Definability Results for Top-Down Tree Transducers

Sebastian Maneth[1] Helmut Seidl[2] Martin Vu[1]

[1]Universität Bremen, Germany

[2]TU München, Germany

July 29, 2021

What are Tree Transducers?

Top-Down Tree Transducers (TOP) were invented by Rounds and Thatcher in the early 1970's as a formal model for compiler theory and linguistics.

What are Tree Transducers?

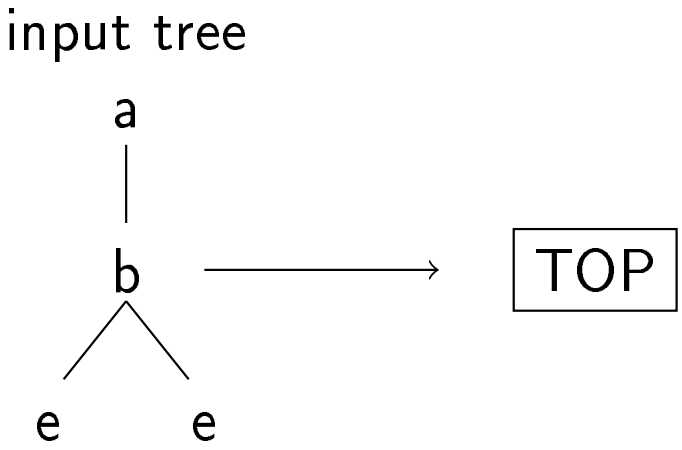
Top-Down Tree Transducers (TOP) were invented by Rounds and Thatcher in the early 1970's as a formal model for compiler theory and linguistics.

TOPs realize *tree translations*:

What are Tree Transducers?

Top-Down Tree Transducers (TOP) were invented by Rounds and Thatcher in the early 1970's as a formal model for compiler theory and linguistics.

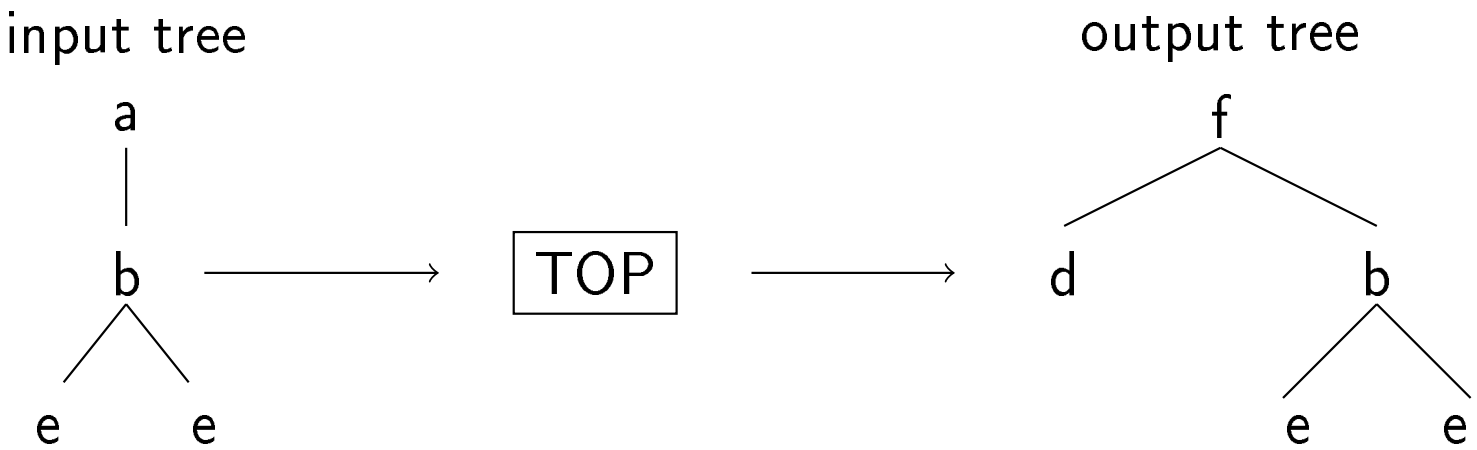
TOPs realize *tree translations*:



What are Tree Transducers?

Top-Down Tree Transducers (TOP) were invented by Rounds and Thatcher in the early 1970's as a formal model for compiler theory and linguistics.

TOPs realize *tree translations*:



Example

Axiom: $q(x_1)$

Rules:

$$
\begin{array}{llll}
q(a(x_1)) & \to & f(q'(x_1), \mathrm{i}(x_1)) & \mathrm{i}(b(x_1, x_2)) \to b(\mathrm{i}(x_1), \mathrm{i}(x_2)) \\
q'(b(x_1, x_2)) & \to & d & \mathrm{i}(e) \to e \\
q'(a(x_1)) & \to & e & \ldots
\end{array}
$$

Example

Axiom: $q(x_1)$

Rules:

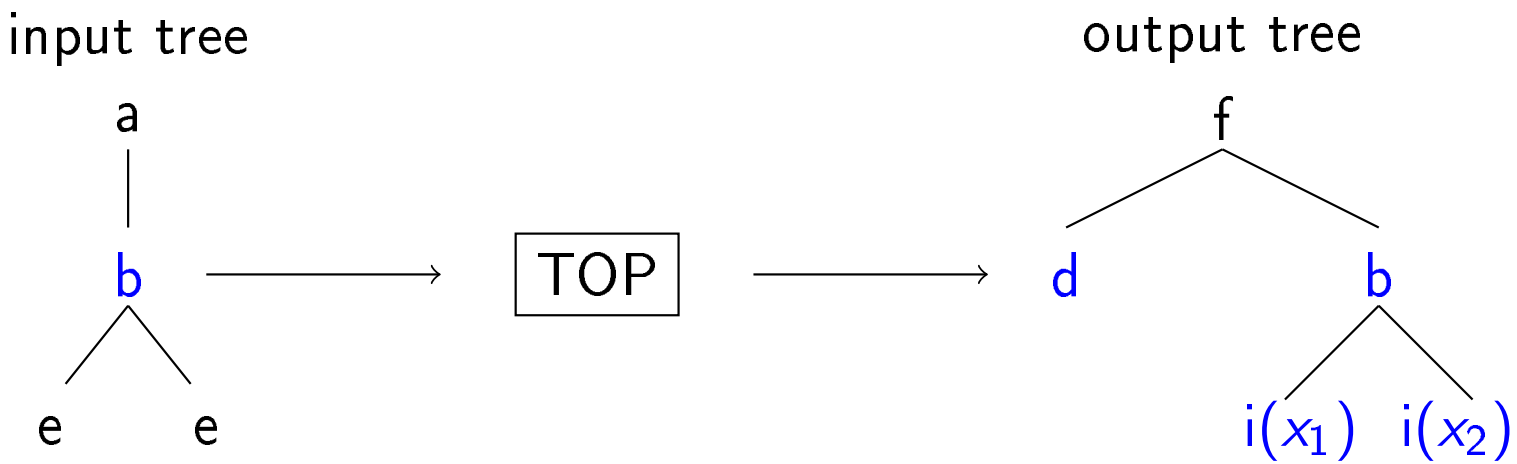
$$q(a(x_1)) \to f(q'(x_1), \mathrm{i}(x_1)) \qquad \mathrm{i}(b(x_1, x_2)) \to b(\mathrm{i}(x_1), \mathrm{i}(x_2))$$
$$q'(b(x_1, x_2)) \to d \qquad \mathrm{i}(e) \to e$$
$$q'(a(x_1)) \to e \qquad \ldots$$

Example

Axiom: $q(x_1)$

Rules:

$$q(a(x_1)) \rightarrow f(q'(x_1), \mathrm{i}(x_1)) \qquad \mathrm{i}(b(x_1, x_2)) \rightarrow b(\mathrm{i}(x_1), \mathrm{i}(x_2))$$
$$q'(b(x_1, x_2)) \rightarrow d \qquad \mathrm{i}(e) \rightarrow e$$
$$q'(a(x_1)) \rightarrow e \qquad \ldots$$



Example

Axiom: $q(x_1)$

Rules:

$$q(a(x_1)) \to f(q'(x_1), \mathrm{i}(x_1)) \qquad \mathrm{i}(b(x_1, x_2)) \to b(\mathrm{i}(x_1), \mathrm{i}(x_2))$$
$$q'(b(x_1, x_2)) \to d \qquad \mathrm{i}(e) \to e$$
$$q'(a(x_1)) \to e \qquad \ldots$$

Example

Axiom: $q(x_1)$

Rules:

$$q(a(x_1)) \to f(q'(x_1), \mathrm{i}(x_1)) \qquad \mathrm{i}(b(x_1, x_2)) \to b(\mathrm{i}(x_1), \mathrm{i}(x_2))$$
$$q'(b(x_1, x_2)) \to d \qquad \mathrm{i}(e) \to e$$
$$q'(a(x_1)) \to e \qquad \ldots$$

This TOP is not linear.

Our Result

Question: Given a TOP, is its translation definable by a linear transducer ($\mathrm{TOP}_{\mathrm{lin}}$)?

Our Result

Question: Given a TOP, is its translation definable by a linear transducer ($\mathrm{TOP}_{\mathrm{lin}}$)?

Answer:

Our Result

Question: Given a TOP, is its translation definable by a linear transducer ($\mathrm{TOP}_{\mathrm{lin}}$)?

Answer:

Decidable!

Our Result

Question: Given a TOP, is its translation definable by a linear transducer ($\mathrm{TOP}_{\mathrm{lin}}$)?

Answer:

Decidable!

In the affirmative case we can construct such a linear transducer.

Definability Question

Question: Given M in a class C, is M's translation definable in the class $C' \subset C$, i.e., is C' decidable within C?

Definability Question

Question: Given M in a class C, is M's translation definable in the class $C' \subset C$, i.e., is C' decidable within C?

Examples:

- decide subsequential string transducer within string transducer [Choffrut 1977]

Definability Question

Question: Given M in a class C, is M's translation definable in the class $C' \subset C$, i.e., is C' decidable within C?

Examples:

- decide subsequential string transducer within string transducer [Choffrut 1977]
- decide one-way string transducers within two-way string transducer [Baschenis, Gauwin, Muscholl, Puppis 2018]

Definability Question

Question: Given M in a class C, is M's translation definable in the class $C' \subset C$, i.e., is C' decidable within C?

Examples:

- decide subsequential string transducer within string transducer [Choffrut 1977]
- decide one-way string transducers within two-way string transducer [Baschenis, Gauwin, Muscholl, Puppis 2018]
- decide $\mathrm{TOP}_{\mathrm{lin}}$ within $\mathrm{TOP}^{R}_{\mathrm{lin}}$ [Maneth,Seidl 2020]

Definability Question

Question: Given M in a class C, is M's translation definable in the class $C' \subset C$, i.e., is C' decidable within C?

Why is this interesting?

Definability Question

Question: Given M in a class C, is M's translation definable in the class $C' \subset C$, i.e., is C' decidable within C?

Why is this interesting?

- implementing the class C' might requires less resources

Definability Question

Question: Given M in a class C, is M's translation definable in the class $C' \subset C$, i.e., is C' decidable within C?

Why is this interesting?

- implementing the class C' might requires less resources
- the class C' mght have better closure properties than C

Definability Question

Question: Given M in a class C, is M's translation definable in the class $C' \subset C$, i.e., is C' decidable within C?

Why is this interesting?

- implementing the class C' might requires less resources
- the class C' mght have better closure properties than C
- class specific advantages

Definability Question

Question: Given M in a class C, is M's translation definable in the class $C' \subset C$, i.e., is C' decidable within C?

Why is this interesting?

- implementing the class C' might requires less resources
- the class C' mght have better closure properties than C
- class specific advantages

specific advantages of $\mathrm{TOP}_{\mathrm{lin}}$ compared to TOP:

$\mathrm{TOP}_{\mathrm{lin}}$ preserves the regular tree language
(TOPs do not)
$\Rightarrow$ $\mathrm{TOP}_{\mathrm{lin}}$ allow for forward type checking

Is this Translation realizable by a Linear Transducer?

Axiom: $q(x_1)$

Rules:

$$q(a(x_1)) \rightarrow q'(x_1)$$
$$q'(b(x_1, x_2)) \rightarrow f(d, b(i(x_1), i(x_2)))$$
$$q'(a(x_1)) \rightarrow f(e, a(i(x_1)))$$

Is this Translation realizable by a Linear Transducer?

