How to decide Functionality of Compositions of Top-Down Tree Transducers

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

Universität Bremen, Germany
{maneth,martin.vu}@uni-bremen.de

TU München, Germany
seidl@in.tum.de

October 26, 2022

What are Tree Transducers?

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

What are Tree Transducers?

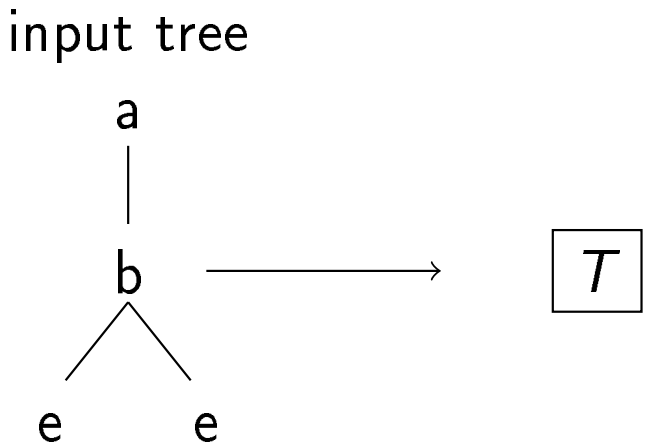
Top-Down Tree Transducers were invented by Rounds and Thatcher in the early 1970's as a formal model for compiler theory and linguistics.

Top-Down Tree Transducers realize *tree translations*:

What are Tree Transducers?

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

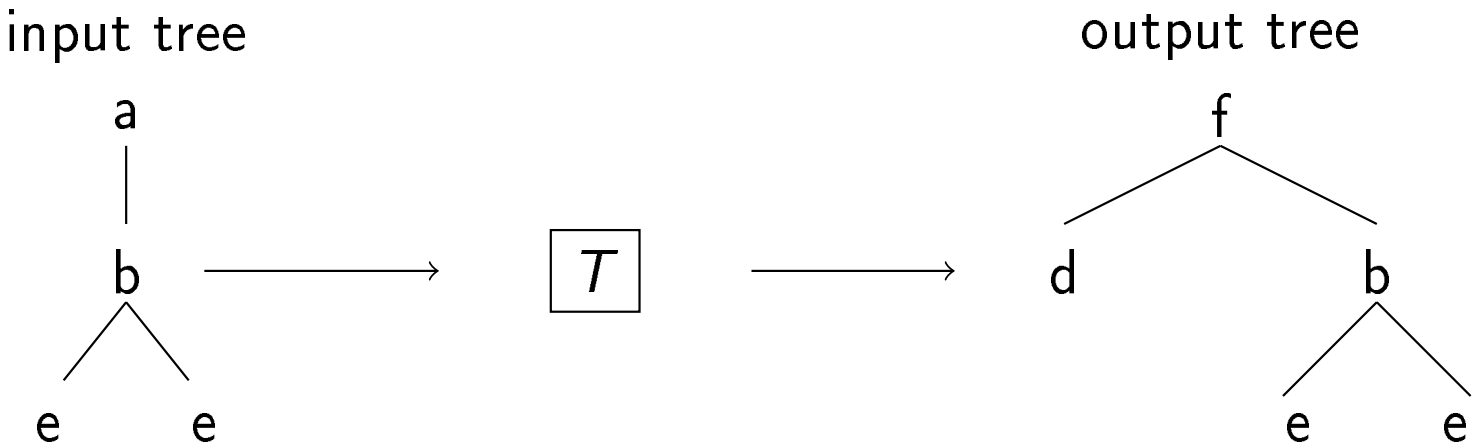
Top-Down Tree Transducers realize *tree translations*:



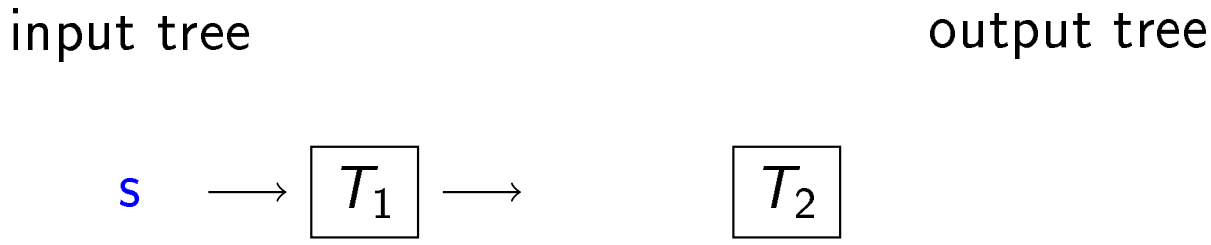
What are Tree Transducers?

