Outline

1. One More Dynamic Programming
2. P and NP
3. NP-Complete

RNA Secondary Structure

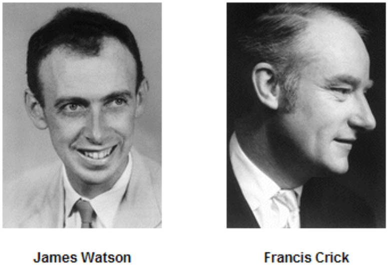
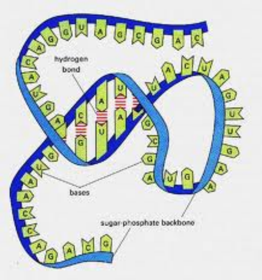
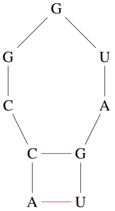


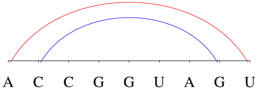
Figure: Watson and Crick

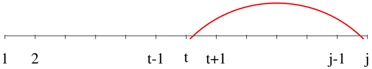
RNA Secondary Structure

RNA:









RNA Secondary Structure

Definition (RNA Secondary Structure)

Alphabet: $B = \{ A, C, G, U \}$

RNA String: $R = b_1 b_2 \dots b_n; b_i \in B$

Secondary structure of R: $S = \{ (i, j) \}$

1. $\forall s \in S, s \in (A, U), (U, A), (C, G), (G, C)$。
2. (No sharp turns)$\forall (i, j) \in S, i < j - 4$。
3. S is a matching
4. (Noncrossing) $\forall (i, j), (k, l) \in S, \neg (i < k < j < l)$。

$Goal : \max(| S |)$.

RNA Secondary Structure

$P(i,j)$: the maximum number of pairs on $b_i \dots b_j$.

$$P(i,j) = 0, i \le j - 4; P(1,n)$$

- j is not involved in a pair, *or*
- j pairs with t for some $t < j - 4$.

$$P(i,j) = \max\left(P(i,j-1), \max_{t:(1),(2)} (1 + P(i,t-1) + P(t+1,j-1))\right).$$

RNA Secondary Structure

begin

  Initialize $P(i, j) = 0, i \geq j - 4$;

  foreach $l = 5, 6, \ldots, n-1$ **do**

    foreach $i = 1, 2, \ldots, n-l$ **do**

      Set $j = i + l$;

      Compute $P(i, j) =$

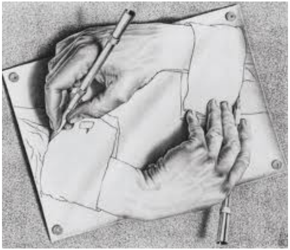
      $\max\left(P(i, j-1), \max_{(1),(2)}(1 + P(i, t-1) + P(t+1, j-1))\right)$;

  return $P(1, n)$;

Outline

Computability vs. Complexity Theory

Computability first:

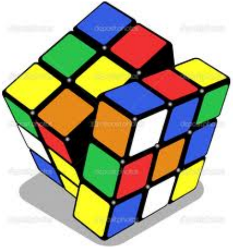


Halting problem is undecidable.

Computability vs. Complexity Theory

Complexity to follow:

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9



Easy? It is NP-Complete.

Hard? God's Number is 20.

P vs. NP

Definition (The Class P)

Problems decidable in polynomial time.

- P to the input size
- closed property
- $\approx$ realistically efficient

P vs. NP

Definition (The Class P)

Problems decidable in polynomial time.

- P to the input size
- closed property
- $\approx$ realistically efficient

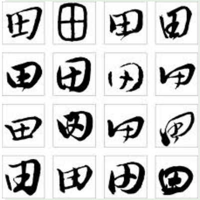
COROLLARY 1. *The number of steps in our algorithm is at most* $1752484608000\, n^{79} L^{25} / D^{26} (\Theta_0)$.

COROLLARY 2. *The number of steps in our algorithm is at most* $117607251220365312000\, n^{79} (\ell_{\max} / d_{\min}(\Theta_0))^{26}$.

P vs. NP

Example (The Class P)

Euler path:



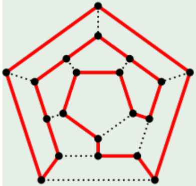
Reachability:

GCD:

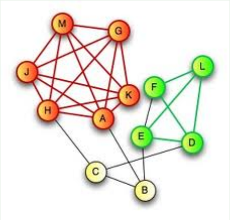
P vs. NP

Example (not *known* to be in P)

Hamiltonian path:



Clique:

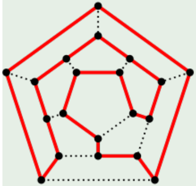


Subset sum:

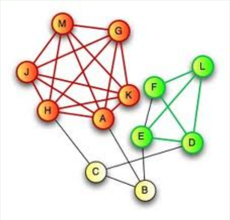
P vs. NP

Example (not *known* to be in P)

Hamiltonian path:



Clique:



Subset sum:

avoiding brute force without success

P vs. NP

Definition (The Class NP)

Problems decidable in polynomial time by nondeterministic algorithm.