Axiom: $q(x_1)$

Rules:

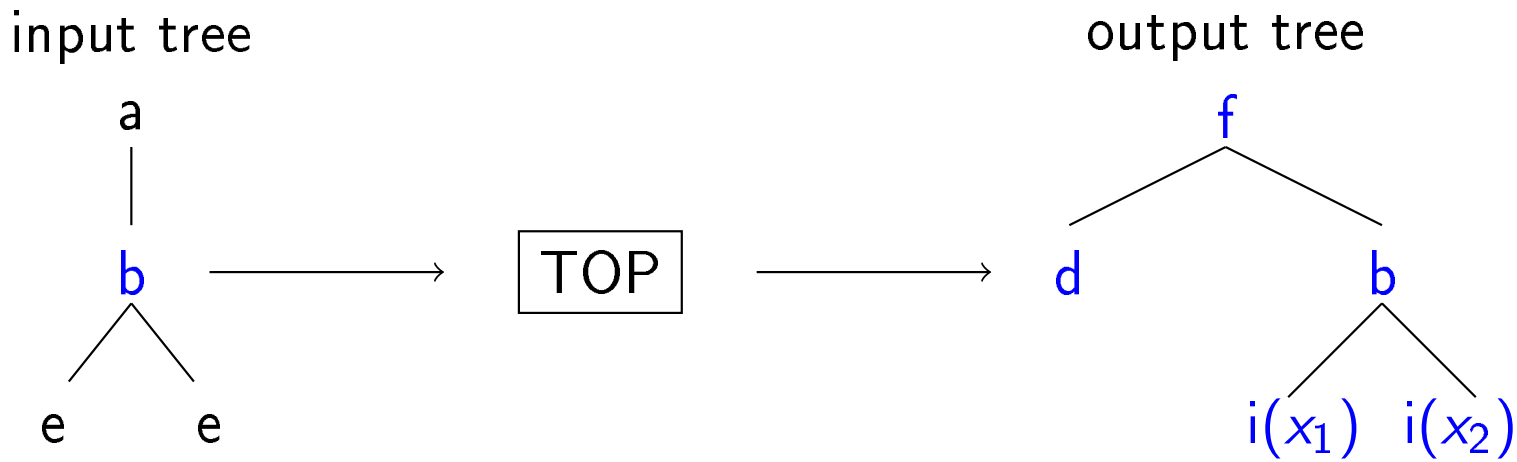
$$q(a(x_1)) \rightarrow q'(x_1)$$
$$q'(b(x_1, x_2)) \rightarrow f(d, b(i(x_1), i(x_2)))$$
$$q'(a(x_1)) \rightarrow f(e, a(i(x_1)))$$

Is this Translation realizable by a Linear Transducer?

Axiom: $q(x_1)$

Rules:

$$q(a(x_1)) \rightarrow q'(x_1)$$
$$q'(b(x_1, x_2)) \rightarrow f(d, b(i(x_1), i(x_2)))$$
$$q'(a(x_1)) \rightarrow f(e, a(i(x_1)))$$



Is this Translation realizable by a Linear Transducer?

Axiom: $q(x_1)$

Rules:

$$q(a(x_1)) \rightarrow q'(x_1)$$
$$q'(b(x_1, x_2)) \rightarrow f(d, b(i(x_1), i(x_2)))$$
$$q'(a(x_1)) \rightarrow f(e, a(i(x_1)))$$

Is this Translation realizable by a Linear Transducer?

Axiom: $q(x_1)$

Rules:

$$\begin{aligned} q(a(x_1)) &\to q'(x_1) \\ q'(b(x_1, x_2)) &\to f(d, b(i(x_1), i(x_2))) \\ q'(a(x_1)) &\to f(e, a(i(x_1))) \end{aligned}$$

Question: When is the translation of a TOP realizable by a $\mathrm{TOP}_{\mathrm{lin}}$?

When is the translation of a TOP realizable by a $\mathrm{TOP}_{\mathrm{lin}}$

To answer this question, we take a closer look at

- the canonical earliest normal form
- the zero output twinned property
- the lowest common ancestor conform property

When is the translation of a TOP realizable by a $\mathrm{TOP}_{\mathrm{lin}}$

For simplicity we only discuss total TOPs in the following.

Definition

A TOP T is called total if for all states q and all input symbols f of T, there is exactly one rule of T such that q and f appear together on the left-hand side.

Canonical Earliest Normal Form

The canonical earliest normal form is introduced by Engelfriet, Maneth and Seidl in 2009 as a generalization of a similar result for string transducers [Choffrut 1979]

Canonical Earliest Normal Form

The canonical earliest normal form is introduced by Engelfriet, Maneth and Seidl in 2009 as a generalization of a similar result for string transducers [Choffrut 1979]

Idea: Output is produced as 'early' as possible

Canonical Earliest Normal Form

The canonical earliest normal form is introduced by Engelfriet, Maneth and Seidl in 2009 as a generalization of a similar result for string transducers [Choffrut 1979]

Idea: Output is produced as 'early' as possible

$\Rightarrow$ Used to decide equivalence of TOPs

Canonical Earliest Normal Form

The canonical earliest normal form is introduced by Engelfriet, Maneth and Seidl in 2009 as a generalization of a similar result for string transducers [Choffrut 1979]

Idea: Output is produced as 'early' as possible

$\Rightarrow$ Used to decide equivalence of TOPs

A variant of the canonical earliest normal form is also used to decide $\mathrm{TOP}_{\mathrm{lin}}$ within $\mathrm{TOP}^{R}_{\mathrm{lin}}$

Canonical Earliest Normal Form Example

Idea: Output is produced as ‘early’ as possible

Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

Example: axiom: $q(x_1)$
rules: $q(a(x_1)) \to f(q(x_1), q(x_1))$ and $q(e) \to f(e, e)$

Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

Example: axiom: $q(x_1)$
rules: $q(a(x_1)) \to f(q(x_1), q(x_1))$ and $q(e) \to f(e, e)$

input tree

output tree

$q(x_1)$

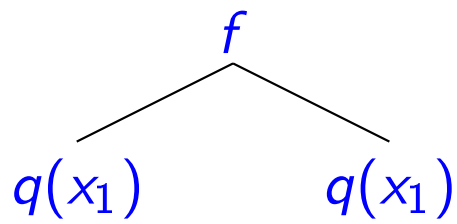
Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

Example: axiom: $q(x_1)$
rules: $q(a(x_1)) \to f(q(x_1), q(x_1))$ and $q(e) \to f(e, e)$

input tree

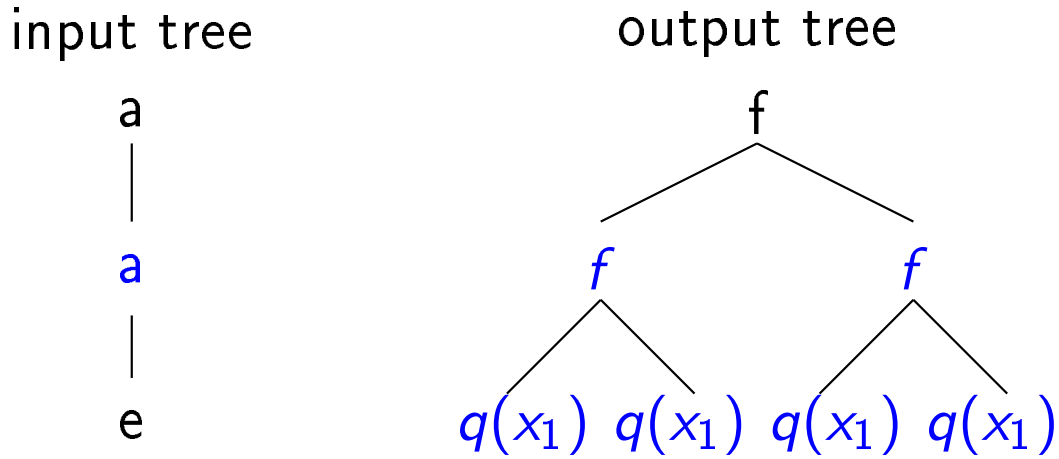
output tree



Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

Example: axiom: $q(x_1)$
rules: $q(a(x_1)) \to f(q(x_1), q(x_1))$ and $q(e) \to f(e, e)$



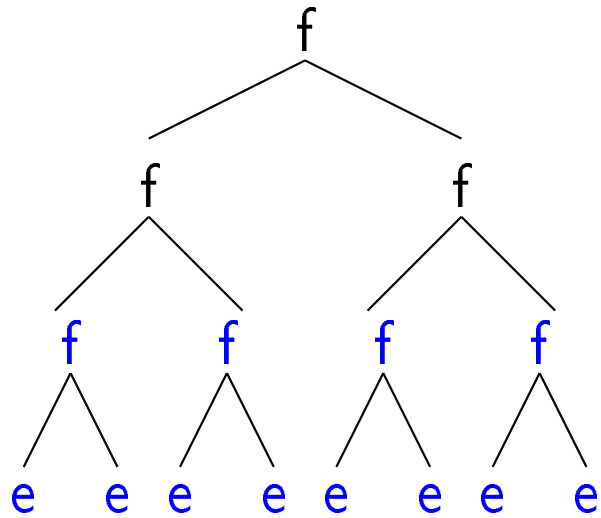
Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

Example: axiom: $q(x_1)$
rules: $q(a(x_1)) \to f(q(x_1), q(x_1))$ and $q(e) \to f(e, e)$

input tree

output tree



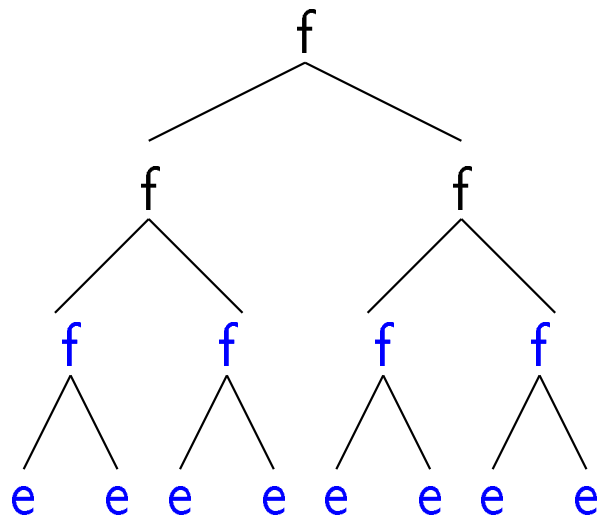
Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

Example: axiom: $q(x_1)$
rules: $q(a(x_1)) \to f(q(x_1), q(x_1))$ and $q(e) \to f(e, e)$

input tree

output tree



Not Earliest!