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

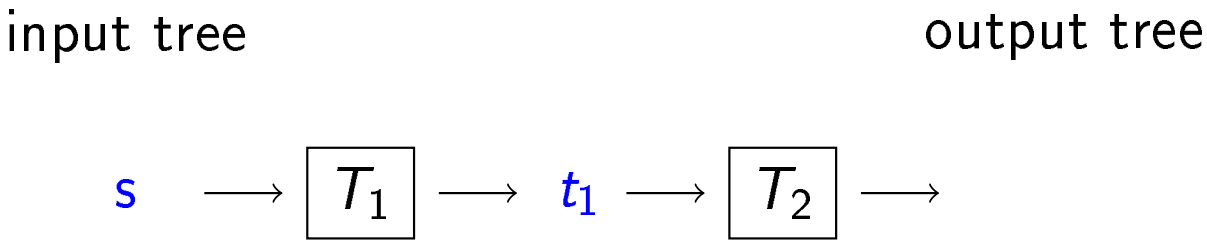
Top-Down Tree Transducers realize *tree translations*:



Composition of Transducers

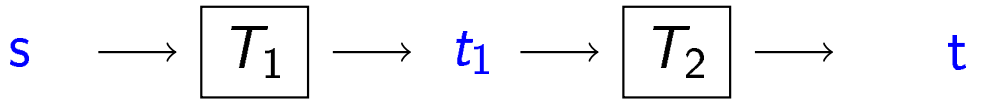


Composition of Transducers



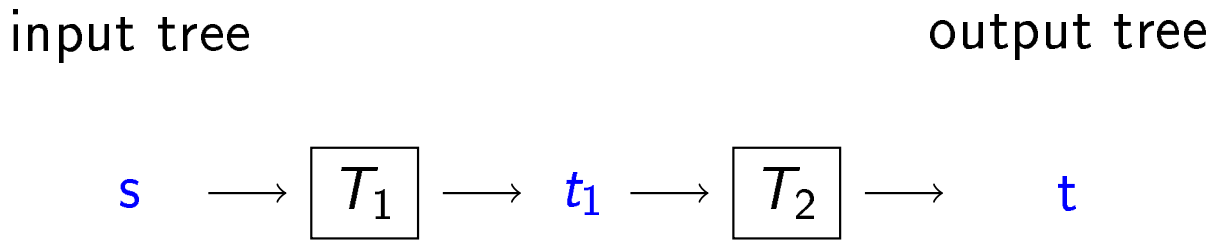
Composition of Transducers

input tree output tree



This is not ideal...

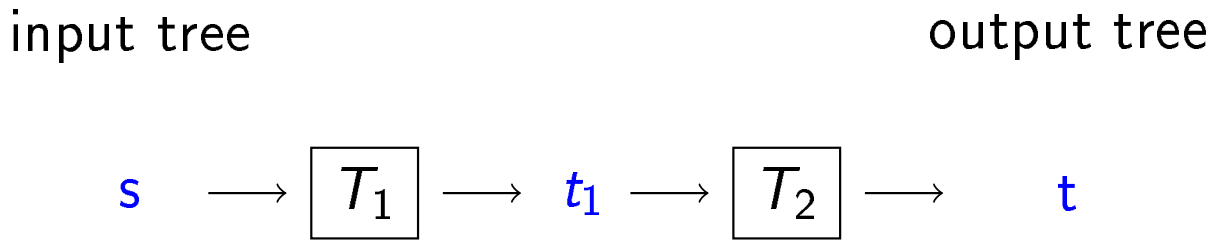
Composition of Transducers



This is not ideal...

