AUTOMATA THEORY AND FORMAL LANGUAGES

UNIT 8: COMPUTATIONAL COMPLEXITY

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

OUTLINE

- Theory of Computation and Computational Complexity Theory
- Computational Models. Turing Machine
- Non-Deterministic Turing Machines
- Classification of problems
- Summary

OUTLINE

- **Theory of Computation and Computational Complexity Theory**
- Computational Models. Turing Machine
- Non-Deterministic Turing Machines
- Classification of problems
- Summary

Theory of Computation:

- Branch of the Theoretical Computer Science that deals with whether and how efficiently problems can be solved on a model of computation, using an algorithm.

- It studies which problems are decidable using different formal Computation Models (Turing Machines, Lambda calculus, recursive functions, etc).

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

- **Computational Complexity Theory:**
  - Branch of the Theory of Computation focused on classifying computational problems according to their inherent difficulty.
  - It studies the complexity level of the algorithm required to solve a decidable problem.

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Universidad Carlos III de Madrid
www.uc3m.es

OUTLINE

- Theory of Computation and Computational Complexity Theory
- **Computational Models. Turing Machine**
- Non-Deterministic Turing Machines
- Classification of problems
- Summary

Utilities of a Turing Machine

- Formalizes the concept of algorithm.
- It is a formal model of a computer, i.e.
  - "The recognition of the strings that make up a language is a formal way to express any problem."
- Theoretical *human* computer.
- Provides means to prove:
  1. Whether **there is an algorithm** or procedure to solve a specific problem
  2. The **form/complexity** of the algorithm
  3. Time spent by this algorithm or procedure to solve the problem.

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

OUTLINE

- Theory of Computation and Computational Complexity Theory
- Computational Models. Turing Machine
- **Non-Deterministic Turing Machines**
- Classification of problems
- Summary

Non-Deterministic Turing Machine

- They are similar to deterministic TM except:
  - Zero or more alternatives can be defined for the transitions between states
  - the machine can choose the transition to be used.
  - The best of the possible transitions is taken into account to estimate complexity (the machine will guess it).
  - The transition function δ for the nondeterministic case is as follows:

$$\delta : Q \times \Gamma \longrightarrow \mathcal{P}(Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}).$$

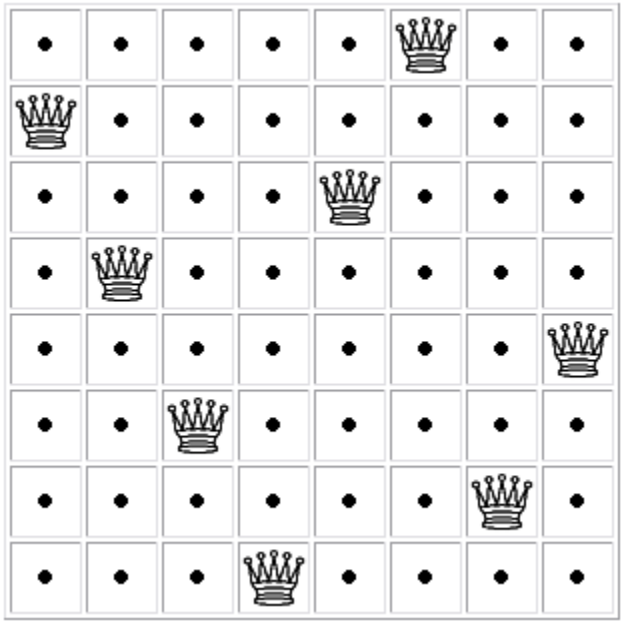
is the set defined as parts of

$$\mathcal{P}(Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}). \qquad Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}$$

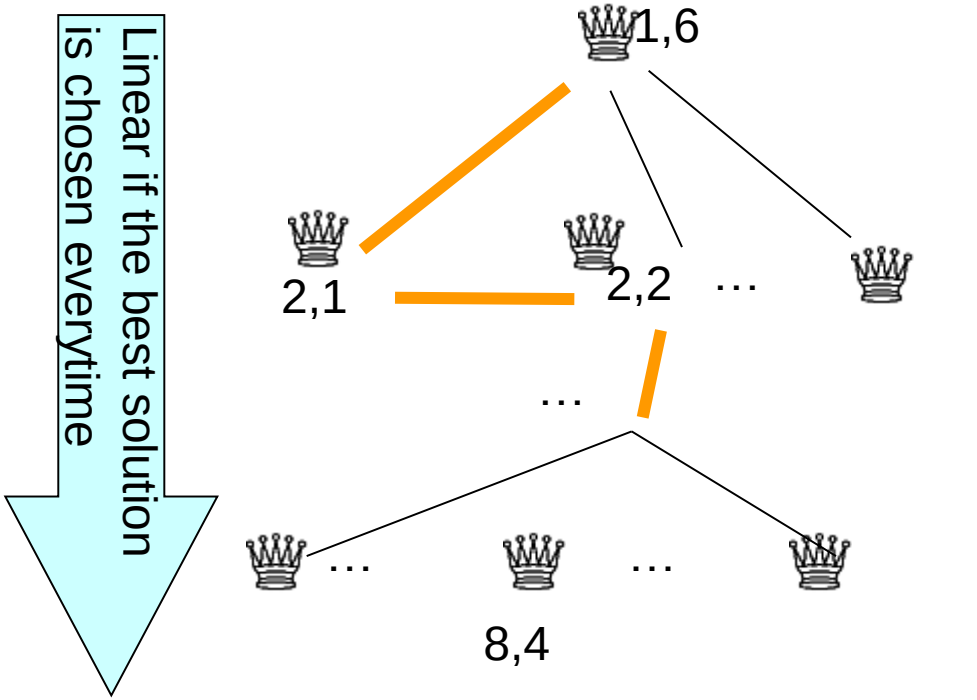
David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Non-Deterministic Turing Machine