Canonical Earliest Normal Form Example

Idea: Output is produced as ‘early’ as possible

Canonical Earliest Normal Form Example

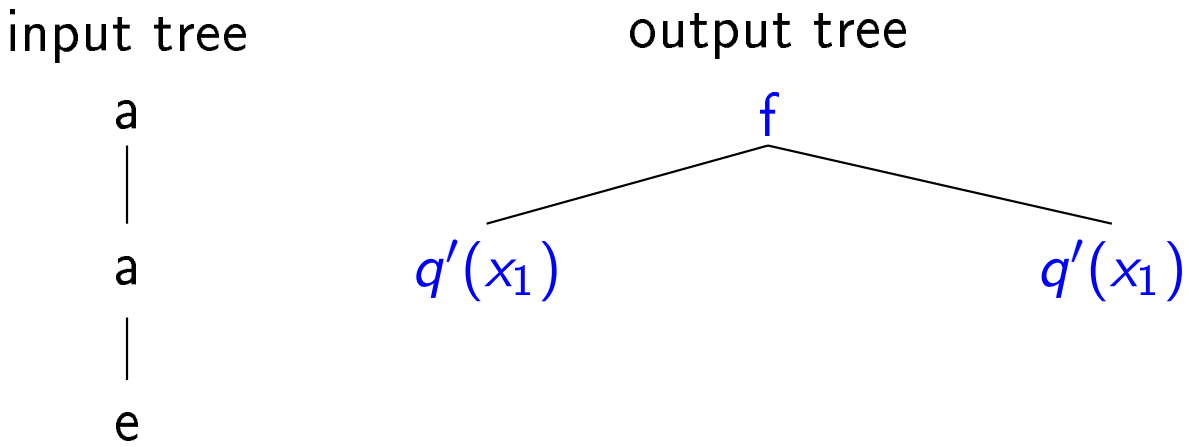
Idea: Output is produced as 'early' as possible

Example: axiom: $f(q'(x_1), q'(x_1))$
rules: $q'(a(x_1)) \to f(q'(x_1), q'(x_1))$ and $q'(e) \to e$

Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

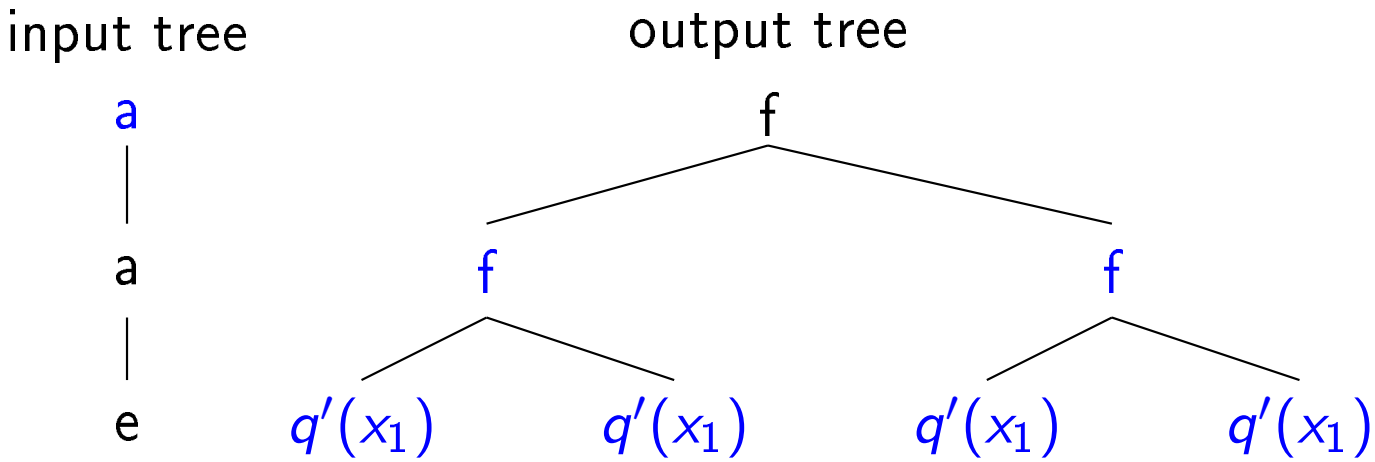
Example: axiom: $f(q'(x_1), q'(x_1))$
rules: $q'(a(x_1)) \to f(q'(x_1), q'(x_1))$ and $q'(e) \to e$



Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

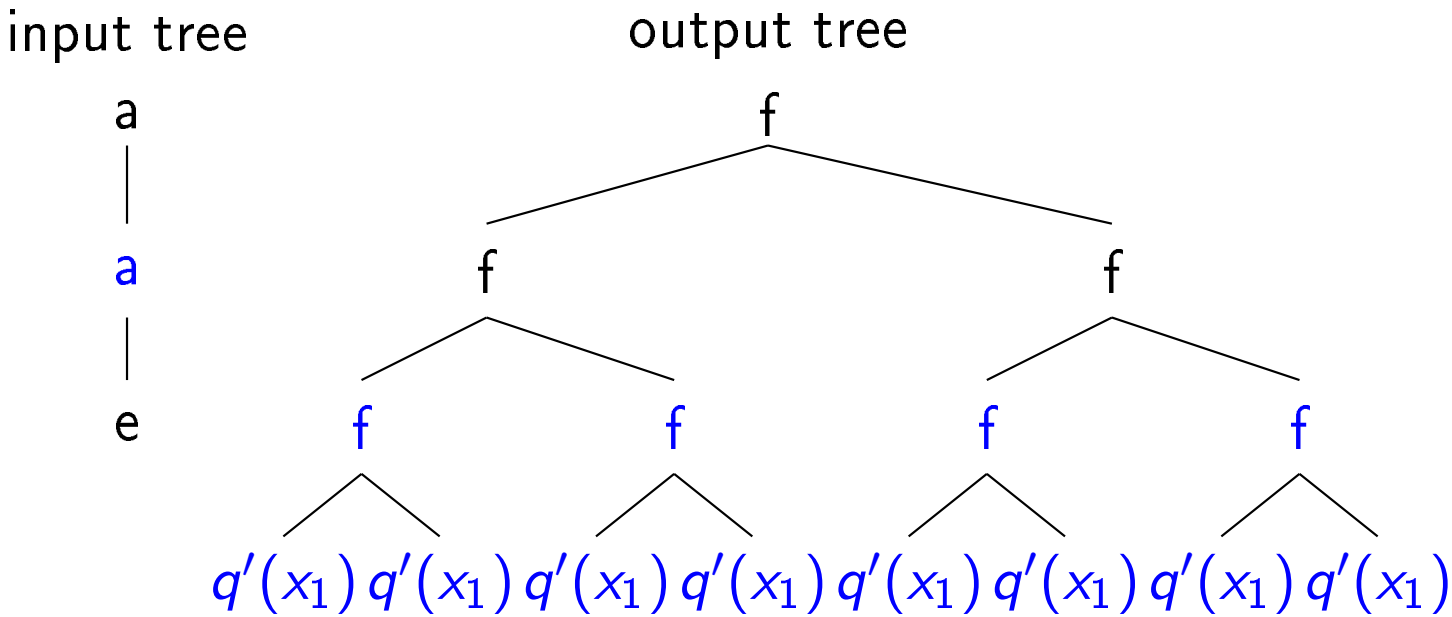
Example: axiom: $f(q'(x_1), q'(x_1))$
rules: $q'(a(x_1)) \to f(q'(x_1), q'(x_1))$ and $q'(e) \to e$



Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

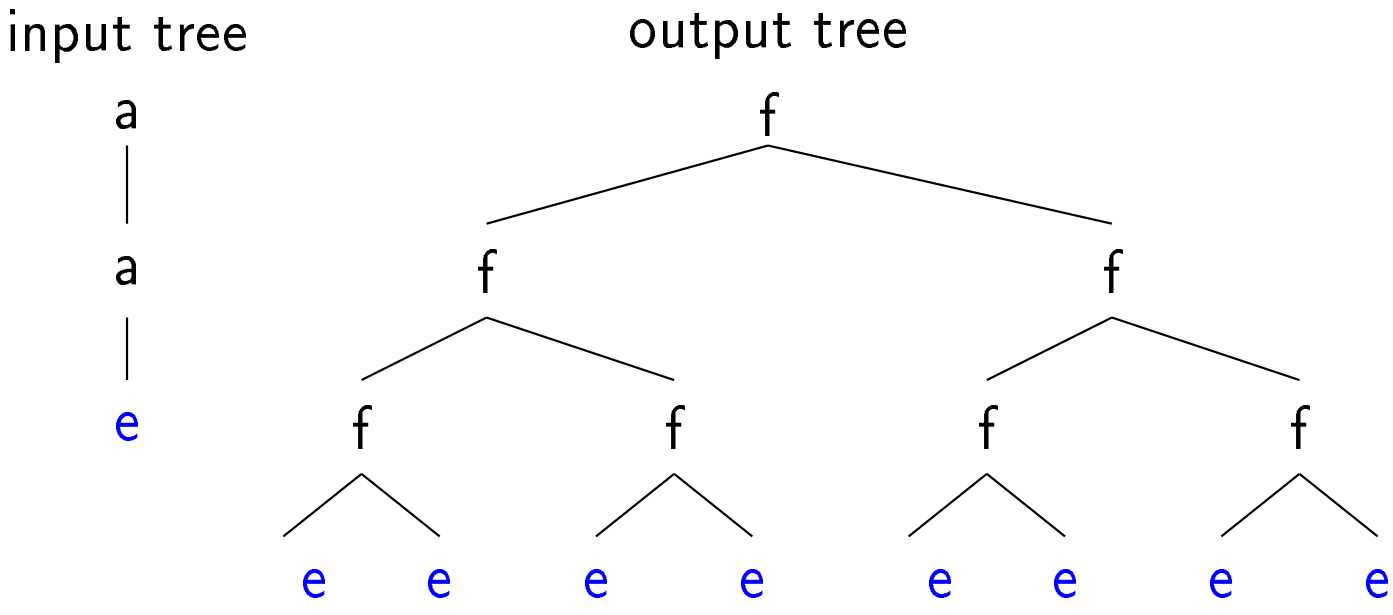
Example: axiom: $f(q'(x_1), q'(x_1))$
rules: $q'(a(x_1)) \to f(q'(x_1), q'(x_1))$ and $q'(e) \to e$



Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

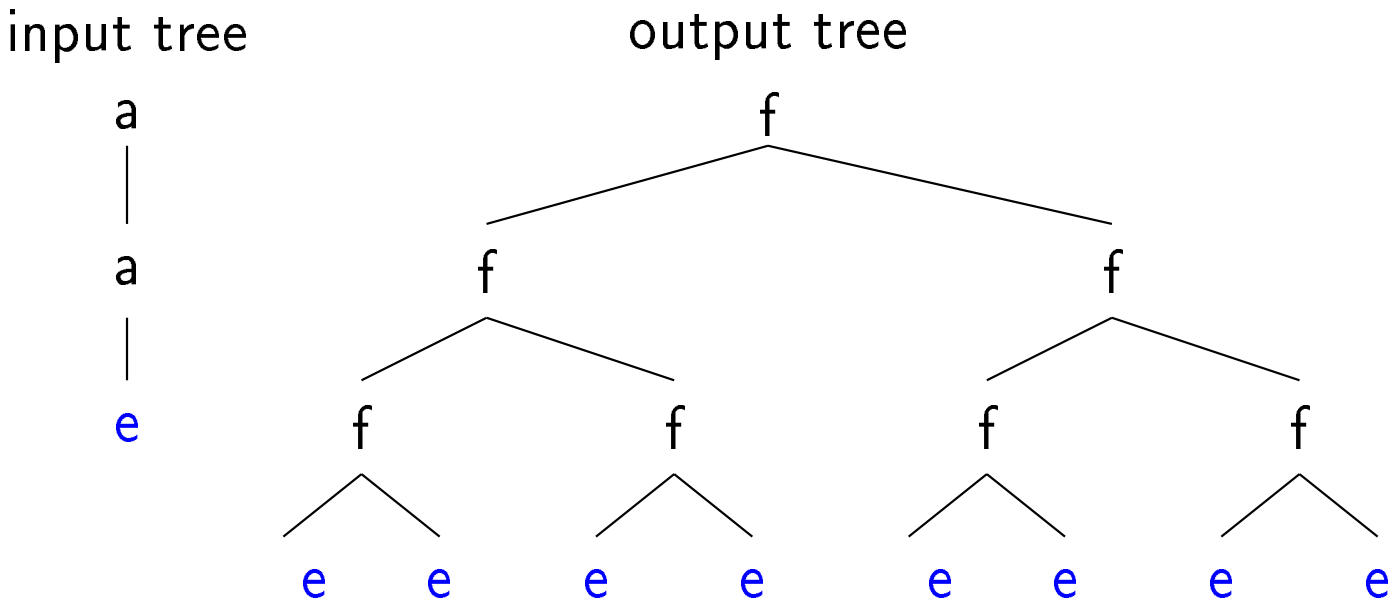
Example: axiom: $f(q'(x_1), q'(x_1))$
rules: $q'(a(x_1)) \to f(q'(x_1), q'(x_1))$ and $q'(e) \to e$



Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

Example: axiom: $f(q'(x_1), q'(x_1))$
rules: $q'(a(x_1)) \to f(q'(x_1), q'(x_1))$ and $q'(e) \to e$



This transducer is earliest!