Question: Does a deterministic transducer M (with look-ahead) that realizes the same translation exist?

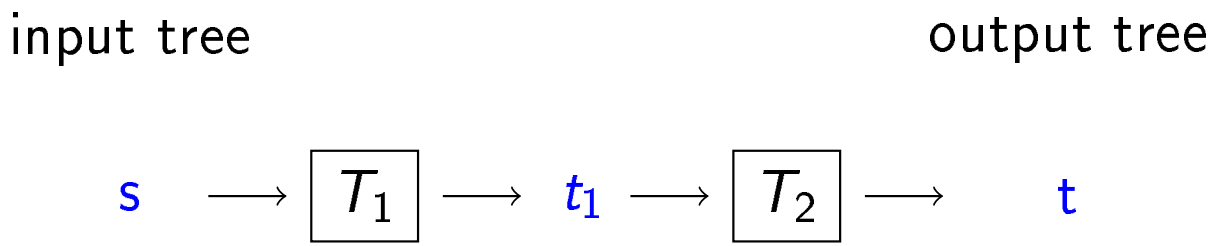
Composition of Transducers



This is not ideal...

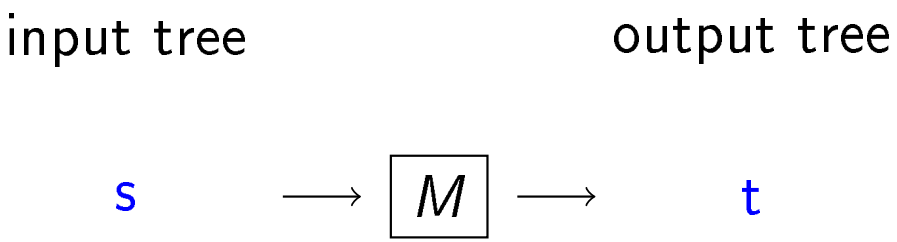
Question: Does a deterministic transducer M (with look-ahead) that realizes the same translation exist?

Composition of Transducers



This is not ideal...

Question: Does a deterministic transducer M (with look-ahead) that realizes the same translation exist?



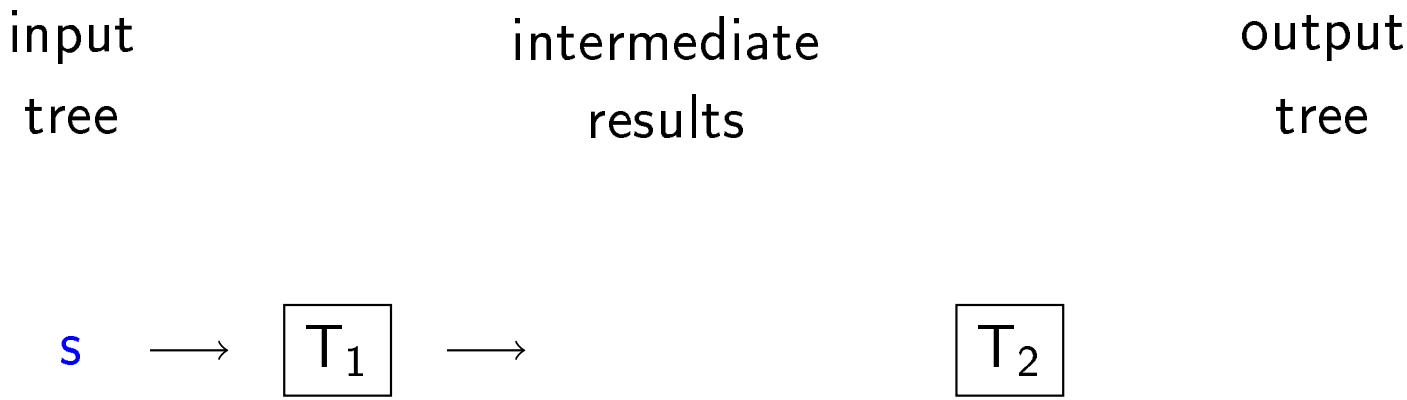
Main Idea

Observation: If the composition of T_1 and T_2 is not functional then M does not exist.

