Testing Tree-Consistency with k-Missing Quartets

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

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

The 28th Workshop on Combinatorial Mathematics and Computation Theory, 2011

28 May 2011

Outline

Background of property testing

- In the real world nowadays, we are faced with imperious need to process increasing larger amounts of data in faster times.
- Many practical problems have inputs of very large size.
- Sometimes it is not realistic to solve a problem in the time even linear in the input size.
- Then here comes property testing.

Background of property testing (contd.)

- Property testing: a new field in computational complexity theory and algorithm design.
- It delves into the possibilities of getting answers ("yes" or "no" for decision problems) by observing only a small fraction of the input.
- However, the decision problems considered here are *relaxed*:
  - Does the input satisfy a designated property, or
  - is far from satisfying the property?

Background of property testing (contd.)

- The idea was motivated by the connection to program checking
  - Blum, Luby, & Rubinfeld [*J. Comput. System Sci.* 1993]
- The general notion of property testing was first formulated by Rubinfeld and Sudan in 1996 [*SIAM J. Comput.* 1996]
- The study on testing *combinatorial objects* was first introduced by Goldreich, Goldwasser & Ron [*J. ACM* 1998]

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.

Property testers

- A property tester for $\mathbb{P}$ is an algorithm such that::
  - it answers "yes" with probability of $\geq 2/3$ if G satisfies $\mathbb{P}$, and
  - it answers "no" with probability of $\geq 2/3$ if G is ϵ-far from satisfying $\mathbb{P}$.

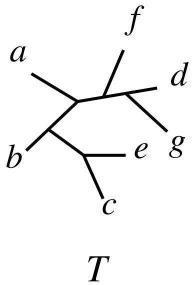
Property testers of one-sided error

- A property tester of one-sided error for $\mathbb{P}$ is an algorithm such that:
  - ▷ it answers "yes" with probability 1 if G satisfies $\mathbb{P}$, and
  - ▷ it answers "no" with probability of $\geq 2/3$ if G is ϵ-far from satisfying $\mathbb{P}$.

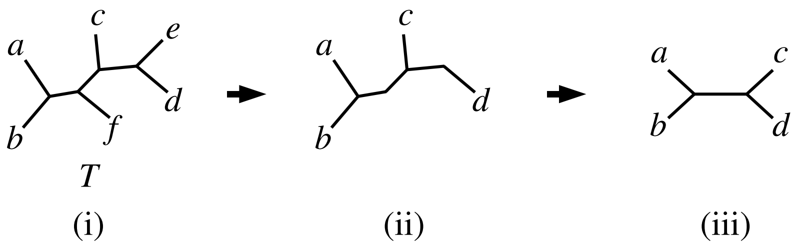
We focus on a property related to *evolutionary trees*.

Evolutionary trees

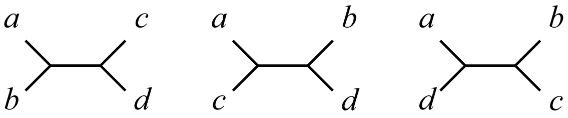
- S: a set of taxa; $|S| = n$.
- An evolutionary tree T on S:
  - An *unrooted*, *leaf-labeled* tree
  - The leaves are bijectively labeled by the taxa in S
  - Each internal node has degree *three*



Quartet topologies (contd.)



Quartet topologies (contd.)



$[ab|cd]$ $[ac|bd]$ $[ad|bc]$

Tree-consistency

- Q_T : the set of quartet topologies induced by T.
  - $|Q_T| = \binom{n}{4}$.
- Q is tree-consistent (with T):
  - $\exists T$ s.t. $Q \subseteq Q_T$.

  ▷ tree-like if $Q = Q_T$.
- Q is called complete:
  - Exactly one topology for every quartet;
  - Otherwise, incomplete.

Tree-consistency

- Q_T : the set of quartet topologies induced by T.
  - $|Q_T| = \binom{n}{4}$.
- Q is tree-consistent (with T):
  - $\exists T$ s.t. $Q \subseteq Q_T$.

▷ tree-like if $Q = Q_T$.

- Q is called complete:
  - Exactly one topology for every quartet;
  - Otherwise, incomplete.

Tree-consistency

- Q_T : the set of quartet topologies induced by T.
  - $|Q_T| = \binom{n}{4}$.
- Q is tree-consistent (with T):
  - $\exists T$ s.t. $Q \subseteq Q_T$.

  ▷ tree-like if $Q = Q_T$.