Canonical Earliest Normal Form Example

Idea: Output is produced as 'early' as possible

Example: axiom: $f(q'(x_1), q'(x_1))$
rules: $q'(a(x_1)) \to f(q'(x_1), q'(x_1))$ and $q'(e) \to e$

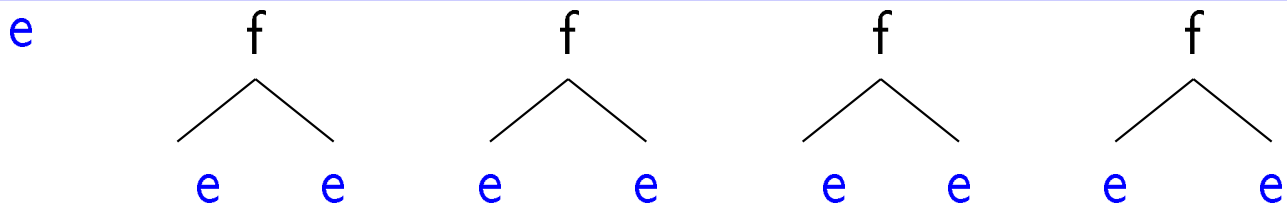
input tree

a

output tree

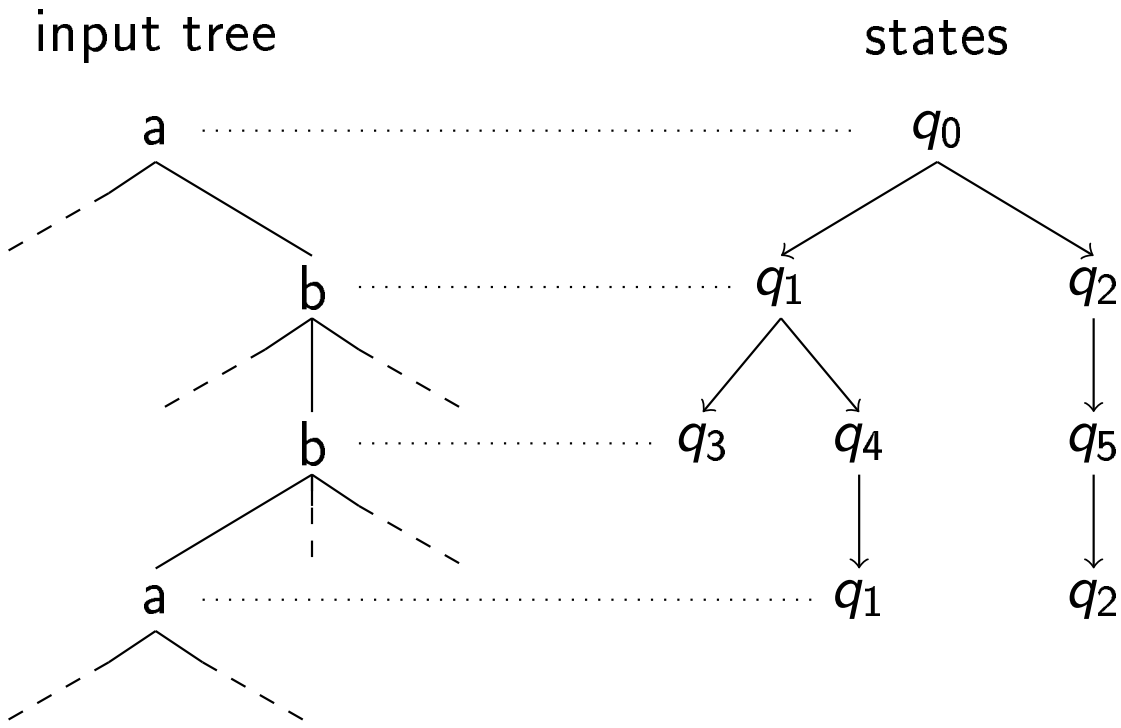
f

For every total transducer T a canonical earliest transducer T' can be constructed. Let T_1, T_2 be equivalent canonical transducers. Then T_1 and T_2 are the same (up to renaming of states).

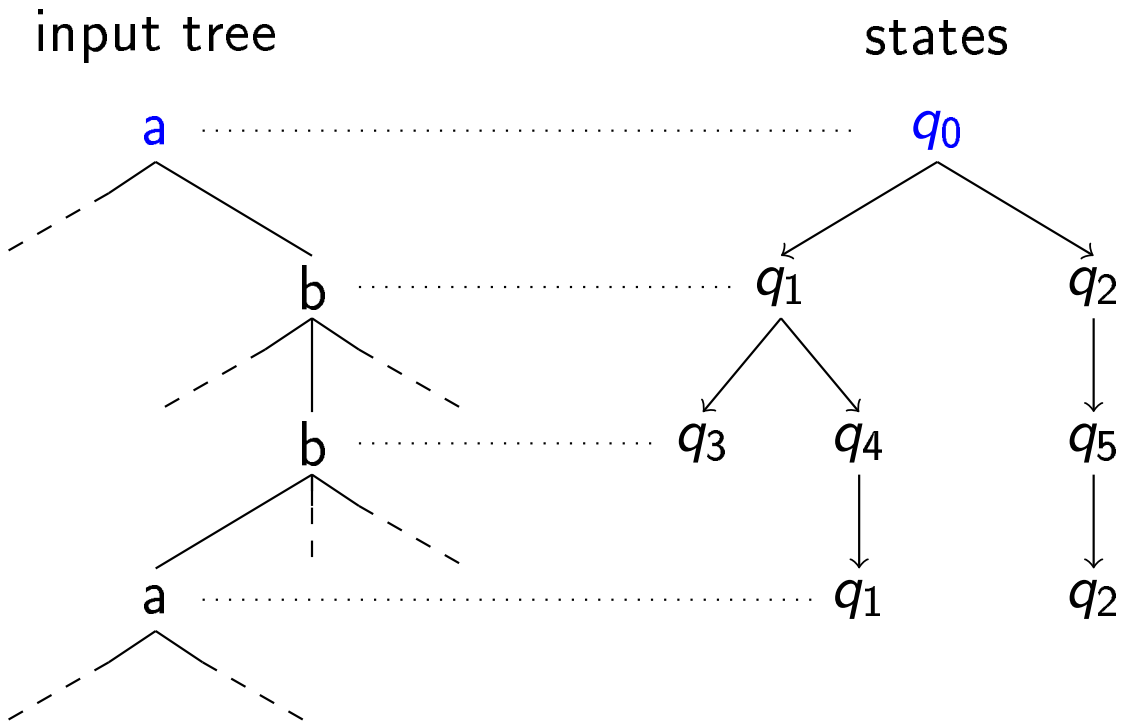


This transducer is earliest!