Main Idea

Observation: If the composition of T_1 and T_2 is not functional then M does not exist.

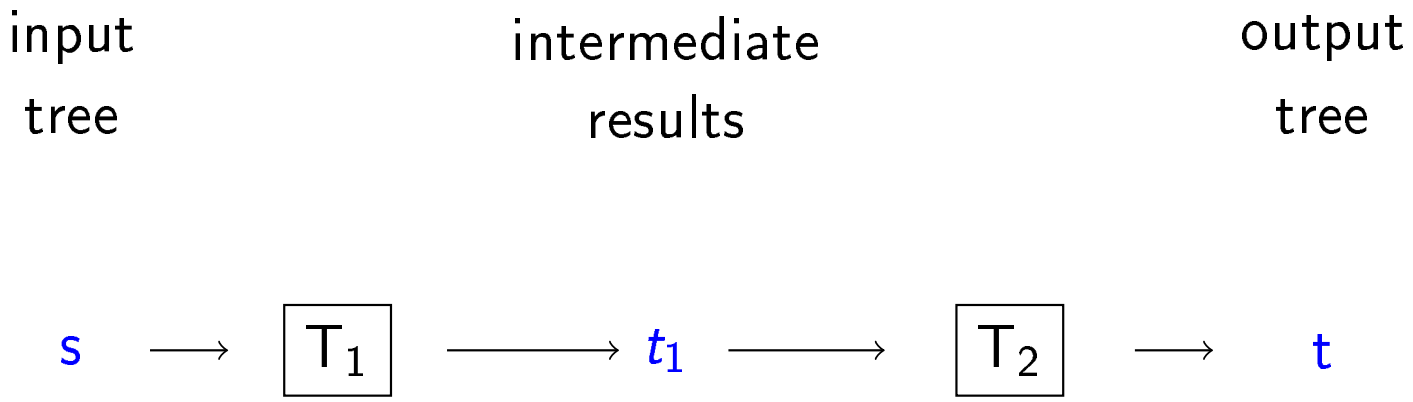
What does functional mean?



Main Idea

Observation: If the composition of T_1 and T_2 is not functional then M does not exist.

What does functional mean?

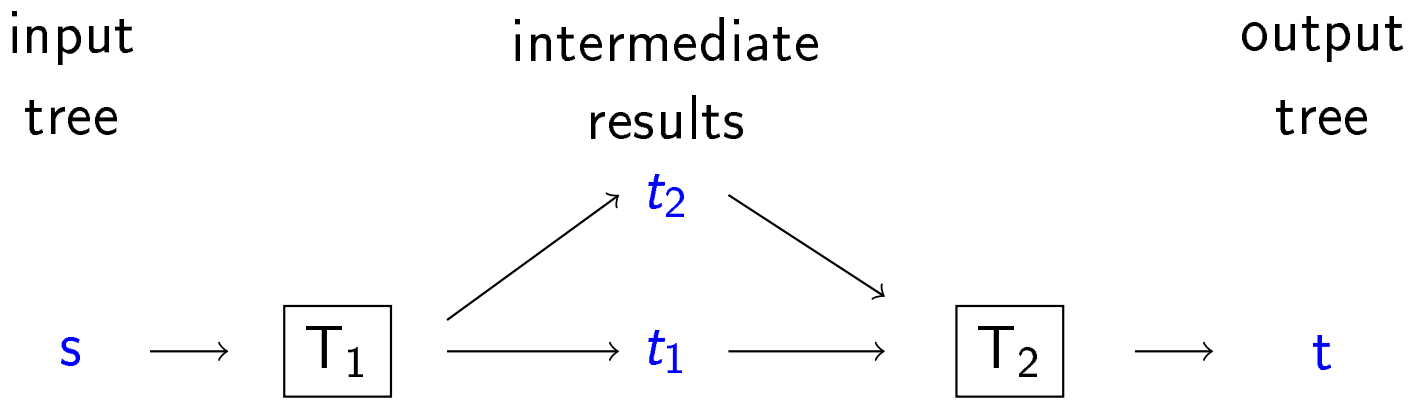


partial function

Main Idea

Observation: If the composition of T_1 and T_2 is not functional then M does not exist.