P vs. NP

Definition (The Class NP)

Problems decidable in polynomial time by nondeterministic algorithm.

$$NP \neq \text{Non-Polynomial}$$

$$NP \neq \text{No Problem}$$

P vs. NP

Example (Hamiltonian path $\in$ NP)

Input: $< G, s, t >$

Problem: Is there Hamiltonian path between s and t ?

Proof.

1. select n vertices, $v_1, \ldots, v_n$ nondeterministically
2. repetition ? *reject* : 2
3. $v_1 = s, v_n = t$? *reject* : 4
4. $(v_i, v_{i+1}) \in E(G)$? *accept* : *reject*.
5. polynomial

$\square$

P vs. NP

Example (Clique $\in$ NP)

Input: $< G, k >$

Problem: Does G contain a k-clique?

Proof.

1. select a subset $K \subseteq V(G)$ with size k nondeterministically
2. $\forall (K_i, K_j) \in E(G)$? $accept$: $reject$.
3. polynomial

$\square$

P vs. NP

Example (Subset sum $\in$ NP)

Input: $< S, t >$

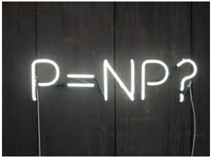
Problem: Is there a subset summing to t ?

Proof.

1. select a subset $C \subseteq S$ nondeterministically
2. $\sum C = t \ ? \ accept : \ reject.$
3. polynomial

$\square$

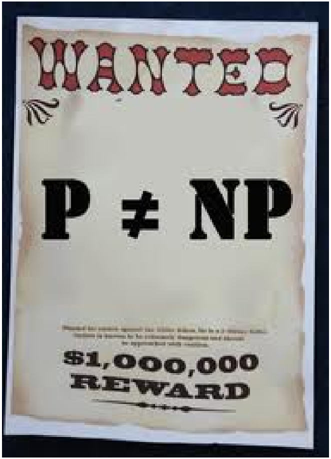
P vs. NP



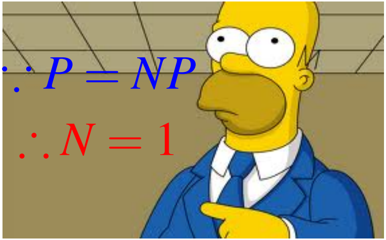
- P = *decided* quickly
- solving problem
- composing

- NP = *verified* quickly
- checking the solution
- appreciating symphony

P vs. NP



P vs. NP



P vs. VP

"Milestones" in

http://www.win.tue.nl/~gwoegi/P-versus-NP.htm

It will be solved by either 2048 or 4096. (Knuth)

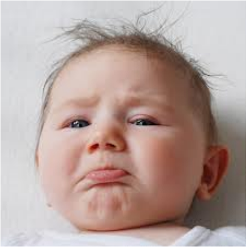
Outline

1. One More Dynamic Programming
2. P and NP
3. NP-Complete

NP-Complete

How hard are they ?

1. Hamiltonian path
2. Clique
3. Subset sum



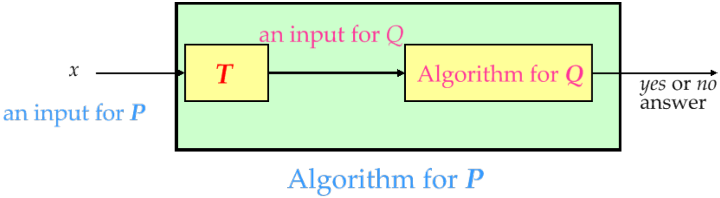
NP-Complete

NP-Complete problem ([@1970s]):

1. subset of P
2. any problem in NP *reducible to* any one in NPC
3. hard core of NP
4. one is polynomial $\Rightarrow$ all are polynomial