- **Eight queens puzzle**



Place eight chess queens on an 8×8 chessboard so that no two queens attack each other.



http://en.wikipedia.org/wiki/Eight_queens_puzzle

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Non-Deterministic Turing Machine

- Calculations in a ND-TM can be represented as a tree whose branches denote the possible possibilities for each transition.

  If a “branch” reaches an acceptance state, the ND-TM accepts the input.

- Theorem:

  Every Non-Determinist TM has an equivalent Determinist TM

- Therefore, Non-Determinist TM and Determinist TM are equivalent considering the computing capacity.

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

OUTLINE

- Theory of Computation and Computational Complexity Theory
- Computational Models. Turing Machine
- Non-Deterministic Turing Machines
- **Classification of problems**
- Summary

Classification of Problems

- Criterion (Theory of Computation):
  - Decision Problem (Definition, instance, YES o NOT answer). Determine whether a “word” is included in the language or not.
  - Types:
    - Decidable.
    - Partially Decidable (Recognizable).
    - Not Decidable.

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Classification of Problems

- Criterion (Computational Complexity Theory):
  - In terms of a Deterministic or Non-Deterministic TM solving the problem in a requiring polynomial time.
  - Types:
    - Class P, NP, NP-Complete (CC Temporal).
    - Class PSPACE, PSPACE-Complete, L, NL,NL-Complete (CC-Spatial).

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Classification of Problems

- Let *A* be the universal language, *TM* a Turing Machine and *x* Є A:

  Determine if *x* included in the language is a decision problem.

- Considering the *Theory of Computation,* **a Decision Problem can be:**

  1. <u>Decidable</u> (o with solution using an algorithm):

  - It it is possible to find a device (TM) to solve the problem (i.e., for every *x* the TM returns a1 or a 0 if *x* is included in the language or not.
  - In addition, the TM stops for every element in A (every input).

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Classification of Problems

- Let *A* be the universal language, *TM* a Turing Machine and *x* Є A:
  Determine if *x* included in the language is a decision problem.

- Considering the *Theory of Computation,* **a Decision Problem can be:**

  2. <u>Partially Decidable</u> (Recognizable):

  - It is possible to find a device (TM) to solve the problem (i.e., for every *x* the TM returns a1 or a 0 if *x* is included in the language or not.
  - In addition, the TM stops for every element in A included in the language recognized by the TM.

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Classification of Problems

- Let *A* be the universal language, *TM* a Turing Machine and *x* Є A:
  Determine if *x* included in the language is a decision problem.

- Considering the *Theory of Computation,* **a Decision Problem can be:**

  3. <u>No Decidable</u>:
  - The problem is not decidable.

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Classification of Problems

□ Criterion (Computational Complexity Theory):

- Categorize computational problems according to their behavior related to the Computational Complexity Theory:
  - Class **P** and Class **NP,**
  - whose solutions in a **polynomial time** are found considering **the determinism or non-determinism behavior respectively** of the TM.

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Classification of Problems

□ Criterion (Computational Complexity Theory):

- **Class P**:
  - Problems that a Deterministic TM can solve in a polynomial time related to the length of the input string.
    - Example: The language $L=\{a^n b^n c^n \mid n>=0\} \in P$ can be solved in a polynomial time using a Deterministic TM.
  - Most of the usual problems are included in this class:
    - Order, Search, Operations with arrays…

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Classification of Problems

- Criterion (Computational Complexity Theory):
  - **Class NP**:
    - Problems that a Non-Deterministic TM can solve in a polynomial time related to the length of the input string.

Given that every Deterministic TM is a particular case of a Non-Deterministic TM:

$$P \subseteq NP$$

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

Classification of Problems

- Criterion (Computational Complexity Theory):

- **Class NP-Complete**:
  - A specific problem P is NP-Complete if:
    - It is P
    - All the other problems of NP can be reduced to it in polynomial time in a Deterministic-TM.

Reduce a problem::

* Transform a problem into another and then use the solution of the latter to obtain the solution of the former problem.

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es

OUTLINE

- Theory of Computation and Computational Complexity Theory
- Computational Models. Turing Machine
- Non-Deterministic Turing Machines
- Classification of problems
- **Summary**

Summary

- We have described the most common concepts and problems.
- There are problems which are more complex than other according to our computational model (Turing Machine)
- Although we cannot solve the most complex problems, we can know whether a specific problem is included in this category or not.
- The most complex problem is the NP-Complete.

David Griol Barres - Computer Science Department – UC3M - dgriol@inf.uc3m.es