- Q is called complete:
  - Exactly one topology for every quartet;
  - Otherwise, incomplete.

Tree-consistency

- Q_T : the set of quartet topologies induced by T.
  - $|Q_T| = \binom{n}{4}$.
- Q is tree-consistent (with T):
  - $\exists T$ s.t. $Q \subseteq Q_T$.

  ▷ tree-like if $Q = Q_T$.
- Q is called complete:
  - Exactly one topology for every quartet;
  - Otherwise, incomplete.

Quartet errors

- Q: a set of quartet topologies;
  Q^*: a tree-like set of quartet topologies.
- **# quartet errors of Q w.r.t. Q^***: $Q \setminus Q^*$.
- **# quartet errors of Q**: $\text{err}(Q) = \min\{|Q \setminus Q^*| : Q^* \text{ is tree-like}\}$.

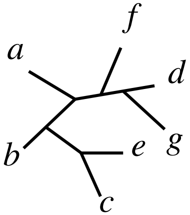
Quartet errors

- Q: a set of quartet topologies;
  Q^*: a tree-like set of quartet topologies.
- **# quartet errors of Q w.r.t. Q^*:** $Q \setminus Q^*$.
- **# quartet errors of Q:** $\mathrm{err}(Q) = \min\{|Q \setminus Q^*| : Q^* \text{ is tree-like}\}$.

Quartet errors

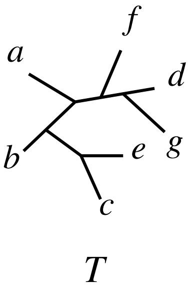
- Q: a set of quartet topologies;
  Q^*: a tree-like set of quartet topologies.
- **# quartet errors of Q w.r.t.** Q^*: $Q \setminus Q^*$.
- **# quartet errors of Q**: $\mathrm{err}(Q) = \min\{|Q \setminus Q^*| : Q^* \text{ is tree-like}\}$.

$[ab \mid cd]$, $[ab \mid ce]$, $[af \mid bc]$, $[ag \mid bc]$,
$[ad \mid be]$, $[ab \mid df]$, $[ab \mid dg]$, $[af \mid be]$,
$[ag \mid be]$, $[ab \mid fg]$, $[ad \mid ce]$, $[ac \mid df]$,
$[ac \mid dg]$, $[ac \mid ef]$, $[ag \mid ce]$, $[af \mid cg]$,
$[ae \mid df]$, $[ae \mid dg]$, $[af \mid dg]$, $[ae \mid fg]$,
$[bd \mid ce]$, $[bc \mid df]$, $[bc \mid dg]$, $[bf \mid ce]$,
$[bg \mid ce]$, $[bc \mid fg]$, $[be \mid df]$, $[be \mid dg]$,
$[bd \mid fg]$, $[be \mid fg]$, $[ce \mid df]$, $[ce \mid dg]$,
$[cf \mid dg]$, $[ce \mid fg]$, $[de \mid fg]$.



T

$[ab \mid cd]$, $[ab \mid ce]$, $[af \mid bc]$, $[ag \mid bc]$,
$[ad \mid be]$, $[ab \mid df]$, $[ab \mid dg]$, $[af \mid be]$,
$[ag \mid be]$, $[ab \mid fg]$, $[ad \mid ce]$, $[ac \mid df]$,
$[ac \mid dg]$, $[ac \mid ef]$, $[ag \mid ce]$, $[af \mid cg]$,
$[ae \mid df]$, $[ae \mid dg]$, $[af \mid dg]$, $[ae \mid fg]$,
$[bd \mid ce]$, $[bc \mid df]$, $[bc \mid dg]$, $[bf \mid ce]$,
$[bg \mid ce]$, $[bc \mid fg]$, $[be \mid df]$, $[be \mid dg]$,
$[bd \mid fg]$, $[be \mid fg]$, $[ce \mid df]$, $[ce \mid dg]$,
$[cf \mid dg]$, $[ce \mid fg]$, $[de \mid fg]$.



Previous work: testing tree-likeness

Testing tree-likeness

Input: A complete set Q of $\binom{n}{4}$ quartet topologies.
Task: Answer "yes" if Q is tree-like and answer "no" if Q is ϵ-far from being tree-like.

- Q is ϵ-far from being *tree-like*: $\text{err}(Q) \geq \epsilon\binom{n}{4}$.

⋆ $O(n^3/\epsilon)$ with one-sided error [Chang *et al.* 2010].

