Testing if a bounded-degree graph has a simple k-path

Joseph Chuang-Chieh Lin

Manuscript.

Speaker: Joseph Chuang-Chieh Lin
Advisor: Professor Maw-Shang Chang

Department of Computer Science and Information Engineering
National Chung Cheng University.
Taiwan

7 June 2011

Outline

1. Introduction
   - Property testing
   - Fixed-parameter algorithms
   - Parameterized property testers
   - Our contribution
2. Parameterized property testers for $\mathcal{P}_{k\text{-path}}$
   - An $O(4^k k^8/(\epsilon^4 d^3))$ parameterized tester
   - An $O(d^{k-2}/\epsilon)$ parameterized property tester

Outline

1. Introduction
   - Property testing
   - Fixed-parameter algorithms
   - Parameterized property testers
   - Our contribution

2. Parameterized property testers for $\mathcal{P}_{k\text{-path}}$
   - An $O(4^k k^8/(\epsilon^4 d^3))$ parameterized tester
   - An $O(d^{k-2}/\epsilon)$ parameterized property tester

Background of property testing

- General notion: Rubinfeld & Sudan [*SIAM J. Comput.* 1996].
  - Graph property testing: Goldreich, Goldwasser & Ron [*J. ACM* 1998].

Task of property testing

Input: An input object I and a designated property $\mathcal{P}$

Task: Fulfill the following requirements in $o(|I|)$ time.

- If I satisfies $\mathcal{P}$
  $\Rightarrow$ answer "yes" with probability $\geq \frac{2}{3}$;
- If I is **far** from satisfying $\mathcal{P}$
  $\Rightarrow$ answer "no" with probability $\geq \frac{2}{3}$.

- **Property testers**: algorithms accomplishing the above task.

Background of property testing

- General notion: Rubinfeld & Sudan [*SIAM J. Comput.* 1996].
  - Graph property testing: Goldreich, Goldwasser & Ron [*J. ACM* 1998].

Task of property testing

Input: An input object I and a designated property $\mathcal{P}$

Task: Fulfill the following requirements in $o(|I|)$ time.

- If I satisfies $\mathcal{P}$
  $\Rightarrow$ answer "yes" with probability $\geq \frac{2}{3}$;
- If I is **far** from satisfying $\mathcal{P}$
  $\Rightarrow$ answer "no" with probability $\geq \frac{2}{3}$.

- **Property testers**: algorithms accomplishing the above task.

Background of property testing

- General notion: Rubinfeld & Sudan [*SIAM J. Comput.* 1996].
  - Graph property testing: Goldreich, Goldwasser & Ron [*J. ACM* 1998].

Task of property testing

Input: An input object I and a designated property $\mathcal{P}$

Task: Fulfill the following requirements in $o(|I|)$ time.

- If I satisfies $\mathcal{P}$
  $\Rightarrow$ answer "yes" with probability $\geq \frac{2}{3}$;
- If I is **far** from satisfying $\mathcal{P}$
  $\Rightarrow$ answer "no" with probability $\geq \frac{2}{3}$.

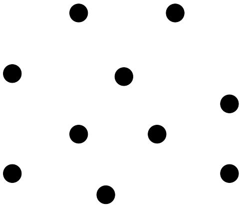
- **Property testers**: algorithms accomplishing the above task.

Background of property testing (contd.)

- In property testing, we use ϵ-far to say that the input is far from a certain property.
- ϵ: the least fraction of the input needs to be modified.
- For example, given a sequence of integers $L = (0, 2, 3, 4, 1)$ is 0.2-far from being monotonically nondecreasing.

An easy example of graph property testing

- Property: emptiness
  - ▷ i.e., being P_2-free.
- Graph model: the sparse model.
  - adjacency-list for graphs with vertex degree bounded by d
  - ⋆ $f_G : V(G) \times [d] \mapsto V(G) \cup \varnothing$
- Task: testing if a graph is empty.

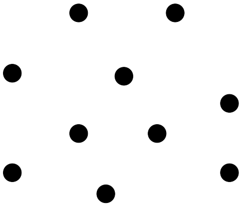


Time complexity: $O(1/\epsilon)$.

How can it be done?

An easy example of graph property testing

- Property: emptiness
  - ▷ i.e., being P_2-free.
- Graph model: the sparse model.
  - adjacency-list for graphs with vertex degree bounded by d
  - ⋆ $f_G : V(G) \times [d] \mapsto V(G) \cup \varnothing$
- Task: testing if a graph is empty.



Time complexity: $O(1/\epsilon)$.

How can it be done?