NP-Complete

Key concept: polynomial time reduction

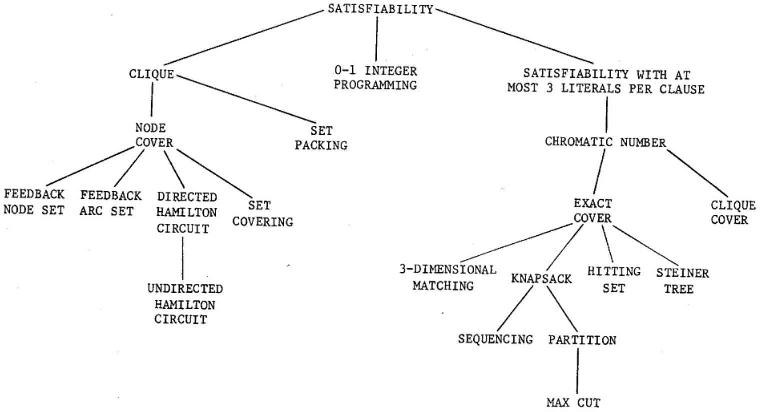


$P \leq_P Q$

- *difficulty* flows forward
- *efficient* flows backward
- *composition*

NP-Complete

Reducibility among known problems ([Karp@70s])



NP-Complete: Known NPC Problems

1. Packing Problems:

Example (Independent Set)

Given a graph G and a number k,

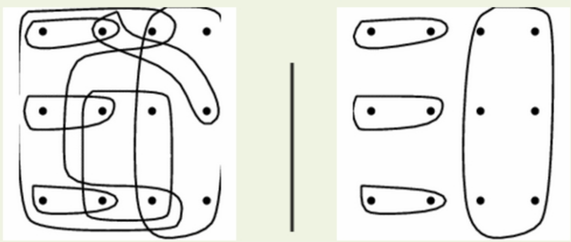
does G contain an independent set of size k?

NP-Complete: Known NPC Problems

1. Packing Problems:

Example (Set Packing)

Given a universe $\mathcal{U}$, a family $\mathcal{F}$ of subsets of $\mathcal{U}$ and a number k, is there a subfamily $\mathcal{C} \subseteq \mathcal{F}$ of size k such that all sets in $\mathcal{C}$ are pairwise disjoint?



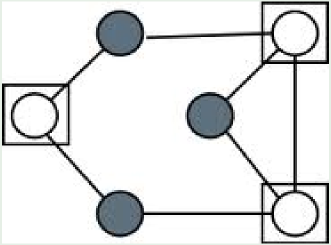
NP-Complete: Known NPC Problems

2. Covering Problems:

Example (Vetex Cover)

Given a graph G and a number k,

does G contain a vertex cover (*a subset D of vertices where every edge of G touches one of those nodes*) of size k?



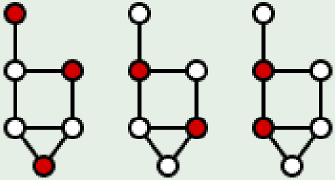
NP-Complete: Known NPC Problems

2. Covering Problems:

Example (Dominating Set)

Given a graph G and a number k,

does G contain a dominating set (*a subset D of vertices where every other vertex is joined to at least one member of D*) of size k?



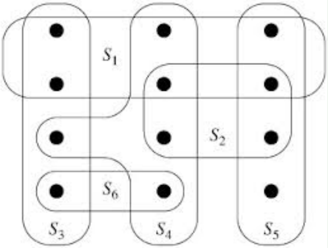
(a) (b) (c)

NP-Complete: Known NPC Problems

2. Covering Problems:

Example (Set Cover)

Given a universe $\mathcal{U}$, a family $\mathcal{F}$ of subsets of $\mathcal{U}$ and a number k, is there a subfamily $\mathcal{C} \subseteq \mathcal{F}$ of size k whose union is $\mathcal{U}$?



$$\mathcal{U} = \{1, 2, 3, 4, 5\}$$

$$\mathcal{F} = \{\{1, 2, 3\}, \{2, 4\}, \{3, 4\}, \{4, 5\}\}$$

$$\mathcal{C} = \{\{1, 2, 3\}, \{4, 5\}\}$$

NP-Complete: Known NPC Problems

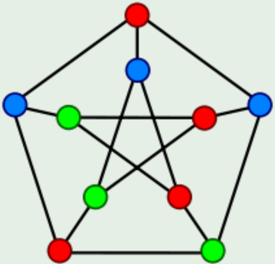
3. Partition Problems:

Example (Graph Coloring)

Given a graph G and a number k,

does G have a k-coloring

$(c : V \to \{1, \ldots, k\}; \forall(u, v) \in E, c(u) \neq c(v))$?



NP-Complete: Known NPC Problems

3. Partition Problems:

Example (3D Matching)

Given disjoint sets X, Y, Z, each of size n, and a set

$T \subseteq X \times Y \times Z$ of ordered triples,

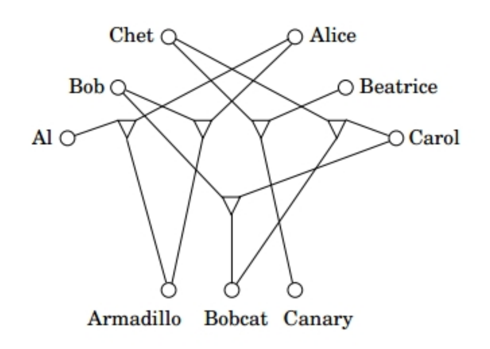
does there exist a set of n triples in T such that each element of

$X \cup Y \cup Z$ is contained in exactly one of these triples?

