Computational Complexity III: Limits of Computation

Maria-Eirini Pegia

School of Informatics Thessaloniki

Seminar on Theoretical Computer Science and Discrete Mathematics
Aristotle University of Thessaloniki

Context

Computability vs Complexity

Computability

What can be computed and what can not be computed?

Complexity

What can be computed fast and what can not be computed?

Computability vs Complexity

Computability

What can be computed and what can not be computed?

Complexity

What can be computed fast and what can not be computed?

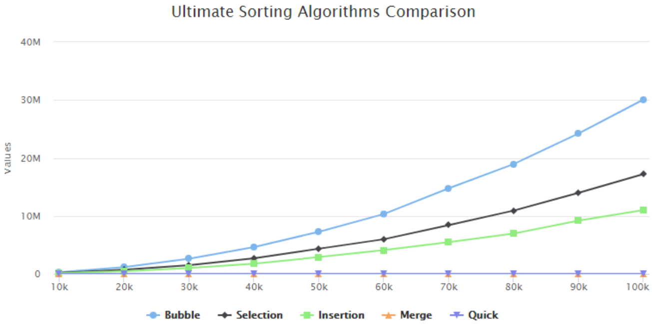


Figure: Comparison of sorting algorithms

Time Complexity of DTM

Definition

Let t: $\mathbb{N} \longrightarrow \mathbb{N}$ increasing function. The **time complexity** of DTIME[t(n)] is the collection of all languages that are decidable by an $O(t(n))$ time DTM.

DTIME[t(n)] ≡ { P: P is solved in O(t(n)) time }

Time Complexity of DTM

Definition

Let t: $\mathbb{N} \longrightarrow \mathbb{N}$ increasing function. The **time complexity** of DTIME[t(n)] is the collection of all languages that are decidable by an $O(t(n))$ time DTM.

DTIME[t(n)] ≡ { P: P is solved in O(t(n)) time }

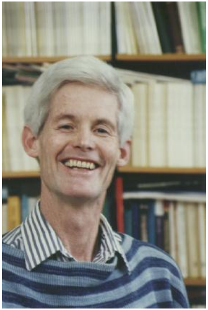
Definition

Complexity class $\mathcal{P}$ is the set of decision problems that can be solved by a DTM in a polynomial time of steps.

$\mathcal{P} \equiv \bigcup_{k \geq 0} DTIME[n^k]$

Cook - Karp Thesis

The **Cook - Karp Thesis** states that decision problems that are "tractably computable" can be computed by a DTM in polynomial time, i.e., are in $\mathcal{P}$.



Time Complexity of NTM

Definition

Let t: $\mathbb{N} \longrightarrow \mathbb{N}$ increasing function. The **time complexity** of NTIME[t(n)] is the collection of all languages that are decidable by an $O(t(n))$ time NTM.

NTIME[t(n)] ≡ { P: P is solved in non deterministic time O(t(n)) }

Time Complexity of NTM

Definition

Let t: $\mathbb{N} \longrightarrow \mathbb{N}$ increasing function. The **time complexity** of NTIME[t(n)] is the collection of all languages that are decidable by an $O(t(n))$ time NTM.

NTIME[t(n)] ≡ { P: P is solved in non deterministic time O(t(n)) }

Definition

Complexity class $\mathcal{NP}$ is the set of decision problems that can be solved by a NTM in a polynomial time of steps or is the set of decision problems for which there exists a **poly time certifier**.

$\mathcal{NP} \equiv \bigcup_{k\geq 0} NTIME[n^k]$

$\mathcal{P}$ vs $\mathcal{NP}$

How much easier is to **find** a solution than to **confirm** it?