This paper: testing tree-consistency with k missing quartets

Testing tree-consistency with k missing quartets

Input: A set Q of $\binom{n}{4} - k$ quartet topologies & a set T_{miss} of k quartets whose topologies are unknown.
Task: Answer "yes" if Q is tree-consistent and answer "no" if Q is ϵ-far from being tree-consistent.

- Q is ϵ-far from being *tree-consistent*: $\text{err}(Q) \geq \epsilon\binom{n}{4}$.
- missing quartets: quartets with unknown topologies.

Contribution of this paper

Theorem

Given a set of quartet topologies Q and a set of k missing quartets $T_{miss} = \{\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_k\}$, there exists an $O(k3^k n^3/\epsilon)$ property tester to test if Q is tree-consistent.

Related works (Constructing T and QCP)

- Construct T by a given tree-like Q:
  - ⋆ $O(n^4)$ [Berry and Gascuel 2000].
- The Quartet Compatibility Problem (QCP):

  > Determine whether there exists an evolutionary tree T satisfying all quartet topologies in Q.

  - ⋆ **NP**-complete [Steel 1992].
  - ⋆ Polynomial time solvable if Q is complete [Erdős *et al.* 1999].
- Consider the cases of **complete** Q.

Related works (Constructing T and QCP)

- Construct T by a given tree-like Q:
  - ⋆ $O(n^4)$ [Berry and Gascuel 2000].
- The Quartet Compatibility Problem (QCP):

> Determine whether there exists an evolutionary tree T satisfying all quartet topologies in Q.

  - ⋆ **NP**-complete [Steel 1992].
  - ⋆ Polynomial time solvable if Q is complete [Erdős *et al.* 1999].
- Consider the cases of **complete** Q.

Related works (Constructing T and QCP)

- Construct T by a given tree-like Q:
  - $\star$ $O(n^4)$ [Berry and Gascuel 2000].
- The Quartet Compatibility Problem (QCP):

  > Determine whether there exists an evolutionary tree T satisfying all quartet topologies in Q.

  - $\star$ **NP**-complete [Steel 1992].
  - $\star$ Polynomial time solvable if Q is complete [Erdős *et al.* 1999].
- Consider the cases of **complete** Q.

Related works (MQI & MQC)

Minimum Quartet Inconsistency (MQI)

Construct an evolutionary tree $\mathcal{T}$ s.t. $|Q \setminus Q_{\mathcal{T}}|$ is minimized.

- **NP**-hard [Berry *et al.* 1999].
- Approx. ratio: $O(n^2)$ [Jiang *et al.* 2000].
- $O(3^n n^4)$ [Ben-Dor *et al.* 1998].
- $O(n^4)$ if $\mathrm{err}(Q) < (n-3)/2$ [Berry *et al.* 1999].
- $O(n^5 + 2^{4c} n^{12c+2})$ if $\mathrm{err}(Q) < cn$ [Wu *et al.* 2006].

Maximum Quartet Consistency (MQC)

Dual problem of MQI.

- **NP**-hard [Berry *et al.* 1999].
- PTAS [Jiang *et al.* 2001].

parameterized complexity

Determine if $\mathrm{err}(Q) \leq k$.

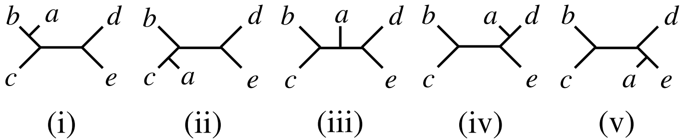
- $O(4^k n + n^4)$ [Gramm & Niedermeier 2003].
- $O(3.0446^k n + n^4)$, $O(2.0162^k n^3 + n^5)$, $O^*((1+\varepsilon)^k)$ [Chang *et al.* 2010].

Quintet topologies

- A quintet is a set of five taxa in S.
- Quintet topologies:

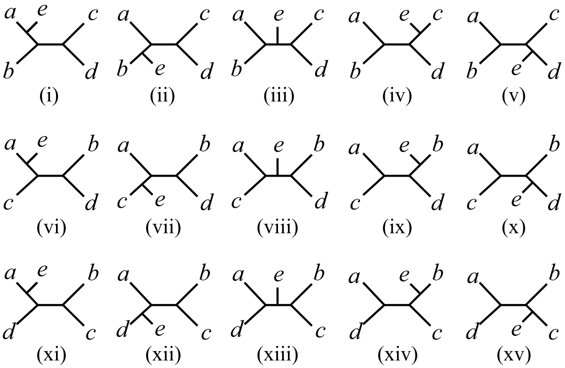
Quintet topologies

- A quintet is a set of five taxa in S.
- Quintet topologies:



Quintet topologies

- A quintet is a set of five taxa in S.
- Quintet topologies:



Resolved quintets

- What is a resolved quintet?