An easy example of graph property testing (contd.)

```
Emptiness-Tester(G)
```

1. **repeat**
   a. pick a vertex v and $i \in \{1, \ldots, d\}$ uniformly at random;
   b. **if** $f_G(v, i) \neq \varnothing$ **then** /* $\exists$ an edge incident to v */
      return "no";
2. **until** $2/\epsilon$ times;
3. **return** "yes";

An easy example of graph property testing (contd.)

ϵ-far from being empty: $\geq \epsilon dn$ entries are not $\varnothing$.

$$\mathbf{Pr}[f_G(v,i) \neq \varnothing] \geq \epsilon dn/dn = \epsilon$$
for random $v \in V$ and $i \in \{1, \ldots, d\}$.

- $\mathbf{Pr}[\texttt{Emptiness-Tester} \text{ returns "yes"} \mid \text{G is empty}] = 1$.
- $\mathbf{Pr}[\texttt{Emptiness-Tester} \text{ returns "yes"} \mid \text{G is } \epsilon\text{-far from being empty}] = (1-\epsilon)^{2/\epsilon} < e^{-2} < 1/3$.

An easy example of graph property testing (contd.)

ϵ-far from being empty: $\geq \epsilon dn$ entries are not $\varnothing$.

$$\mathbf{Pr}[f_G(v,i) \neq \varnothing] \geq \epsilon dn/dn = \epsilon$$
for random $v \in V$ and $i \in \{1, \ldots, d\}$.

- $\mathbf{Pr}[$`Emptiness-Tester` returns "yes" | G is empty $] = 1$.
- $\mathbf{Pr}[$`Emptiness-Tester` returns "yes" | G is ϵ-far from being empty$]$ $= (1-\epsilon)^{2/\epsilon} < e^{-2} < 1/3$.

Fixed-parameter algorithms

Fixed-parameter algorithms

Input: A problem instance I and a parameter $k \in \mathbb{Z}^+$.

Requirement: Solving the problem in $O(f(k)\text{poly}(n))$ time.

- $f(k)$: a function solely depending on k.

Parameterized property testers

Parameterized property testers

Input: A input object I, a designated property $\mathcal{P}$, $\epsilon > 0$, and a parameter $k \in \mathbb{Z}^+$.

Requirement: Testing if I has the property $\mathcal{P}$ with time complexity $O(f(k, 1/\epsilon))$.

- $f(k, 1/\epsilon)$: a function solely depending on k and ϵ.

Our contribution

- The notion of parameterized property testing.
- Two parameterized property testers for testing if a graph has a simple k-path in the sparse model.
  - The time complexities: $O(4^k k^8/(\epsilon^4 d^3))$ and $O(d^{k-2}/\epsilon)$.

Outline

$\mathcal{P}_{k\text{-path}}$: $\exists$ a simple k-path in the graph

- k-path: a path of length $k-1$.
  - ▷ $\mathcal{P}_{k\text{-path}}$: the property that a graph has a simple k-path.

The k-path problem

Input: A graph $G=(V,E)$ and an integer k
Task: Determine if G has a simple k-path.

- The k-path problem is **NP**-complete.
  - $k=n$: the Hamiltonian Path problem.

A randomized fixed-parameter algorithm looking for a k-path

Theorem 2.1 (Kneis *et al.* 2006)

Let G be the input graph. There exists an $O(4^k \cdot poly(n,k))$ algorithm such that

- *it finds a k-path in G with probability $\geq 3/4$ if G has one;*
- *it returns "no" if G does not have any k-path.*

- Actually, the factor $\text{poly}(n,k) = O(k^{\lg 3} k^2 \cdot dn^4)$.
- Denote by $\mathcal{A}$ such an algorithm.

Our first parameterized tester for $\mathcal{P}_{k\text{-path}}$

k-Path-FPT-Tester1(G, k)

1. **if** $k - 1 \leq \epsilon dn$ **then** /* G is ϵ-close to $\mathcal{P}_{k\text{-path}}$ */
   a. **return** "yes";
2. **else** /* $n < (k-1)/\epsilon d$ */
   a. run the randomized fixed-parameter algorithm $\mathcal{A}$;
   b. **if** $\mathcal{A}$ finds a simple k-path **then return** "yes";
      else return "no";
3. **end if**

Theorem 2.2

Algorithm `k-Path-PT-Tester1` *is an* $O(4^k k^8/(\epsilon^4 d^3))$ *parameterized property tester for* $\mathcal{P}_{k\text{-path}}$.

