the class NP
- Some problems in NP are the most difficult ones in the set. We call them NP-complete problems
- SAT is NP-complete
- Other problems in NP can be shown to be NP-complete using polynomial time reduction (from what to what?)

Next Time

- Chapter 8 (not in exam) or Revision?