NP-Complete: Known NPC Problems

3. Partition Problems:

Example (3D Matching (cont.))



$(Al, Alice, Armadillo); (Bob, Carol, Bobcat); (Chet, Beatrice, Canary).$

NP-Complete: Known NPC Problems

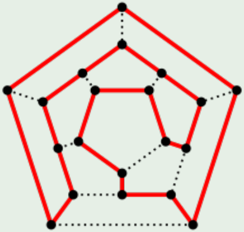
4. Sequencing Problems:

Example (Hamiltonian Path)

Given a (un)directed graph G and vertices s, t,

is there a Hamiltonian path (*visiting every vertex exactly once*)

between s and t?



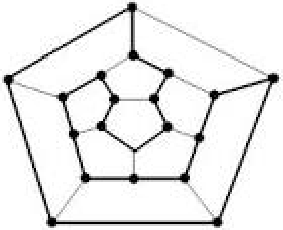
NP-Complete: Known NPC Problems

4. Sequencing Problems:

Example (Hamiltonian Cycle)

Given a (un)directed graph G,

is there a Hamiltonian cycle (*visiting every vertex exactly once*)?



A Hamiltonian Cycle
on the Dodecahedron

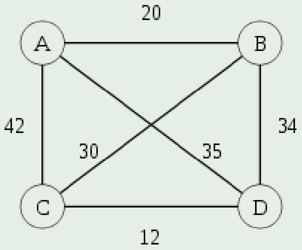
NP-Complete: Known NPC Problems

4. Sequencing Problems:

Example (Traveling Salesman Problem (TSP))

Given a set of distances on n cities and a number k,

is there a cycle visiting every vertex exactly once of total distance k or less?



NP-Complete: Known NPC Problems

5. Numerical Problems

Example (Subset Sum)

Given natural numbers $w_1, \dots, w_n$, and a targe number W, is there a subset of $\{ w_1, \dots, w_n \}$ that sums to precisely W?

$$S = \{ -7, -3, -2, 5, 8 \}, W = 0$$

$$\{ -3, -2, 5 \}.$$

NP-Complete: Known NPC Problems

6. Constraint Satisfaction Problems

Example (3SAT)

Given a set of clauses $C_1, \ldots, C_k$, each of length 3, over a set of variables $X = \{ x_1, \ldots, x_n \}$,

is there a satisfying truth assignment?

$$(x \vee y \vee z) \wedge (x \vee \neg y \vee z) \wedge (\neg x \vee \neg y \vee \neg z) \wedge (z \vee \neg x \vee y)$$

NP-Complete

Definition (NP-Completeness proof)

1. Prove that $L \in NP$.
   - Describe a nondeterministic algorithm A.
   - Prove that A is in polynomial time.
2. Select a known NP-Complete problem L'.
3. Give a reduction R from L' to L.
4. Prove that R satisfies: $x' \in L' \iff x \in L$.
5. Prove that R is in polynomial time.

NP-Complete

Example (Independent Set(IS) $\to$ Vertex Cover(VC))

1. Prove that $VC \in NP$.
2. Reduction R from IS to VC: $(G, k) \to (G' = G, n - k)$
3. Prove that R satisfies:

$$G \text{ has IS with size } k(\mathcal{I}) \iff G' \text{ has VC with size } n - k(\mathcal{C})$$

   - $\Rightarrow$: $\mathcal{I} \to \mathcal{C} = V - \mathcal{I}$.
   - $\Leftarrow$: $\mathcal{C} \to \mathcal{I} = V - \mathcal{C}$.
4. Prove that R is in polynomial time. Trivial.

NP-Complete

Example (Independent Set(IS) $\rightarrow$ Clique(CQ))

1. Prove that $CQ \in NP$.
2. Reduction R from IS to CQ:
   $\big(G = (V, E), k\big) \rightarrow \big(\bar{G} = (V, \bar{E}), k\big)$
3. Prove that R satisfies:

   $$G \text{ has IS with size } k(\mathcal{I}) \iff G' \text{ has CQ with size } k(\mathcal{C})$$

   - $\Rightarrow: \mathcal{I} \rightarrow \mathcal{C} = \mathcal{I}$.
   - $\Leftarrow: \mathcal{C} \rightarrow \mathcal{I} = \mathcal{C}$.
4. Prove that R is in polynomial time. Trivial.

Summary

1. P vs. NP
2. NP proof
3. Reduction
4. NP-Complete
5. NP-Complete proof