- Any graph is ϵ-close to $\mathcal{P}_{k\text{-path}}$ when $k - 1 \leq \epsilon dn$.
- $k - 1 > \epsilon dn \Rightarrow n < k/(\epsilon d)$.
  - The complexity of Algorithm $\mathcal{A}$:

    $$O(4^k \cdot k^{\lg 3} k^2 \cdot d(k/\epsilon d)^4) = O(4^k k^8/(\epsilon^4 d^3)).$$

Another parameterized property tester for $\mathcal{P}_{k\text{-path}}$

k-Path-FPT-Tester2(G, k)

1. **if** $k - 1 \leq \epsilon dn$ **then** /* G is ϵ-close to $\mathcal{P}_{k\text{-path}}$ */
   a. **return** "yes";
2. **else** /* $n < (k-1)/\epsilon d$ */
   a. **repeat**
      i. choose a vertex v from G uniformly at random;
      ii. perform a BFS starting from v until all the vertices of distance $\leq k - 1$ from v are visited;
      iii. **if** a simple k-path is found **then return** "yes";
   b. **until** $\frac{2}{\epsilon d}$ times;
   c. **return** "no";
3. **end if**

Theorem 2.3

Algorithm `k-Path-PT-Tester2` *is an* $O(d^{k-2}/\epsilon)$ *parameterized property tester for* $\mathcal{P}_{k\text{-path}}$.

Sketch of the proof of Theorem 2.3

Consider the case that $k - 1 > \epsilon dn$.

- G satisfies $\mathcal{P}_{k\text{-path}}$:
  - **Pr**[a randomly chosen vertex is on a simple k-path] $\geq k/n > \epsilon d$.
  - **Pr**[No such a vertex found in the $2/(\epsilon d)$ repetitions] $\leq (1 - \epsilon d)^{2/\epsilon d} \leq e^{-2} < 1/3$.
- G is ϵ-far from $\mathcal{P}_{k\text{-path}}$:
  - Each connected component in G has diameter $\leq k - 2$.
  - The breadth-first search never finds a simple k-path.
- The complexity: $O(d^{k-1} \cdot 2/(\epsilon d))$.

Sketch of the proof of Theorem 2.3

Consider the case that $k - 1 > \epsilon dn$.

- G satisfies $\mathcal{P}_{k\text{-path}}$:
  - **Pr**[a randomly chosen vertex is on a simple k-path] $\geq k/n > \epsilon d$.
  - **Pr**[No such a vertex found in the $2/(\epsilon d)$ repetitions] $\leq (1 - \epsilon d)^{2/\epsilon d} \leq e^{-2} < 1/3$.
- G is ϵ-far from $\mathcal{P}_{k\text{-path}}$:
  - Each connected component in G has diameter $\leq k - 2$.
  - The breadth-first search never finds a simple k-path.
- The complexity: $O(d^{k-1} \cdot 2/(\epsilon d))$.

Sketch of the proof of Theorem 2.3

Consider the case that $k - 1 > \epsilon dn$.

- G satisfies $\mathcal{P}_{k\text{-path}}$:
  - $\mathbf{Pr}$[a randomly chosen vertex is on a simple k-path] $\geq k/n > \epsilon d$.
  - $\mathbf{Pr}$[No such a vertex found in the $2/(\epsilon d)$ repetitions] $\leq (1 - \epsilon d)^{2/\epsilon d} \leq e^{-2} < 1/3$.
- G is ϵ-far from $\mathcal{P}_{k\text{-path}}$:
  - Each connected component in G has diameter $\leq k - 2$.
  - The breadth-first search never finds a simple k-path.
- The complexity: $O(d^{k-1} \cdot 2/(\epsilon d))$.

Sketch of the proof of Theorem 2.3

Consider the case that $k - 1 > \epsilon dn$.

- G satisfies $\mathcal{P}_{k\text{-path}}$:
  - Pr[a randomly chosen vertex is on a simple k-path] $> k/n > \epsilon d$
  - Pr[No such a vertex found in the $2/(\epsilon d)$ repetitions] $\leq (1-\epsilon d)^{2/(\epsilon d)} \leq e^{-2} < 1/3$.
- G is ϵ-far from $\mathcal{P}_{k\text{-path}}$:
  - Each connected component in G has diameter $\leq k - 2$.
  - The breadth-first search never finds a simple k-path.
- The complexity: $O(d^{k-1} \cdot 2/(\epsilon d))$.

Sketch of the proof of Theorem 2.3

Consider the case that $k - 1 > \epsilon dn$.

- G satisfies $\mathcal{P}_{k\text{-path}}$:
- G is ϵ-far from $\mathcal{P}_{k\text{-path}}$:
  - Each connected component in G has diameter $\leq k - 2$.
  - The breadth-first search never finds a simple k-path.
- The complexity: $O(d^{k-1} \cdot 2/(\epsilon d))$.

Thank you.