Pina Kolling
piko0011

Summary: Lecture 5

Summary for the chapters X and X. [4]

Reduction

Examples of NP-problems:

- Travelling Salesman Problem
- SATISFIABLE
- REACHBILITY (in P)
- CIRCUIT VALUE (in P)

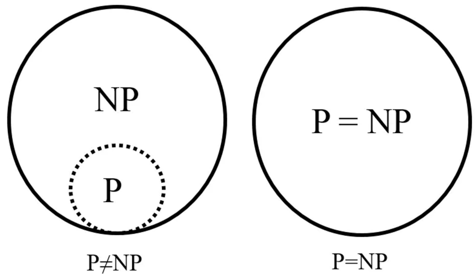

Figure 1: P and NP sets [2]

- reduction: a problem is at least as hard as another
- problem A is at least as hard as problem B if B reduces to A
- B reduces to A if there is a transformation R
  - R produces for every input x of B an equivalent input $R(x)$ of A
  - the answer of input x on B and input $R(x)$ on A have to be the same
- to solve B on input x, A can be solved instead with input $R(x)$

Reduction

Problem A is at least as hard as problem B if B reduces to A.

Transformation function:

- tranformation function R should not be too hard to compute
  $\rightarrow R$ should be limited

- efficient reduction R: $\log n$ space bounded

  Transformation function

  A language L_1 is reducible to L_2 if there is a function R computable by a deterministic Turing Machine in space $O(\log n)$ and $x \in L_1 \Leftrightarrow R(x) \in L_2$.

  R is called a reduction from L_1 to L_2.

- A Turing Machine M that computes a reduction R halts for all inputs x after a polynomial number of steps.
  - there are $O(n \cdot c^{\log n})$ possible configurtions for M on an input of length n
  - deterministic: no configuration can be repeated
  - computation of length at most $O(n^k)$

Pina Kolling
piko0011

Reduction HAMILTONIAN PATH to SATISFIABLE

Problem: HAMILTON PATH

The Hamiltonian Path problem asks whether there is a route in a directed graph G from a start node to an ending node, visiting each node exactly once. (Is there a path in G that visits each node one?) [1]

Problem: SAT

The SAT (satisfiability) problem is the problem of determining if there exists an interpretation that satisfies a given Boolean formula. [5]

- HAMILTON PATH can be reduced to SAT
  $\rightarrow$ demonstrates HAMILTON PATH is not significantly harder that SAT
- construct a boolean expression $R(G)$ that is satisfiable only if G has a Hamilton path
  $\rightarrow$ write a logical formular that only becomes true when HP is true
- instance: Graph $G = (V, E)$ with n nodes $(1, 2, ..., n)$
- $R(G)$ has n^2 boolean variables $x_{i,j}$ then
- node j is the ith node in the HAMILTON PATH
- $R(G)$ is in conjuctive normal form (CNF: $(a \vee b) \wedge (\neg a \vee c)$)
- conjuncted clauses of $R(x)$:
  - each node j must appear in the path $x_{1,j} \vee x_{2,j} \vee ... \vee x_{n,j}$ – for every node j
  - no node j appears twice in the path: $\neg x_{i,j} \vee \neg x_{k,j}$ for all i, j, k with $i \neq k$
  - every position i on the path must be occupied – $x_{i,1} \vee x_{i,2} \vee ... \vee x_{i,n}$ for each i
  - no two nodes j and k occupy the same position in the path – $\neg x_{i,j} \vee \neg x_{i,k}$ for all i, j, k with $j \neq k$
  - nonadjacent nodes i and j cannot be adjacent in the path – $\neg x_{k,i} \vee \neg x_{k+1,j}$ for all $(i, j) \notin E$ and $k = 1, 2, ..., n - 1$

  [3]

- log space reduction from HP to S
- 4, 3, 1, 2 as path
  $x_{1,4} = T, x_{2,3} = T, x_{3,1} = T, x_{4,2} = T,$
- slide is not quite correct
- $(not x_{1,1} or not x_{2,1}) and (not x_{1,1} or not x_{3,1})$
  $and (not x_{1,1} or not x_{4,1}) and (not x_{2,1} or not x_{3,1})$
  $and (not x_{2,1} or not x_{4,1}) and (not x_{3,1} or not x_{4,1}) and ...$
  first index: step, second: node

TODO
Questions:

Pina Kolling
piko0011

Boolean Circuits

TODO
Questions:

Reduction REACHABILITY PATH to CIRCUIT VALUE

TODO
Questions:

Reduction CIRCUIT SAT PATH to SAT

TODO
Questions:

Further examples

TODO
Questions:

Closedness under Composition

TODO
Questions:

References

[1] J. Baumgardner, K. Acker, O. Adefuye, and et al. "Solving a Hamiltonian Path Problem with a bacterial computer". In: *J Biol Eng* 3.11 (2009). DOI: `https://doi.org/10.1186/1754-1611-3-11`.

[2] *Image source: P-NP sets.* `https://www.techno-science.net/actualite/np-conjecture-000-000-partie-denouee-N21607.html`.

[3] Prof. Yuh-Dauh Lyuu. *Lecture slides.* `https://www.csie.ntu.edu.tw/~lyuu/complexity/2011/20111018.pdf`. 2011.

[4] Christos H. Papadimitriou. *Computational Complexity.* Addison-Wesley Publishing Company, 1994.

[5] Prof. Dr. Thomas Schwentick. *Lecture slides in Grundbegriffe der theoretischen Informatik.* `https://www.cs.tu-dortmund.de/nps/de/Studium/Ordnungen_Handbuecher_Beschluesse/Modulhandbuecher/Archiv/Bachelor_LA_GyGe_Inf_Modellv/_Module/INF-BfP-GTI/index.html`.