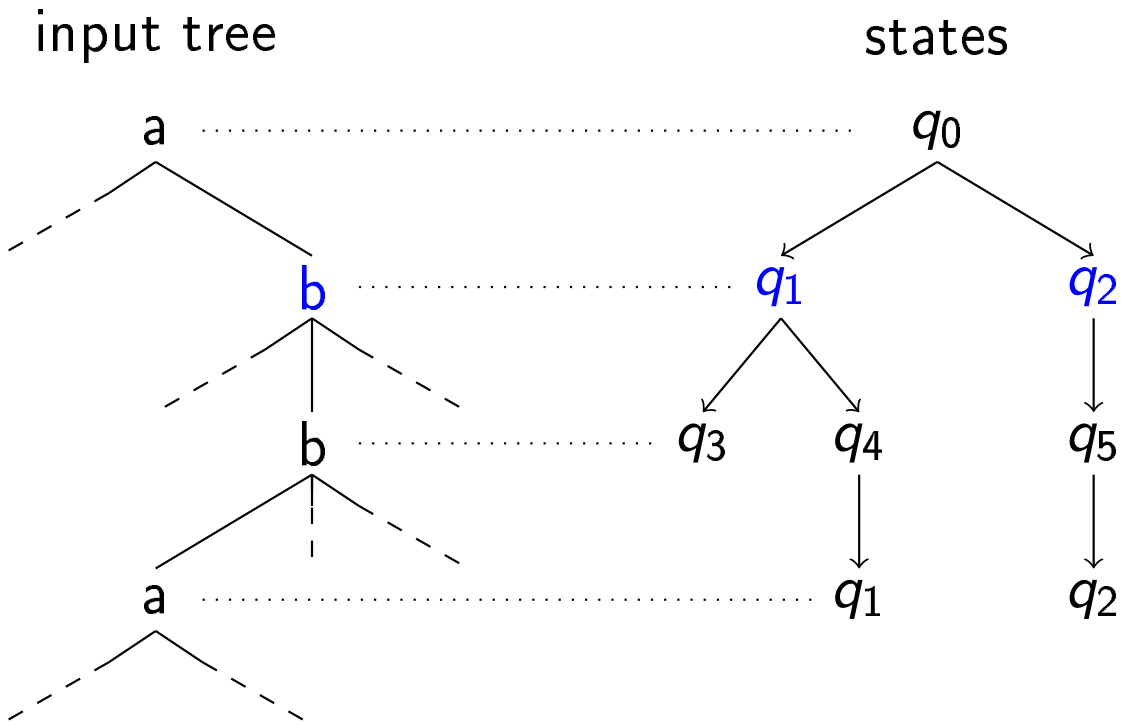
Zero Output Twinned



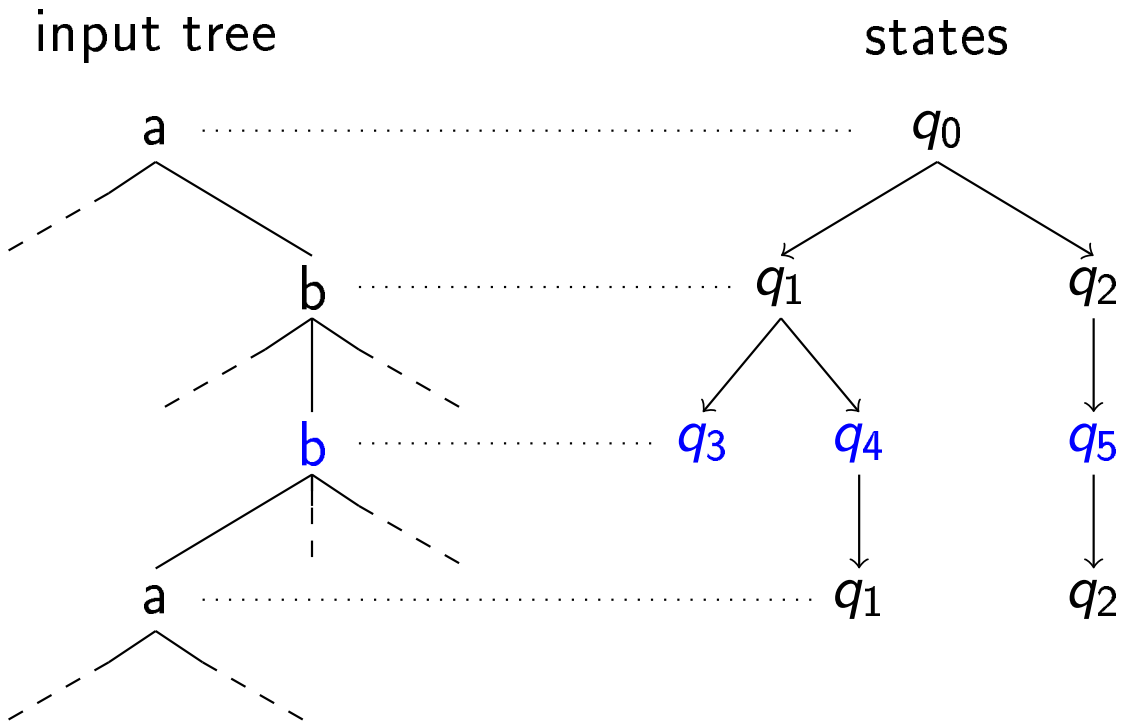
Zero Output Twinned



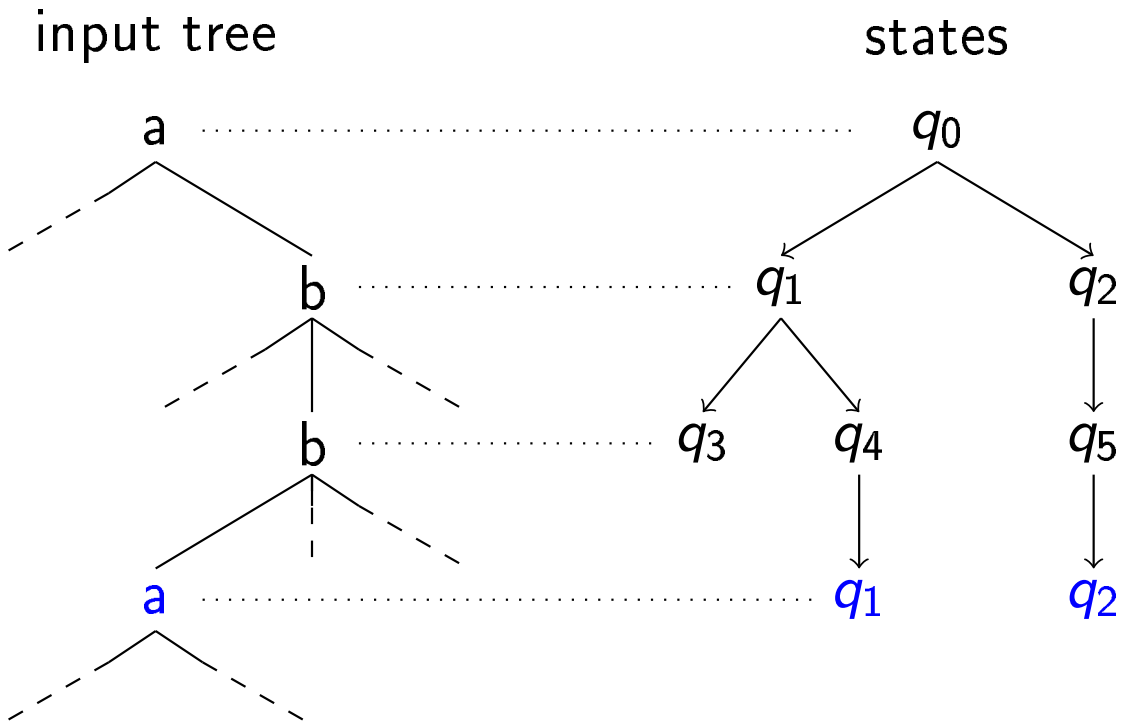
Zero Output Twinned



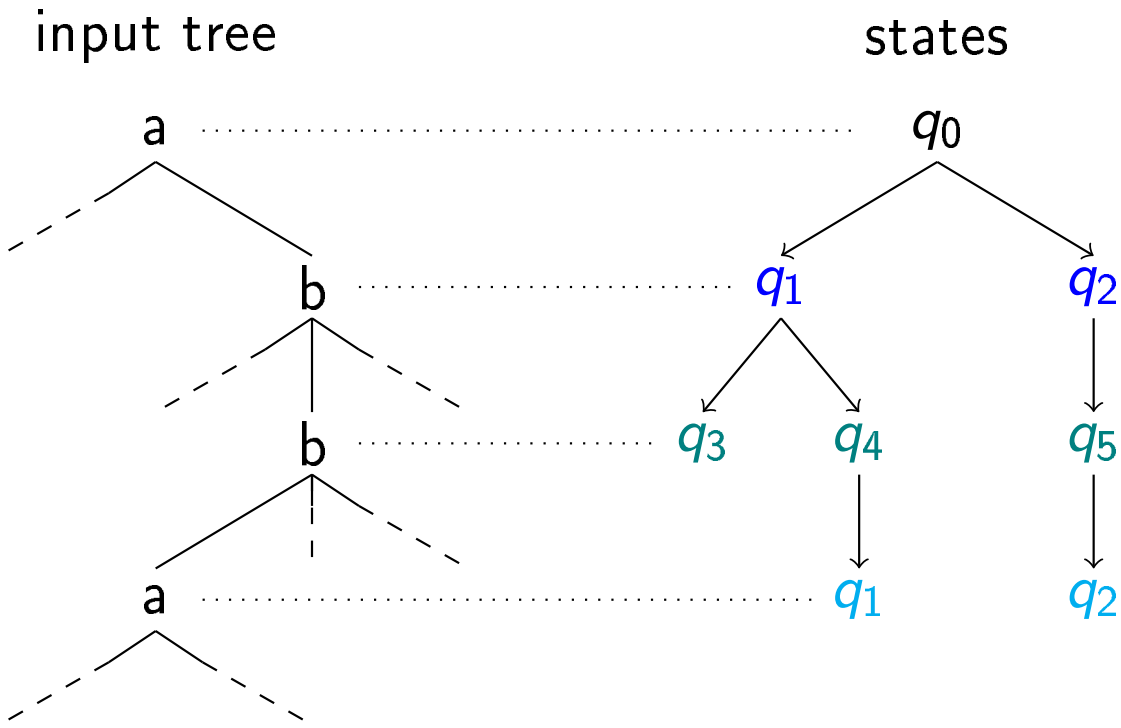
Zero Output Twinned



Zero Output Twinned

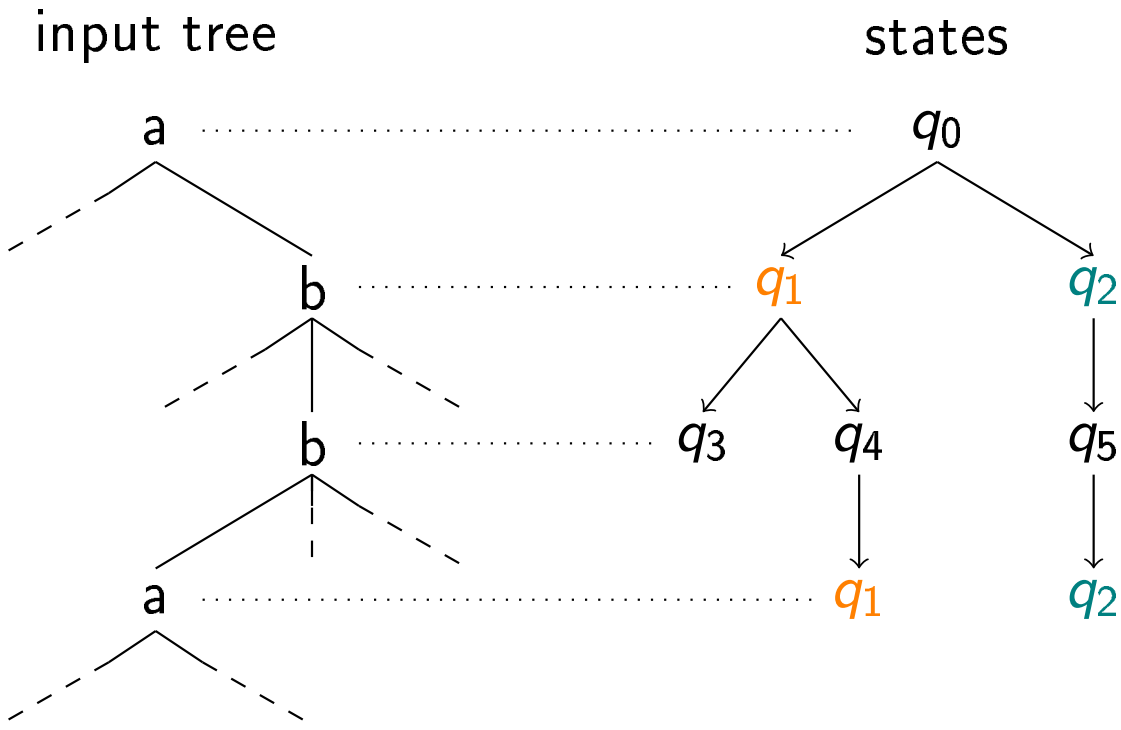


Zero Output Twinned



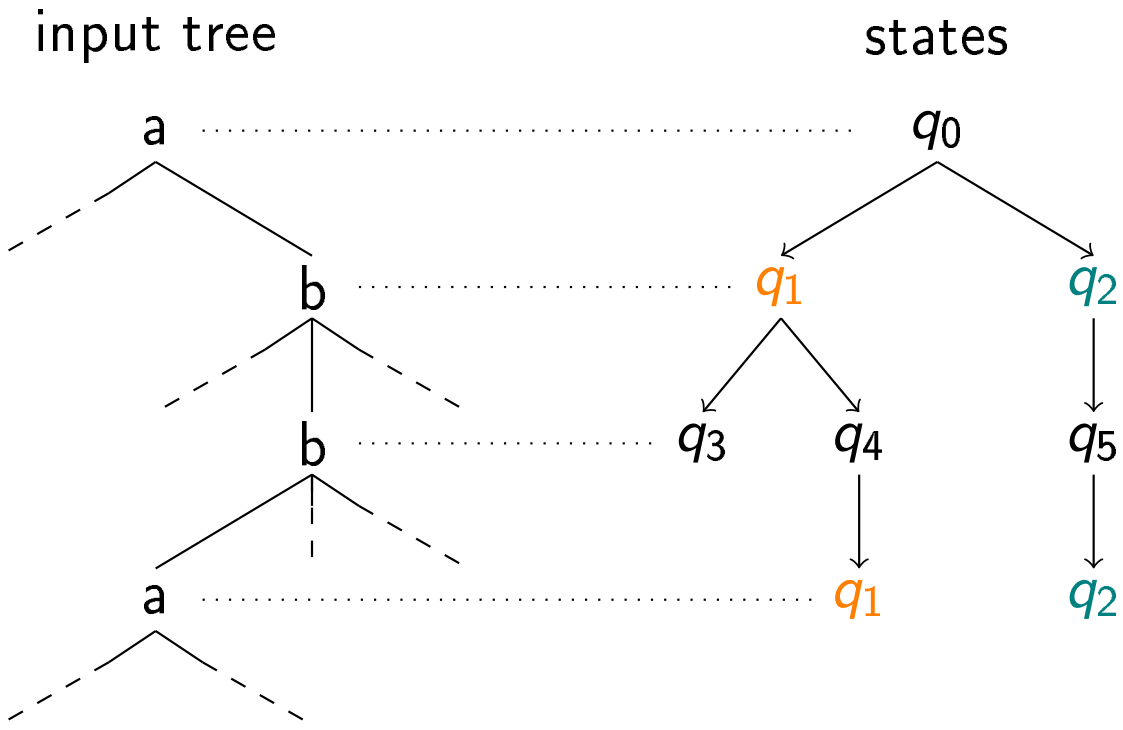
Pairwise Occuring States

Zero Output Twinned



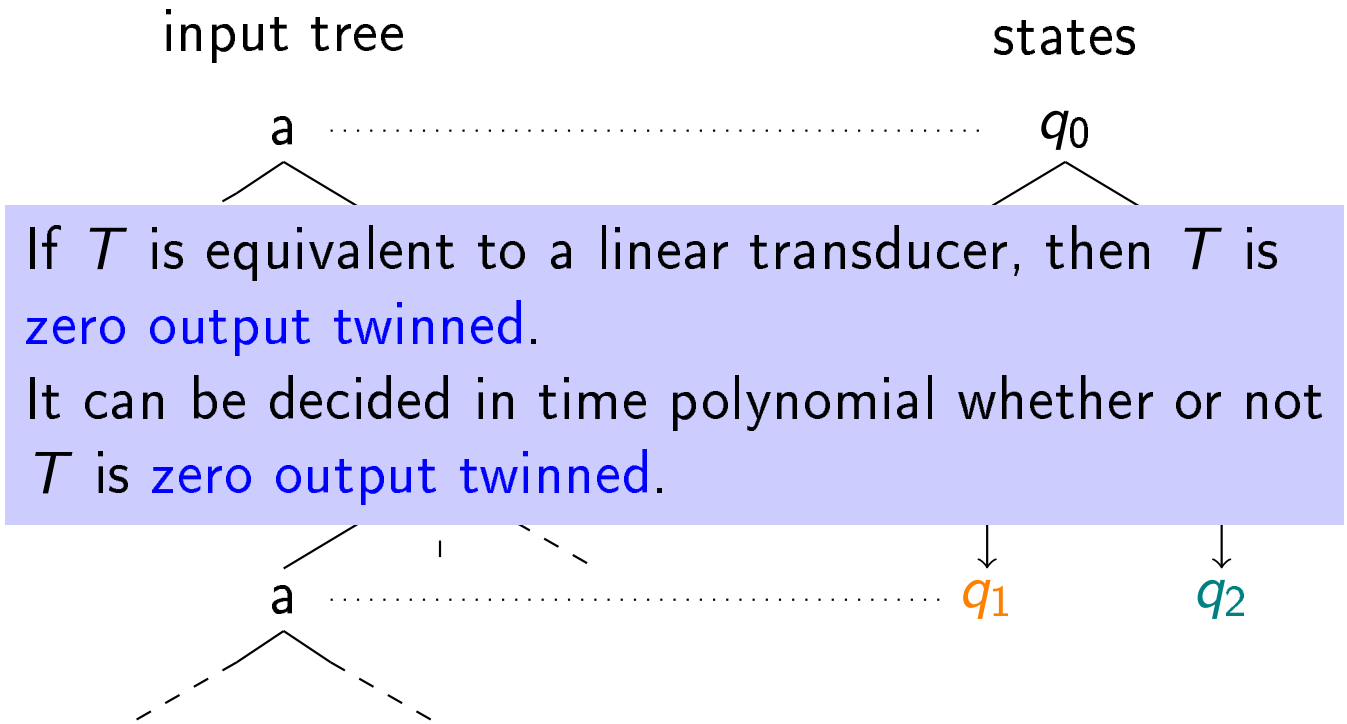
Pairwise Occuring State Loop

Zero Output Twinned



We call a TOP zero output twinned if
no pairwise occuring state loop produces output.