▷ $[ab|cd], [ab|ce], [ab|de], [ac|de], [bc|de] \in Q$.

Resolved quintets

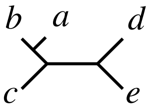
- What is a resolved quintet?

▷ $[ab|cd], [ab|ce], [ab|de], [ac|de]$,
$[bc|de] \in Q$.

Resolved quintets

- What is a resolved quintet?

▷ $[ab|cd], [ab|ce], [ab|de], [ac|de], [bc|de] \in Q$.



The crucial theorem

Theorem (Bandel & Dress 1986; Chang *et al.* 2010)

Given

- Q: *a complete set* Q *of quartet topologies over* S
- $\ell \in S$: *an arbitrarily fixed taxon*

Q *is tree-like* $\Leftrightarrow$ *every quintet containing* ℓ *is resolved.*

The tree-likeness tester [Chang *et al.* 2010]

`tree-like-tester`(Q)

/* Q: A complete set of quartet topologies. */

1. pick an arbitrary taxon $\ell \in S$;
2. **repeat**
   a. pick four taxa $s_1, s_2, s_3, s_4 \in S \setminus \{\ell\}$ uniformly at random;
   b. **if** the quintet $\{s_1, s_2, s_3, s_4, \ell\}$ is not resolved **then return** "no";
3. **until** the loop iterates for $\frac{72}{\epsilon}n^3$ times
4. **return** "yes";

Theorem (Chang *et al*. 2010)

Algorithm `tree-like-tester` *is a one-sided-error property tester for tree-likeness of quartet topologies, which makes at most* $O(n^3/\epsilon)$ *queries.*

A modified tree-likeness tester

tree-like-tester2(Q)

/* Q: A complete set of quartet topologies. */

1. pick an arbitrary taxon $\ell \in S$;
2. **repeat**
   a. pick four taxa $s_1, s_2, s_3, s_4 \in S \setminus \{\ell\}$ uniformly at random;
   b. **if** the quintet $\{s_1, s_2, s_3, s_4, \ell\}$ is not resolved **then return** "no";
3. **until** the loop iterates for $\frac{72(k+1)}{\epsilon} n^3$ times
4. **return** "yes";

Corollary

Algorithm `tree-like-tester2` *is a one-sided-error property tester for tree-likeness of quartet topologies, which makes at most* $O(kn^3/\epsilon)$ *queries.*

Furthermore, the tester returns "yes" with probability less than $1/3^{k+1}$ *whenever the input* Q *is* ϵ*-far from being tree-like.*

Exhaustively examining the topologies of the missing quartets

- There are 3^k possible assignments of topologies of the k missing quartets in $T_{miss} = \{\mathbf{t}_1, \ldots, \mathbf{t}_k\}$.
- For $1 \leq i \leq 3^k$, $Q_{miss}(i) = \{q_1(i), q_2(i), \ldots, q_k(i)\}$:

  the ith assignment of topologies of the missing quartets.
  - $q_j(i)$: the assigned topology of the quartet $\mathbf{t}_j$ in the ith assignment.

A tree-consistency tester with k missing quartets

```
dense-consistency-tester(Q)
/* Q: A set of (n choose 4) − k quartet topologies. */
/* T_miss: A set of k missing quartets */
1. for i ← 1 to 3^k do
   a. if tree-like-tester2(Q ∪ Q_miss(i)) returns "yes"
      then return "yes";
2. end for
3. return "no";
```

- Time complexity: $O(k3^k n^3/\epsilon)$.

A tree-consistency tester with k missing quartets

```
dense-consistency-tester(Q)
/* Q: A set of (n choose 4) − k quartet topologies. */
/* T_miss: A set of k missing quartets */
1. for i ← 1 to 3^k do
   a. if tree-like-tester2(Q ∪ Q_miss(i)) returns "yes"
      then return "yes";
2. end for
3. return "no";
```

- Time complexity: $O(k3^k n^3/\epsilon)$.

Correctness

Consider the case that Q is tree-consistent.

- There exists an evolutionary tree $\mathcal{T}$ such that $Q \subset Q_{\mathcal{T}}$.
- Algorithm `tree-like-tester2` exhaustively tries every assignment of topologies for the missing quartets in T_{miss}, so there must be some $j \in \{1, 2, \ldots, 3^k\}$ s.t. $Q_{miss}(j) = Q_{\mathcal{T}} \setminus Q$.
- The algorithm must return "yes".

