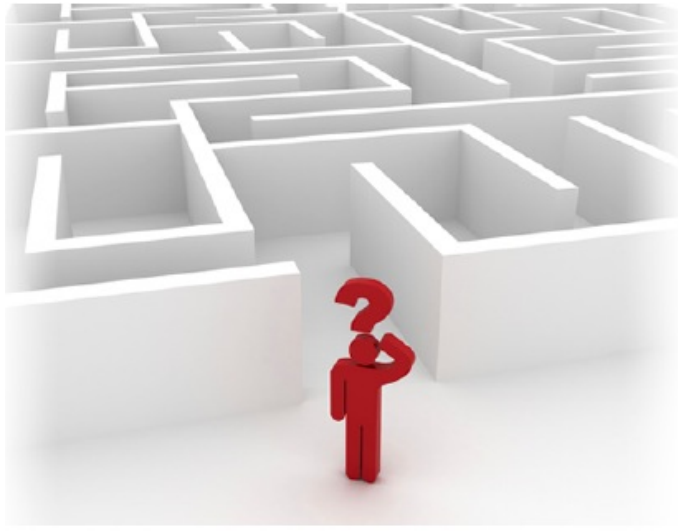

NP-Completeness

Algorithmic Approaches to NP-Complete Problems

Algorithms: Design and Analysis, Part II

NP-Completeness: The Beginning, Not the End

Question: So your problem is NP-complete. Now what?

Important: NP-completeness not a death sentence.

$\Rightarrow$ but, need appropriate expectations/strategy

Three useful strategies:

(1) Focus on computationally tractable special cases

Examples: - WIS in path graphs (and trees, bounded tree width) (NP-c in general graphs)

- Knapsack with polynomial size capacity (e.g., $W = O(n)$)

- 2SAT (P) instead of 3SAT (NP-c)

- Vertex cover when OPT is small

Three Useful Strategies (con'd)

(2) Heuristics - fast algorithms that are not always correct

Examples (forthcoming): Greedy and dynamic programming-based heuristics for knapsack.

(3) Solve in exponential time but faster than brute-force search.

- Knapsack ($O(n)$ instead of 2^n)
- TSP ($\approx 2^n$ instead of $\approx n!$) (forthcoming)
- Vertex cover ($\approx 2^{\mathrm{OPT}} n$ instead of n^{OPT}) (forthcoming)