Zero Output Twinned

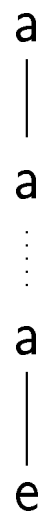


We call a TOP zero output twinned if
no pairwise occuring state loop produces output.

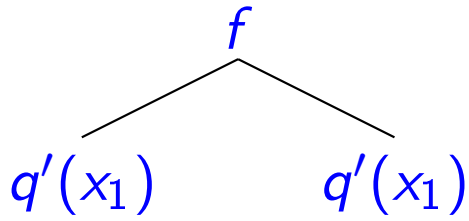
Why is Zero Output Twinned important?

Assume that a linear TOP equivalent to T exists.

input tree



output tree of T

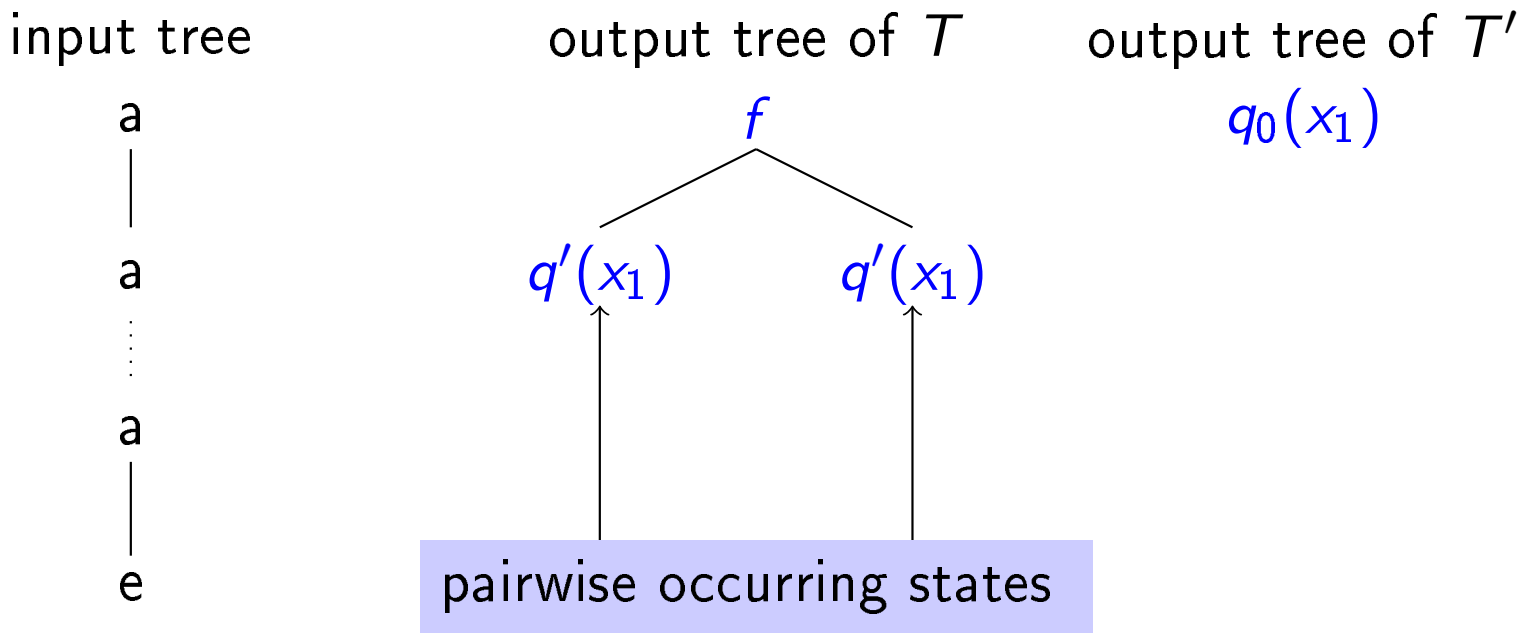


output tree of T'

$q_0(x_1)$

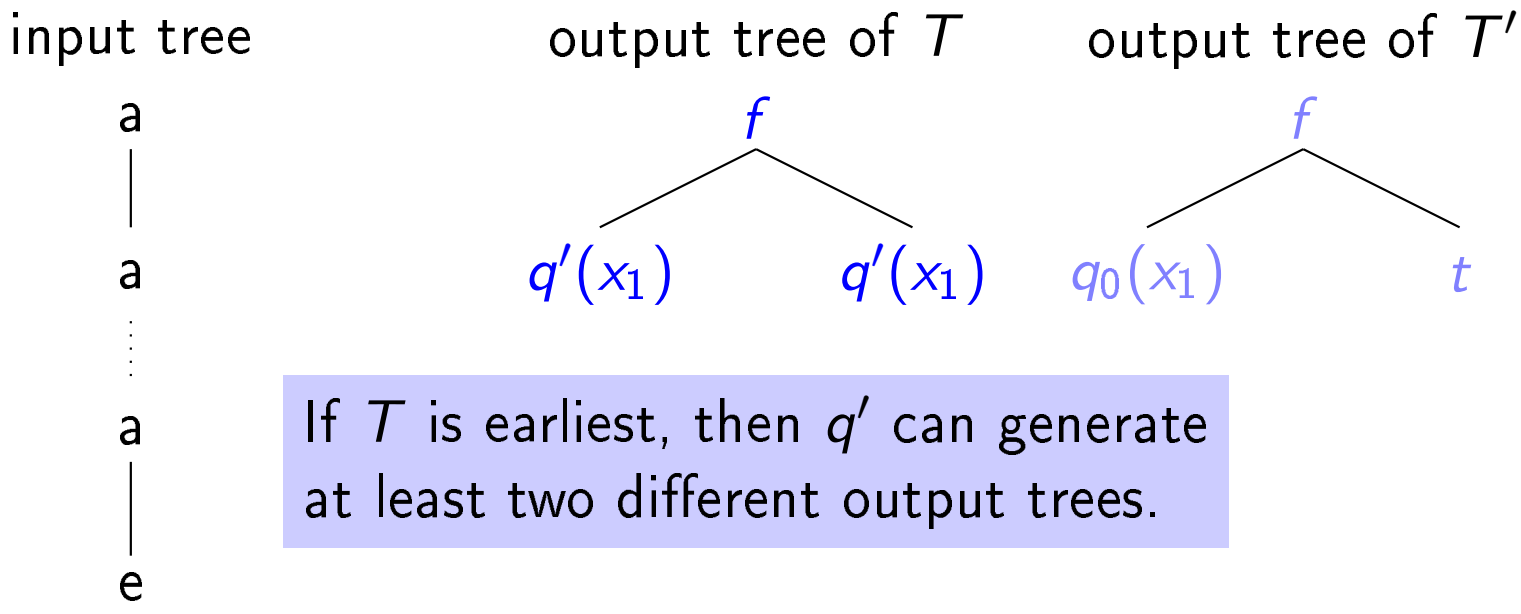
Why is Zero Output Twinned important?

Assume that a linear TOP equivalent to T exists.



Why is Zero Output Twinned important?

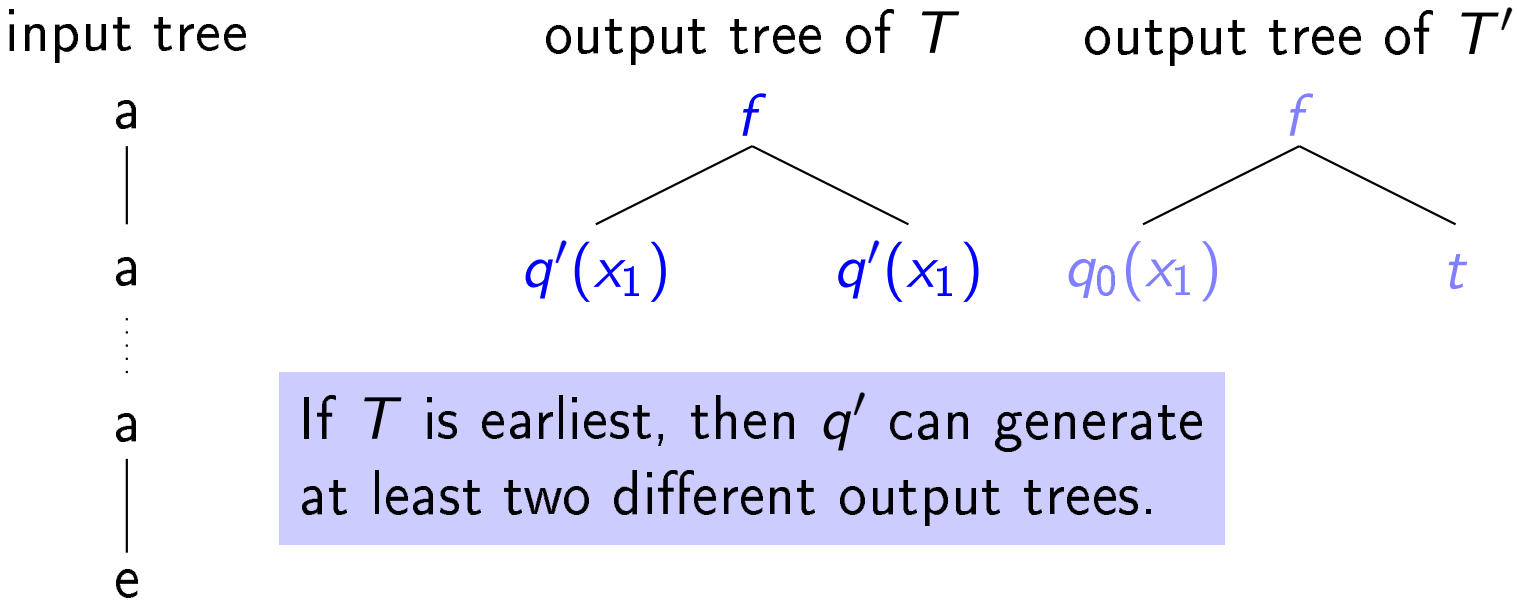
Assume that a linear TOP equivalent to T exists.



If T is earliest, then q' can generate at least two different output trees.

Why is Zero Output Twinned important?

Assume that a linear TOP equivalent to T exists.



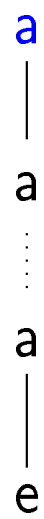
If T is earliest, then q' can generate at least two different output trees.

Contradiction to the earliest property of T!

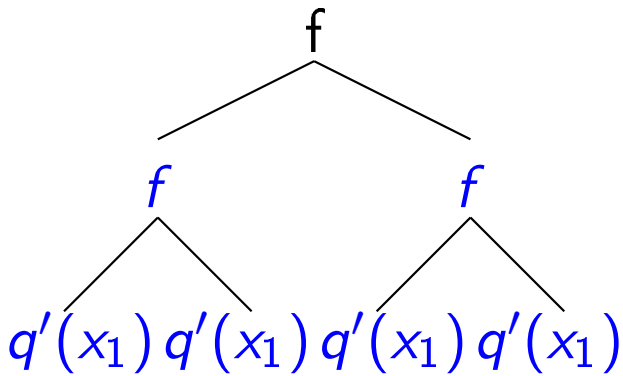
Why is Zero Output Twinned important?