What does functional mean?

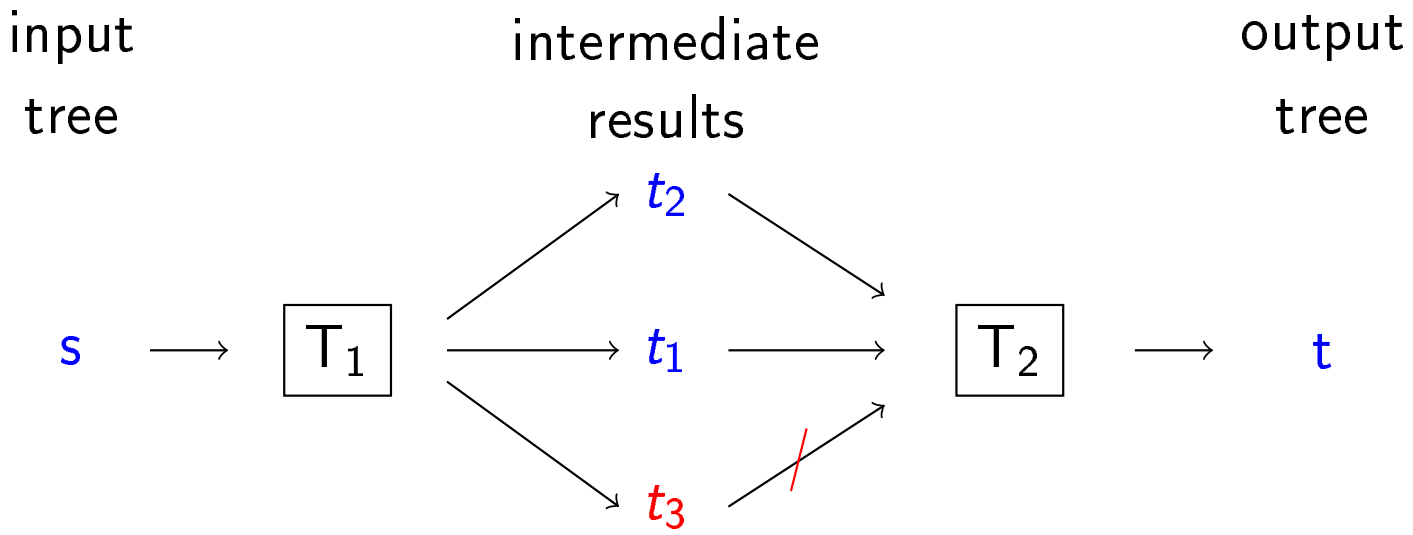


partial function

Main Idea

Observation: If the composition of T_1 and T_2 is not functional then M does not exist.

What does functional mean?



partial function

Main Idea

Observation: If the composition of T_1 and T_2 is not functional then M does not exist.

Main Idea

Observation: If the composition of T_1 and T_2 is not functional then M does not exist.

Engelfriet: If a composition of transducers is functional then an equivalent DT^R can be constructed.

Main Idea

Observation: If the composition of T_1 and T_2 is not functional then M does not exist.

Engelfriet: If a composition of transducers is functional then an equivalent DT^R can be constructed.

Therefore: We only need to check whether the composition of transducers is functional.

Main Idea

Observation: If the composition of T_1 and T_2 is not functional then M does not exist.

Engelfriet: If a composition of transducers is functional then an equivalent DT^R can be constructed.

Therefore: We only need to check whether the composition of transducers is functional.

Question:

How do we test whether or not a composition of transducers is functional?

How do we test whether or not a composition of transducers is functional?

Ésik: It is decidable whether or not a transducer (with look-ahead) is functional

How do we test whether or not a composition of transducers is functional?

Ésik: It is decidable whether or not a transducer (with look-ahead) is functional

Idea:

How do we test whether or not a composition of transducers is functional?

Ésik: It is decidable whether or not a transducer (with look-ahead) is functional

Idea:

1. Construct a transducer N with look-ahead such that

   N is functional $\iff$ the composition of T_1 and T_2 is functional

How do we test whether or not a composition of transducers is functional?