Correctness

Consider the case that Q is tree-consistent.

- There exists an evolutionary tree $\mathcal{T}$ such that $Q \subset Q_{\mathcal{T}}$.
- Algorithm `tree-like-tester2` exhaustively tries every assignment of topologies for the missing quartets in T_{miss}, so there must be some $j \in \{1, 2, \ldots, 3^k\}$ s.t. $Q_{miss}(j) = Q_{\mathcal{T}} \setminus Q$.
- The algorithm must return "yes".

Correctness

Consider the case that Q is tree-consistent.

- There exists an evolutionary tree $\mathcal{T}$ such that $Q \subset Q_{\mathcal{T}}$.
- Algorithm `tree-like-tester2` exhaustively tries every assignment of topologies for the missing quartets in T_{miss}, so there must be some $j \in \{1, 2, \ldots, 3^k\}$ s.t. $Q_{miss}(j) = Q_{\mathcal{T}} \setminus Q$.
- The algorithm must return "yes".

Correctness (contd.)

Consider the case that Q is ϵ-far from being tree-consistent.

- For each $j \in \{1, 2, \ldots, 3^k\}$, $Q \cup Q_{miss}(j)$ must have at least $\epsilon\binom{n}{4}$ quartet errors.
- Algorithm `tree-like-tester2` is guaranteed to return "yes" with probability less than $(1/3)^{k+1}$ in this case (by the previous corollary).
- Thus, by the union bound, we obtain that Algorithm `dense-consistency-tester` returns "yes" with probability less than $(1/3)^{k+1} \cdot 3^k = 1/3$.

Correctness (contd.)

Consider the case that Q is ϵ-far from being tree-consistent.

- For each $j \in \{1, 2, \ldots, 3^k\}$, $Q \cup Q_{miss}(j)$ must have at least $\epsilon\binom{n}{4}$ quartet errors.
- Algorithm `tree-like-tester2` is guaranteed to return "yes" with probability less than $(1/3)^{k+1}$ in this case (by the previous corollary).
- Thus, by the union bound, we obtain that Algorithm `dense-consistency-tester` returns "yes" with probability less than $(1/3)^{k+1} \cdot 3^k = 1/3$.

Correctness (contd.)

Consider the case that Q is ϵ-far from being tree-consistent.

- For each $j \in \{1, 2, \ldots, 3^k\}$, $Q \cup Q_{miss}(j)$ must have at least $\epsilon\binom{n}{4}$ quartet errors.
- Algorithm `tree-like-tester2` is guaranteed to return "yes" with probability less than $(1/3)^{k+1}$ in this case (by the previous corollary).
- Thus, by the union bound, we obtain that Algorithm `dense-consistency-tester` returns "yes" with probability less than $(1/3)^{k+1} \cdot 3^k = 1/3$.

Concluding remarks

- Actually, it can be easily proved that testing tree-consistency with k missing quartets can be done in $O(3^k n^4)$ time.
- This paper extends the previous work by dealing with *incomplete* input sets of quartet topologies, and combines the concepts of *fixed-parameter algorithms* and property testing.
- It would be interesting to show that such a property does not admit a property tester with complexity independent of n.
- Extend our work to *triplets*.

Concluding remarks

- Actually, it can be easily proved that testing tree-consistency with k missing quartets can be done in $O(3^k n^4)$ time.
- This paper extends the previous work by dealing with *incomplete* input sets of quartet topologies, and combines the concepts of *fixed-parameter algorithms* and property testing.
- It would be interesting to show that such a property does not admit a property tester with complexity independent of n.
- Extend our work to *triplets*.

Concluding remarks

- Actually, it can be easily proved that testing tree-consistency with k missing quartets can be done in $O(3^k n^4)$ time.
- This paper extends the previous work by dealing with *incomplete* input sets of quartet topologies, and combines the concepts of *fixed-parameter algorithms* and property testing.
- It would be interesting to show that such a property does not admit a property tester with complexity independent of n.
- Extend our work to *triplets*.

Concluding remarks

- Actually, it can be easily proved that testing tree-consistency with k missing quartets can be done in $O(3^k n^4)$ time.
- This paper extends the previous work by dealing with *incomplete* input sets of quartet topologies, and combines the concepts of *fixed-parameter algorithms* and property testing.
- It would be interesting to show that such a property does not admit a property tester with complexity independent of n.
- Extend our work to *triplets*.

Thank you.