Assume that a linear TOP equivalent to T exists.

input tree



output tree of T

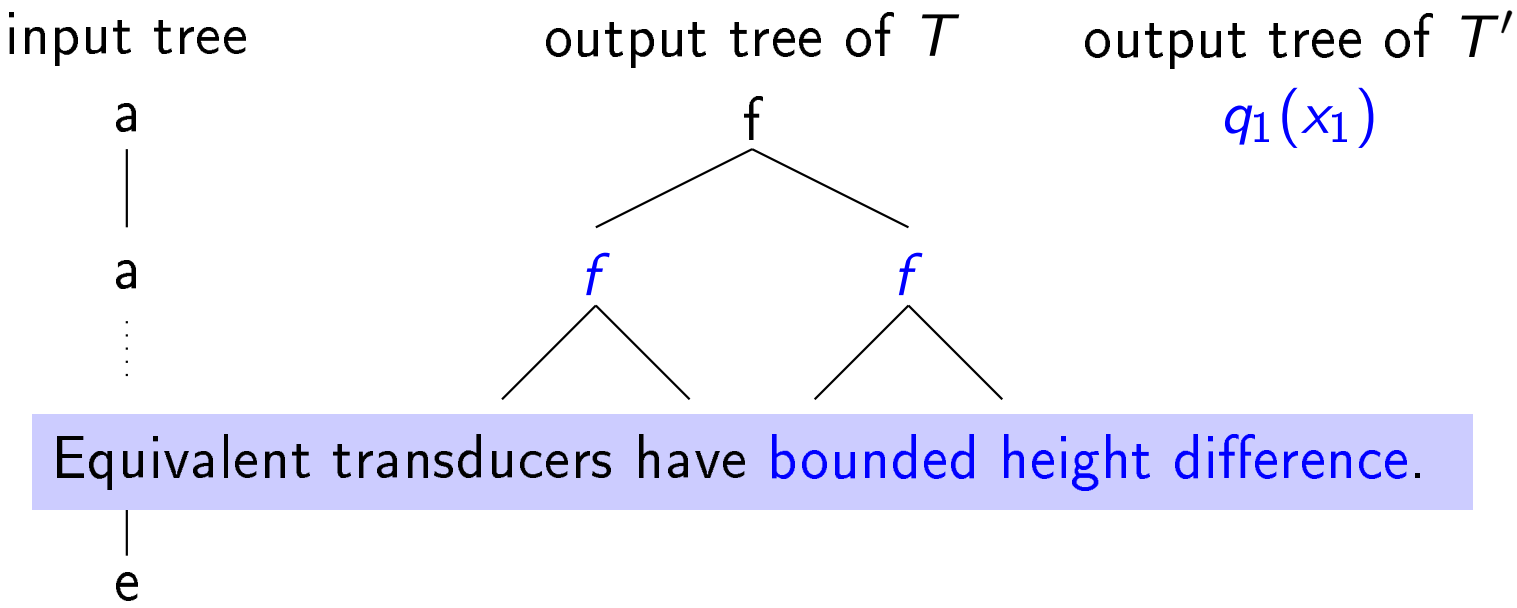


output tree of T'

$q_1(x_1)$

Why is Zero Output Twinned important?

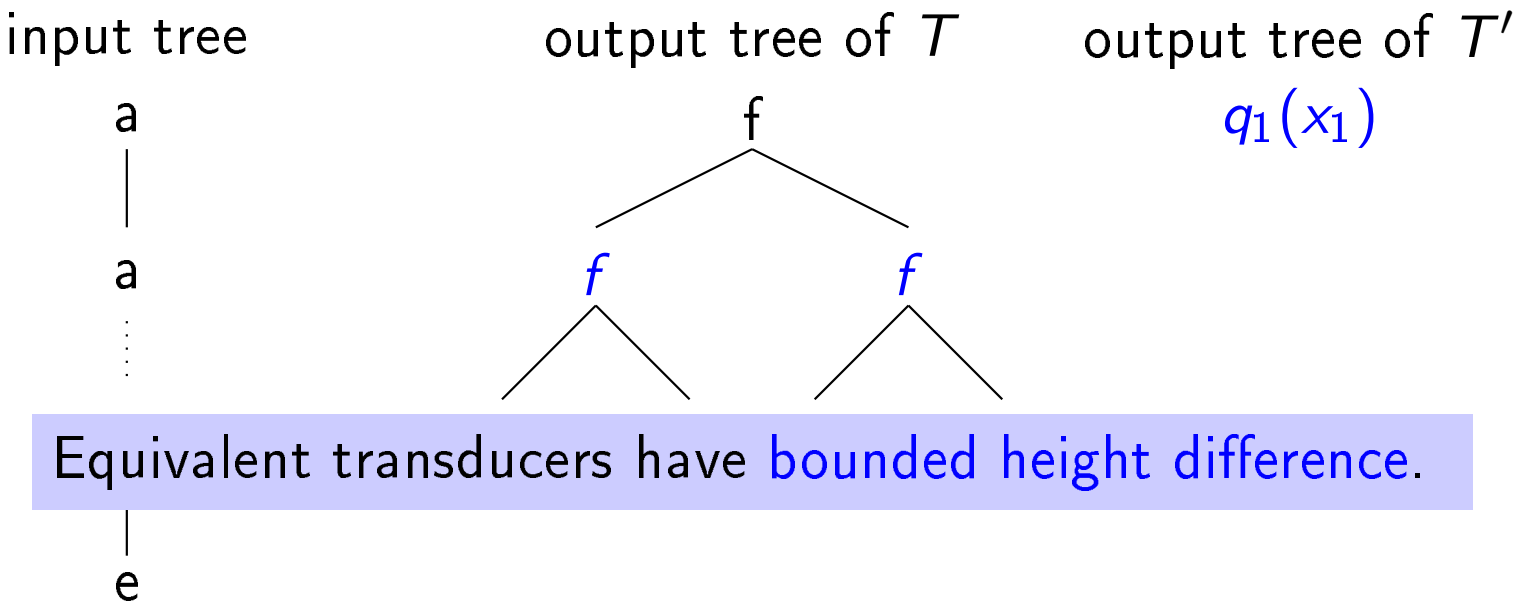
Assume that a linear TOP equivalent to T exists.



Equivalent transducers have bounded height difference.

Why is Zero Output Twinned important?

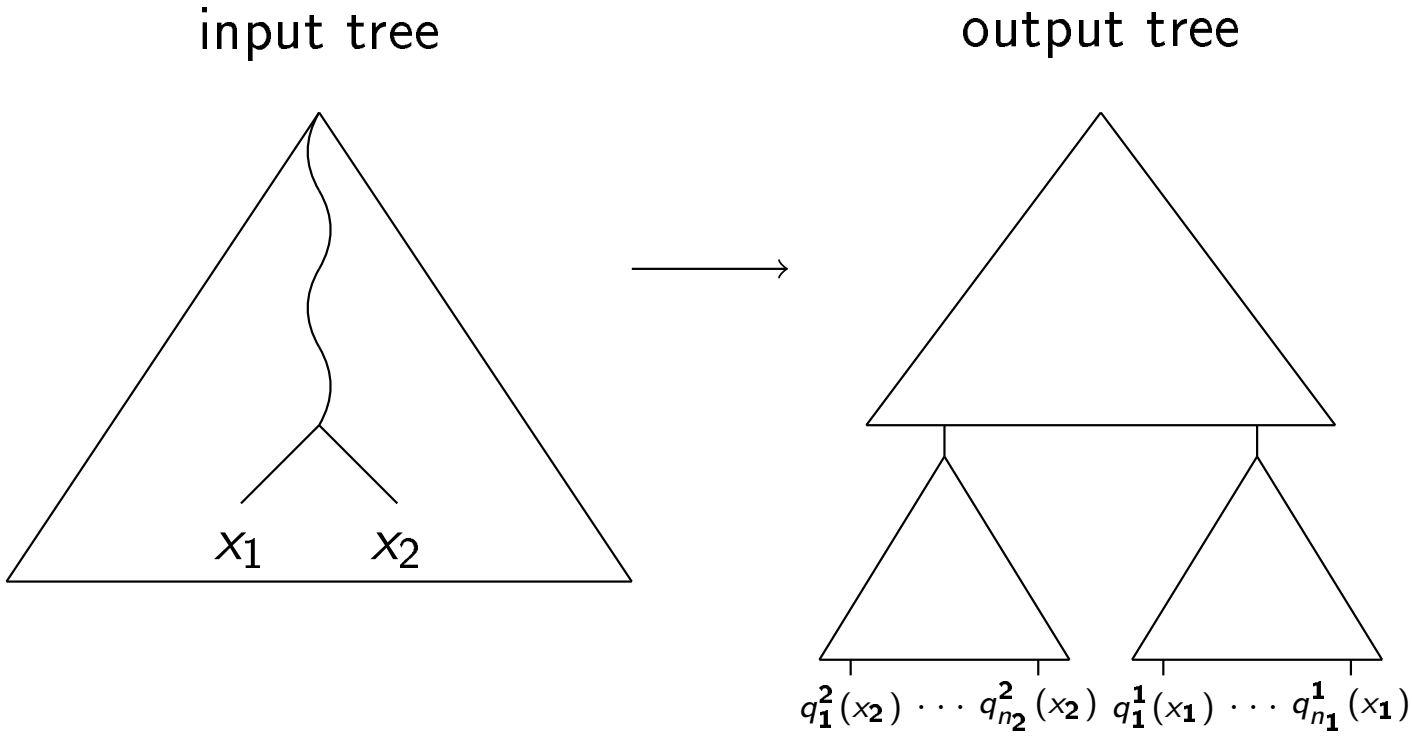
Assume that a linear TOP equivalent to T exists.



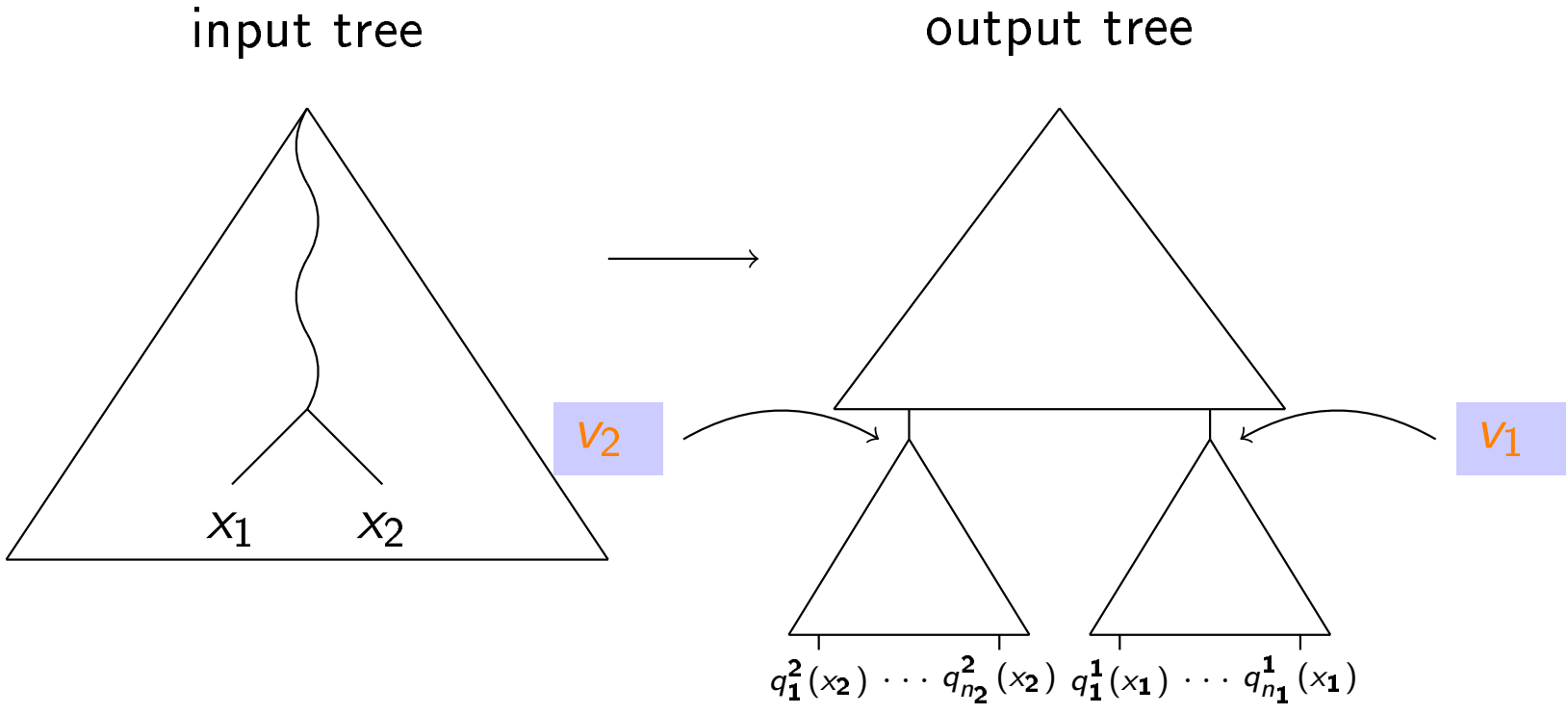
Equivalent transducers have bounded height difference.

T and T' cannot be equivalent.