$\mathcal{P}$ vs $\mathcal{NP}$

How much easier is to **find** a solution than to **confirm** it?

$$99799811 = ? \times ?$$

$\mathcal{P}$ vs $\mathcal{NP}$

How much easier is to **find** a solution than to **confirm** it?

$$99799811 = ? \times ?$$

$$997 \times 10007 = 99799811$$

$\mathcal{P}$ vs $\mathcal{NP}$

How much easier is to **find** a solution than to **confirm** it?

$$99799811 = ? \times ?$$

$$997 \times 10007 = 99799811$$

Theorem

$\mathcal{P} \subseteq \mathcal{NP}$

$\mathcal{P}$ vs $\mathcal{NP}$

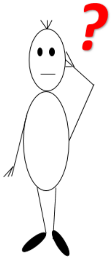
How much easier is to **find** a solution than to **confirm** it?

$$99799811 = ? \times ?$$

$$997 \times 10007 = 99799811$$

Theorem

$\mathcal{P} \subseteq \mathcal{NP}$



Open Problem: $\mathcal{P} \supseteq^{???} \mathcal{NP}$

TSP $\in \mathcal{NP}$

Travelling Salesman Problem (TSP): Given a set of distances on n cities and a bound D, is there a tour of length at most D?

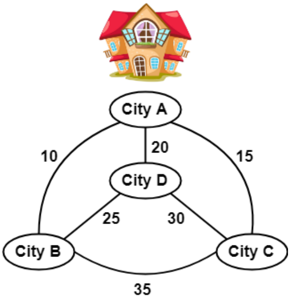


Figure: TSP

TSP ∈ $\mathcal{NP}$

Travelling Salesman Problem (TSP): Given a set of distances on n cities and a bound D, is there a tour of length at most D?

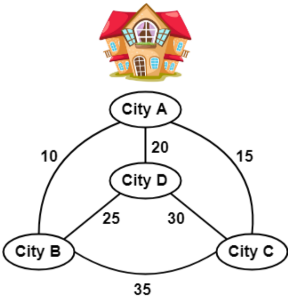


Figure: TSP

Certificate: A tour of given graph.

Certifier:

1. Check that each city appears once.
2. Check that the length of tour is at most D.

Context

Polynomial time reduction

Figure: The casting process

Polynomial time reduction

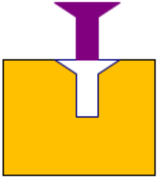


Figure: The casting process

Polynomial time reduction

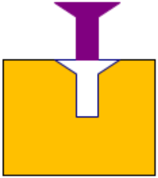


Figure: The casting process

Polynomial time reduction

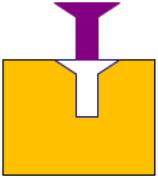


Figure: The casting process

reduction →

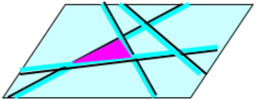


Figure: Half plane intersection

Polynomial time reduction

If a problem X reduces to a problem Y, then a solution to Y can be used to solve X. **(Y is at least as hard as X)**

Polynomial time reduction

If a problem X reduces to a problem Y, then a solution to Y can be used to solve X. **(Y is at least as hard as X)**

Definition

$X \in \mathcal{NP}$-complete if:

- $X \in \mathcal{NP}$
- $\forall\ Y \in \mathcal{NP},\ Y \leq_P X$

Polynomial time reduction

If a problem X reduces to a problem Y, then a solution to Y can be used to solve X. **(Y is at least as hard as X)**

Definition

$X \in \mathcal{NP}$-complete if:

- $X \in \mathcal{NP}$
- $\forall\ Y \in \mathcal{NP},\ Y \leq_P X$

Hamiltonian Cycle Problem $\xrightarrow{\textit{reduction}}$ Travelling Salesman Problem

Hamiltonian Cycle Problem

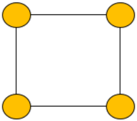
Hamiltonian Cycle Problem

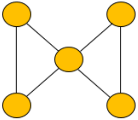
Let $G = (V, E)$ a graph.
Find whether G contains a cycle that passes through all vertices of the graph exactly once.

Hamiltonian Cycle Problem

Hamiltonian Cycle Problem

Let $G = (V, E)$ a graph.
Find whether G contains a cycle that passes through all vertices of the graph exactly once.

