CMPT 307: Polynomial Time Algorithms

Week 12-3

Xian Qiu

Simon Fraser University

xianq@sfu.ca

Encoding Length

how to measure the size of an instance? encoding length of input

encoding graphs: $G = (V, E)$

- ▷ adjacency matrix: $\Theta(n^2)$
- ▷ adjacency lists: $\Theta(n + m)$

encoding numbers: natural number n

- ▷ binary: $\Theta(\log n)$
- ▷ k-ary ($k > 1$): $\Theta(\log_k n)$

Polynomial Time Algorithms

- P – optimization problem $\mathbb{A}$ – algorithm for P
- I – instance of P $|I|$ – encoding length of I
- $t_{\mathbb{A}}(I)$ – running time of algorithm $\mathbb{A}$ for instance I

Polynomial time algorithm

$\mathbb{A}$ is a polynomial time algorithm for P if for any instance $I \in P$

- A terminates with a solution s_I of I;
- there is a polynomial function p such that $t_A(I) \in O(p(|I|))$.

- A solves P if s_I is optimal for all I
- polynomial time algorithms are often regarded as "efficient"

Examples

- P with input: n natural numbers, using binary encoding
- $\mathbb{A}_1$ runs in $O(n^{100})$
- $\mathbb{A}_2$ runs in $O(nK)$, where K is the largest number of input

consider any instance $I \in P$, then $|I| \geq \{n, \log K\}$

- $n^{100} \leq |I|^{100}$, thus $\mathbb{A}_1$ is polynomial time algorithm
- $nK = n2^{\log K}$, if $|I| = \log K$, then $nK = n2^{|I|} \geq 2^{|I|}$, $t_{\mathbb{A}_2}(I)$ is exponential in $|I|$, thus $\mathbb{A}_2$ is not polynomial

is Counting-Sort a polynomial time algorithm?

Examples

PRIME: is $n \in \mathbb{N}$ a prime number?

PRIMALITY-TEST(n)

```
1 for i = 2 to n − 1 do
2     if n mod i = 0 then
3         return false;
4 return true;
```

- ▷ is PRIMALITY-TEST polynomial time?
- ▷ binary encoding: no!
- ▷ unary encoding: yes!
- ▷ for polynomial algorithms, encodings do not matter as long as they are "polynomially equivalent"

Polynomial Equivalence

- P – optimization problem I – instance of P
- encodings E_1, E_2 of lengths $|I|_1$, $|I|_2$, for any $I \in P$

Definition

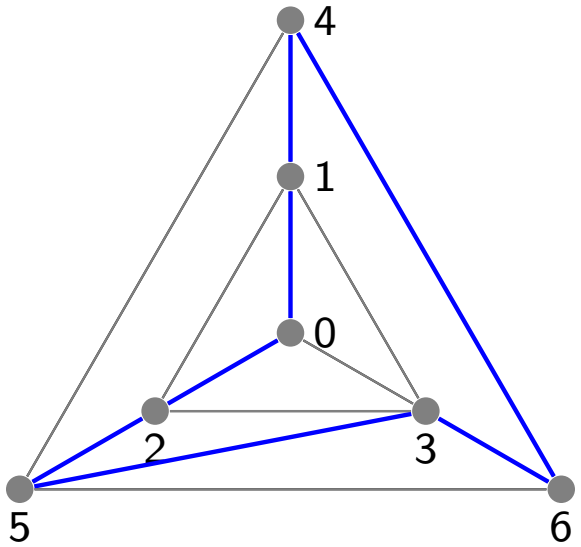
E_1 and E_2 are polynomially equivalent $\Leftrightarrow \exists$ polynomial functions p_1, p_2 s.t.,

$$|I|_1 \leq p_1(|I|_2) \text{ and } |I|_2 \leq p_2(|I|_1), \quad \forall I \in P.$$

- adjacency matrix and adjacency list are polynomially equivalent encodings for graphs
- all k-ary ($k > 1$ constant) encodings for natural numbers are polynomially equivalent $\log_k n = \log n / \log k$
- what about unary?

Hard Problems

traveling salesman problem (TSP)



- ▷ no polynomial (exact) algorithm has been found yet
- ▷ even fail to disprove the existence of such algorithms
- ▷ another perspective is to look at its level of difficulty

Decision Problem

Optimization Problem (OPT)

Given an instance $I = (S, f)$, find a solution $s^* \in S$ minimizing $f(s)$.

Decision Problem (DEC)

Given an instance $I = (S, f)$ and an integer k, decide if there exists a solution $s \in S$ with $f(s) \leq k$.

- I is called a yes-instance if there exists such a solution, otherwise I is a no-instance
- in other words: a decision problem is a set of instances with yes or no answer only

OPT & DEC

Observation

For many discrete optimization problems, if we can solve DEC in poly-time, then we can solve OPT in poly-time.

knowing an algorithm for DEC, how to solve the OPT?

1. compute optimal solution value α for OPT, by binary search
2. construct an optimal solution w.r.t. α

Example: Maximum Matching

DEC: given G and integer k, $\exists$ a matching of size $\geq k$?

know: algorithm $\mathbb{A}(G, k)$ tells us whether DEC is yes or no

SEARCH-MAX-VAL(G)

```
1 initialize: [a, b] = [0, n/2] ;                    // α ≤ n/2
2 while b > a do
3 |   k = ⌊(b − a)/2⌋;
4 |   if 𝔸(G, k) = "yes", then a := k;
5 |   else b := k − 1;
6 return a;
```

- ▷ SEARCH-MAX-VAL(G) returns the size of maximum matching
- ▷ running time $O(\log n) \times t_{\mathbb{A}}$

Example: Maximum Matching

MAXIMUM-MATCHING(G)

```
1 α = SEARCH-MAX-VAL(G);
2 for each edge e ∈ E do
3     if 𝔸(G − e, α) = yes then
4         G := G − e;
5 return G.E;
```

- ▷ running time $O(m + \log n) \times t_{\mathbb{A}}$
- ▷ MAXIMUM-MATCHING is polynomial if $\mathbb{A}$ is polynomial