Lowest Common Ancestor Conform

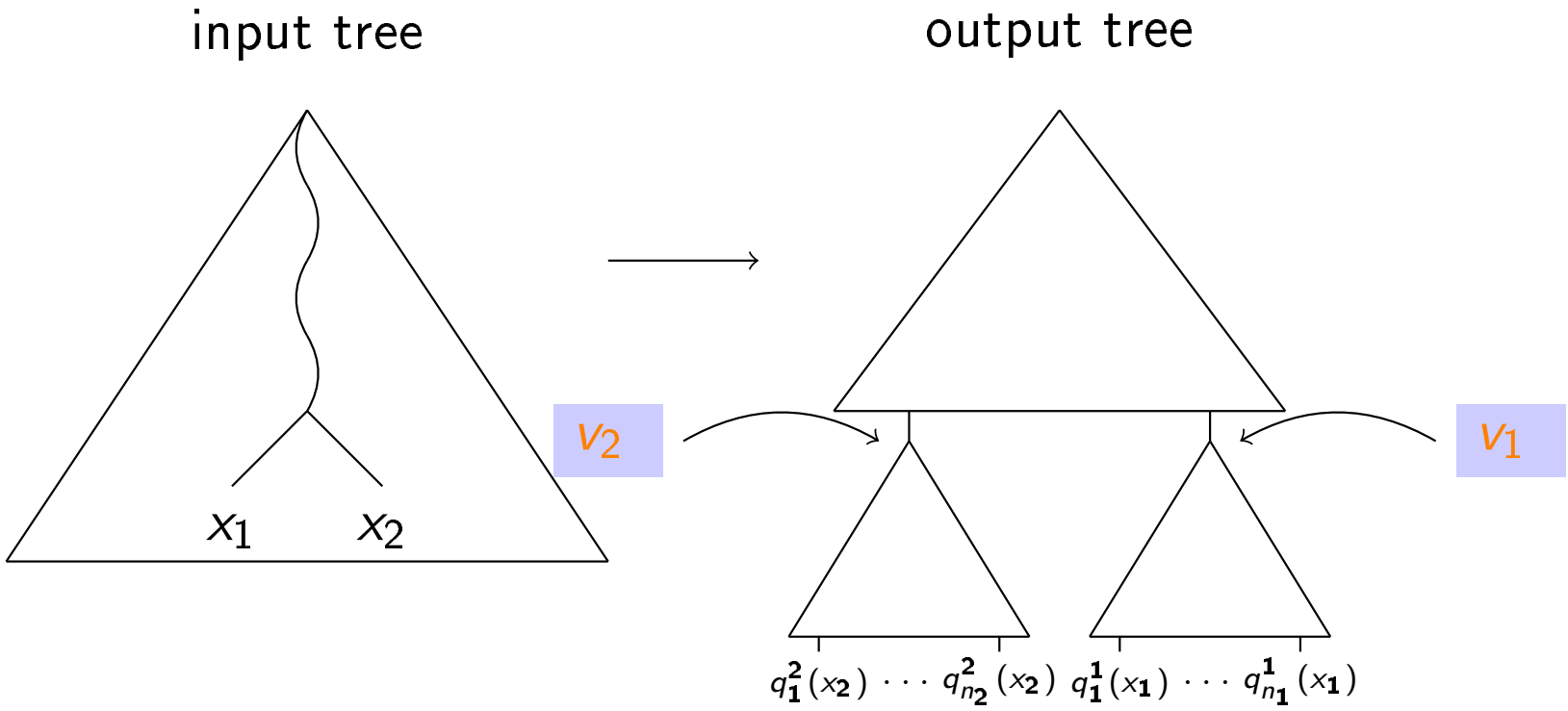


Lowest Common Ancestor Conform



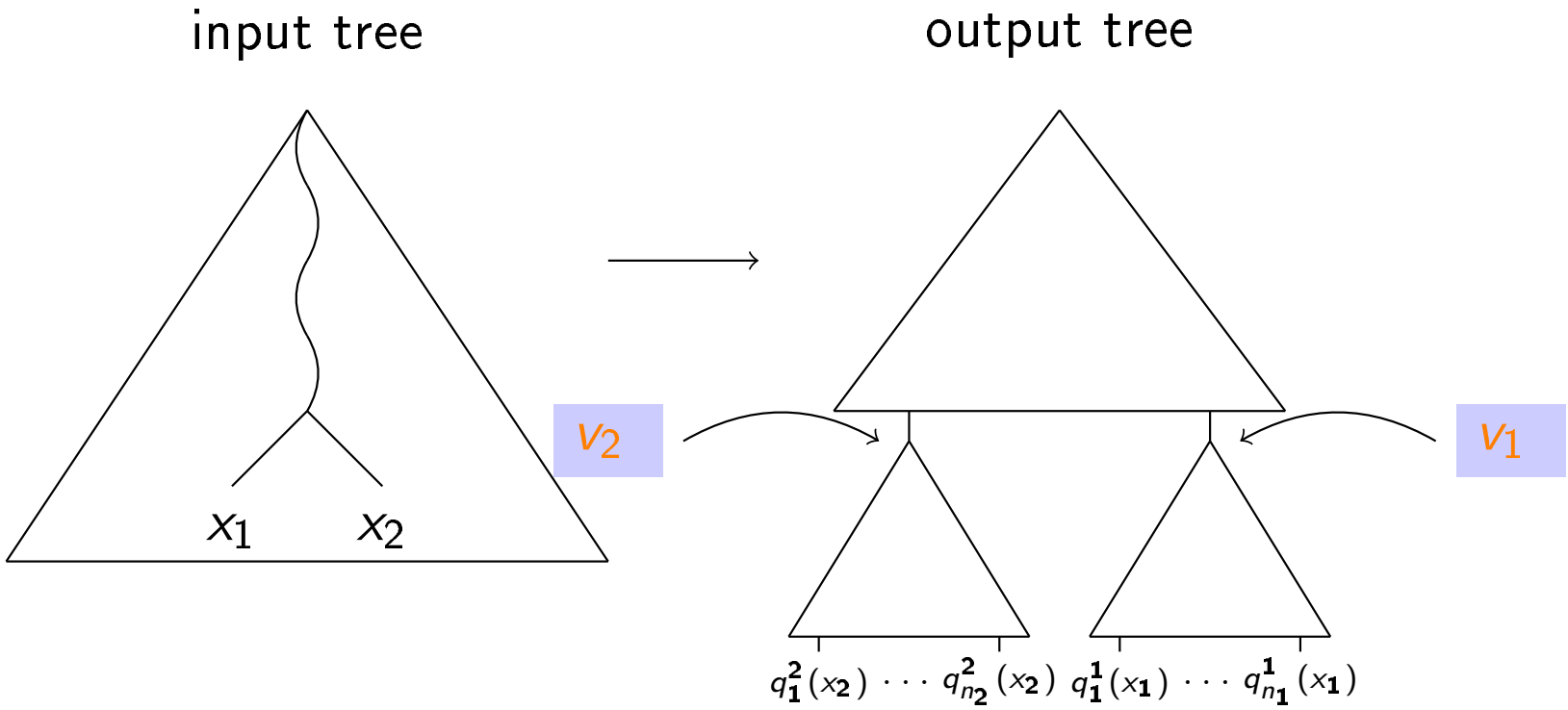
Lowest common ancestor of $q_1^1(x_1), \ldots, q_{n_1}^1(x_1)$ is v_1.
Lowest common ancestor of $q_1^2(x_2), \ldots, q_{n_2}^2(x_2)$is v_2.

Lowest Common Ancestor Conform



Subtree rooted at v_1 only contains x_1.
Subtree rooted at v_2 only contains x_2.
$\Rightarrow$ Lowest Common Ancestor Conform!

Lowest Common Ancestor Conform



If T is equivalent to a linear transducer, then T is lca-conform.
It can be decided in time polynomial in whether or not T is lca-conform.

Why is Lowest Common Ancestor Conformity important?

Example: axiom: $q_0(x_1)$

rules: $q_0(a(x_1, x_2)) \to g(q_1(x_1), q_2(x_2), q_3(x_1))$

$q_0(e) \to e_0$

$q_i(a(x_1, x_2)) \to e_0$ and

$q_i(e) \to e_i$ for $i = 1, 2, 3$

Why is Lowest Common Ancestor Conformity important?

Example: axiom: $q_0(x_1)$

rules: $q_0(a(x_1, x_2)) \to g(q_1(x_1), q_2(x_2), q_3(x_1))$

$q_0(e) \to e_0$

$q_i(a(x_1, x_2)) \to e_0$ and

$q_i(e) \to e_i$ for $i = 1, 2, 3$

Transducer is earliest and zero output twinned.

But no equivalent linear transducer exists!

Why is Lowest Common Ancestor Conformity important?

Example: axiom: $q_0(x_1)$

rules: $q_0(a(x_1, x_2)) \to g(q_1(x_1), q_2(x_2), q_3(x_1))$

$q_0(e) \to e_0$

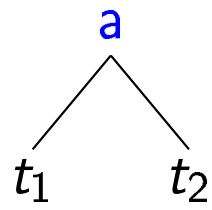
$q_i(a(x_1, x_2)) \to e_0$ and

$q_i(e) \to e_i$ for $i = 1, 2, 3$

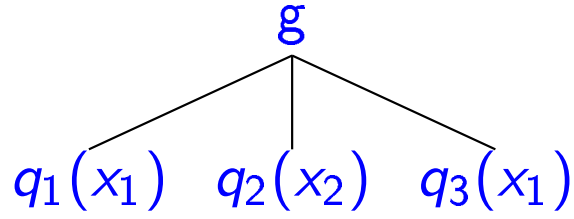
Transducer is earliest and zero output twinned.

But no equivalent linear transducer exists!

input tree



output tree of T



output tree of T'

$q'(x_1)$

(or $q'(x_2)$)

Why is Lowest Common Ancestor Conformity important?

Example: axiom: $q_0(x_1)$

rules: $q_0(a(x_1, x_2)) \to g(q_1(x_1), q_2(x_2), q_3(x_1))$

$q_0(e) \to e_0$

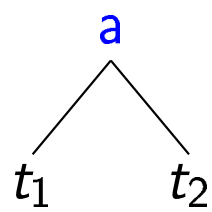
$q_i(a(x_1, x_2)) \to e_0$ and

$q_i(e) \to e_i$ for $i = 1, 2, 3$

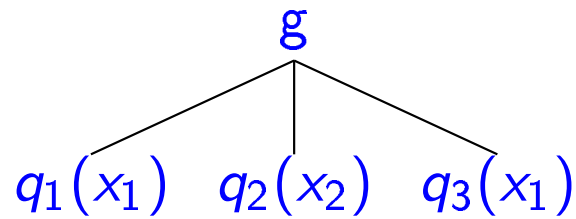
Transducer is earliest and zero output twinned.

But no equivalent linear transducer exists!

input tree



output tree of T



output tree of T'

$q'(x_1)$

(or $q'(x_2)$)

If T and T' were equivalent, then q_2 can generate at least two different output trees.

Contradiction to equivalence!

Main Result

The following holds.

Theorem

Let T be a canonical earliest transducer. An equivalent linear transducer T' exists if and only if T is zero output twinned and lowest ancestor conform.

Main Result

The following holds.

Theorem

Let T be a canonical earliest transducer. An equivalent linear transducer T' exists if and only if T is zero output twinned and lowest ancestor conform.

Theorem

Let T be a total TOP. It is decidable in polynomial time whether or not an equivalent linear transducer T' exists, and if so T' can be effectively constructed.

Main Result

Our results can be extended to partial TOPs:

Theorem

Let T be an arbitrary TOP. It is decidable in polynomial time whether or not an equivalent linear transducer T' exists, and if so T' can be effectively constructed.

Main Result

Our results can be extended to partial TOPs:

Theorem

Let T be an arbitrary TOP. It is decidable in polynomial time whether or not an equivalent linear transducer T' exists, and if so T' can be effectively constructed.

In fact our results even hold for arbitrary transducer with regular look-ahead (TOP^R).

Theorem

Let M be an arbitrary TOP^R. It is decidable in polynomial time whether or not an equivalent linear transducer T' exists, and if so T' can be effectively constructed.

Open Questions and future Research

- Is $\mathrm{TOP}_{\mathrm{lin}}$ decidable within a larger class than TOP^R?
- Do similar results hold for other classes?
  For instance, is $\mathrm{TOP}_{\mathrm{uc}}$ decidable within TOP^R?

Thank You for your Attention!