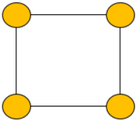


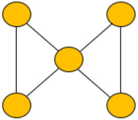


Hamiltonian Cycle Problem

Hamiltonian Cycle Problem

Let $G = (V, E)$ a graph.
Find whether G contains a cycle that passes through all vertices of the graph exactly once.





Travelling Salesman Problem

Let $G' = (V', E')$ a weighted graph with non negative weights and $k' \in \mathbb{Z}$.
Find whether G' contains a cycle that passes through all vertices of the graph exactly once and has length $\leq k'$.

Goal of the study of $\mathcal{NP}$ - completeness

If some $\mathcal{NP}$ - complete problem P is in $\mathcal{P}$, then $\mathcal{P} = \mathcal{NP}$.

Goal of the study of $\mathcal{NP}$ - completeness

If some $\mathcal{NP}$ - complete problem P is in $\mathcal{P}$, then $\mathcal{P} = \mathcal{NP}$.

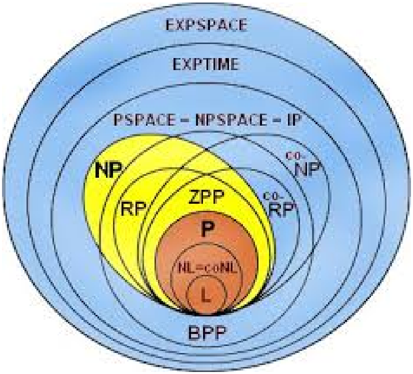


Figure: Scott Aaronson

Context

Definition

Let s: $\mathbb{N} \longrightarrow \mathbb{N}$ increasing function. The **space complexity** of DSPACE[t(n)] is the collection of all languages that are decidable by an $O(s(n))$ space DTM.

NSPACE[s(n)] ≡ { P: P is solved in O(s(n)) space}

Definition

Let s: $\mathbb{N} \longrightarrow \mathbb{N}$ increasing function. The **space complexity** of DSPACE[t(n)] is the collection of all languages that are decidable by an $O(s(n))$ space DTM.
NSPACE[s(n)] ≡ { P: P is solved in O(s(n)) space}

Definition

Complexity class *PSPACE* is the set of decision problems that can be solved by a (multitape) DTM in a polynomial number of SPACEs on the tape.
$PSPACE \equiv \bigcup_{k \geq 0} DSPACE[n^k]$

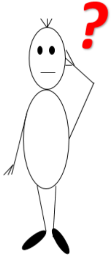
Theorem

$\mathcal{P} \subseteq PSPACE$

Theorem

$\mathcal{P} \subseteq PSPACE$

Open Problem:
$\mathcal{P} \supseteq^{???} \mathcal{NP} \supseteq^{???} PSPACE$



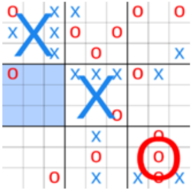
PSPACE-complete

Definition

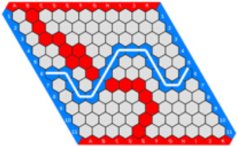
$X \in PSPACE$-complete if:

- $X \in PSPACE$
- $\forall\ Y \in PSPACE,\ Y \leq_P X$

GAMES



ultimate tic tac toe



hex

go

Figure: PSPACE-complete problems

References

- De Berg, M., Van Kreveld, M., Overmars, M., Cheong, O.. Computational Geometry: Algorithms and Applications. Springer Verlag, 3rd Edition, 2008.
- Hopcroft, J. E., Ullman, J. D.. Introduction to Automata Theory, Languages, and Computation. Boston: Addison-Wesley, c2001.
- Kleinberg, J., Tardos, E.. Algorithm Design. Boston, Mass.: Pearson/Addison-Wesley, cop. 2006.
- Papadimitriou, C. H.. Computational Complexity. Reading, Mass.: Addison-Wesley, 1994.
- Garey, M.R., Johnson, D.S.. Computers and Intractability, W.H. Freeman & Co, 1979.

Thank you!