Ésik: It is decidable whether or not a transducer (with look-ahead) is functional

Idea:

1. Construct a transducer N with look-ahead such that

   N is functional $\iff$ the composition of T_1 and T_2 is functional
2. Test whether or not N is functional

How to construct N?

Baker has shown that linear, nondeleting transducer are closed under composition

How to construct N?

Baker has shown that linear, nondeleting transducer are closed under composition

Main challenges in the construction of N:

1. copying rules and
2. deleting rules

Copying Rules

Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$\begin{array}{lll} q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1)) & & q_2'(e_j) \rightarrow e \mid j = 1, 2 \\ q_2''(e_3) \rightarrow e' & & q_2''(e_j) \rightarrow e \mid j = 1, 2. \end{array}$$

Copying Rules

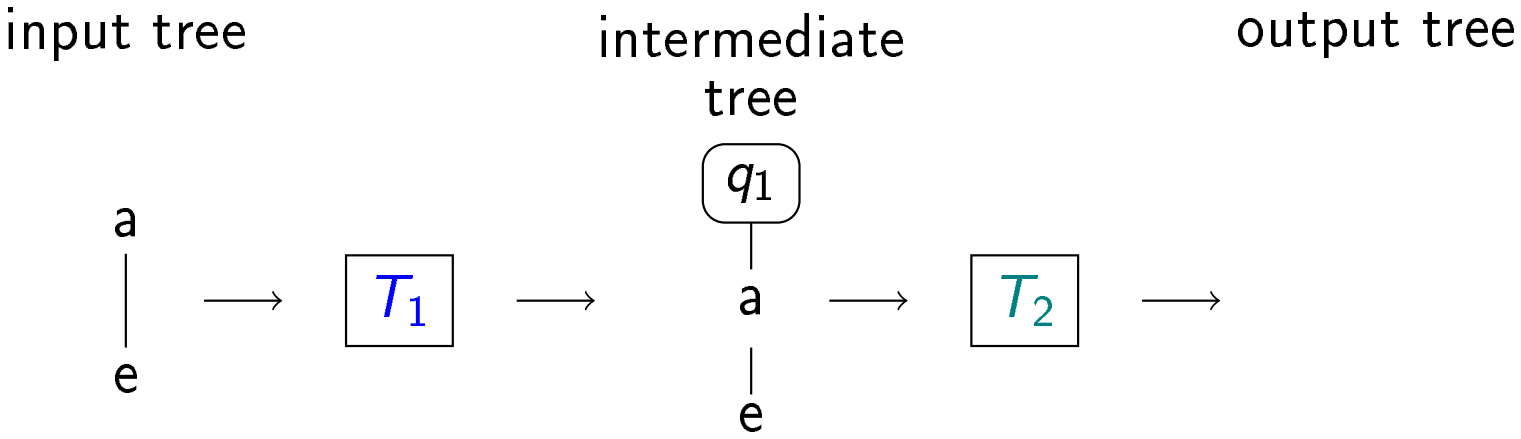
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$\begin{array}{llll} q_2(b(x_1)) & \rightarrow & f(q_2'(x_1), q_2''(x_1)) & q_2'(e_j) \rightarrow e \mid j = 1, 2 \\ q_2''(e_3) & \rightarrow & e' & q_2''(e_j) \rightarrow e \mid j = 1, 2. \end{array}$$

Transduction:



Copying Rules

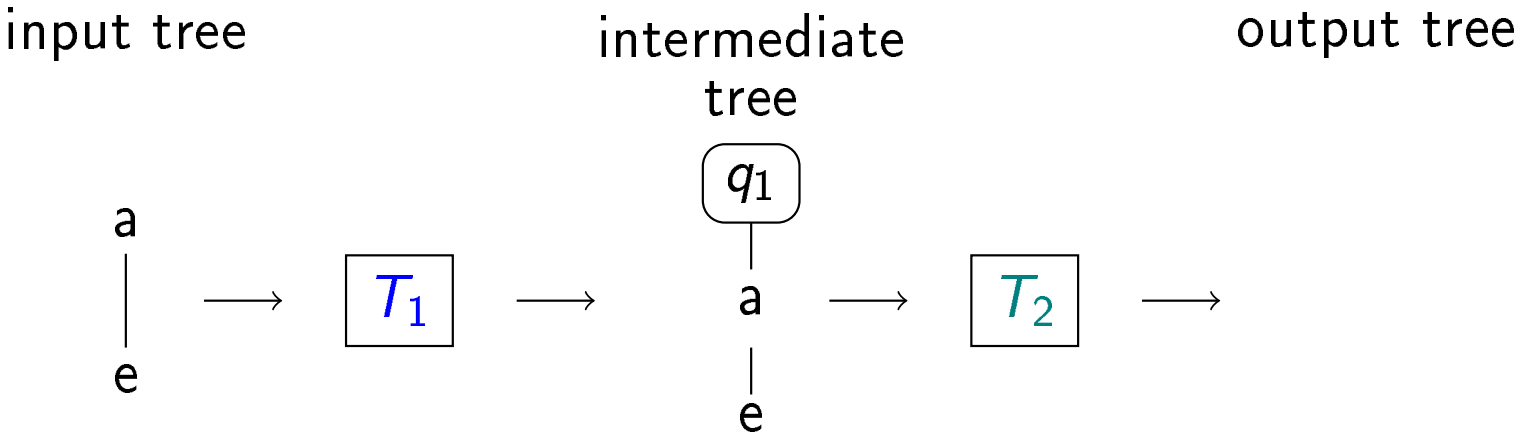
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1)) \qquad q_2'(e_j) \rightarrow e \mid j = 1, 2$$
$$q_2''(e_3) \rightarrow e' \qquad q_2''(e_j) \rightarrow e \mid j = 1, 2.$$

Transduction:



Copying Rules

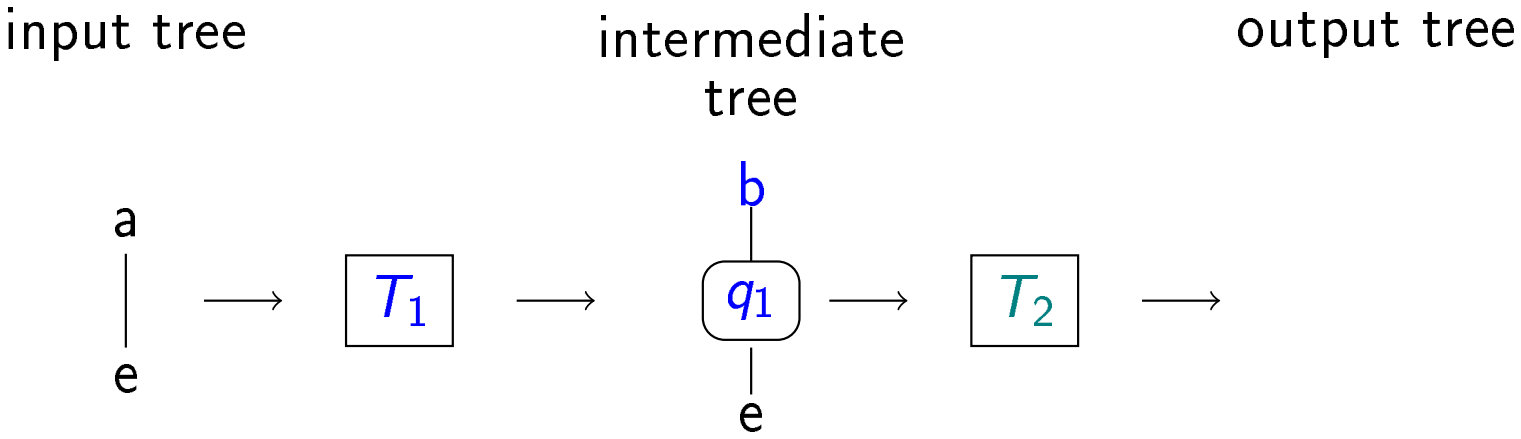
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1)) \qquad q_2'(e_j) \rightarrow e \mid j = 1, 2$$
$$q_2''(e_3) \rightarrow e' \qquad q_2''(e_j) \rightarrow e \mid j = 1, 2.$$

Transduction:



Copying Rules

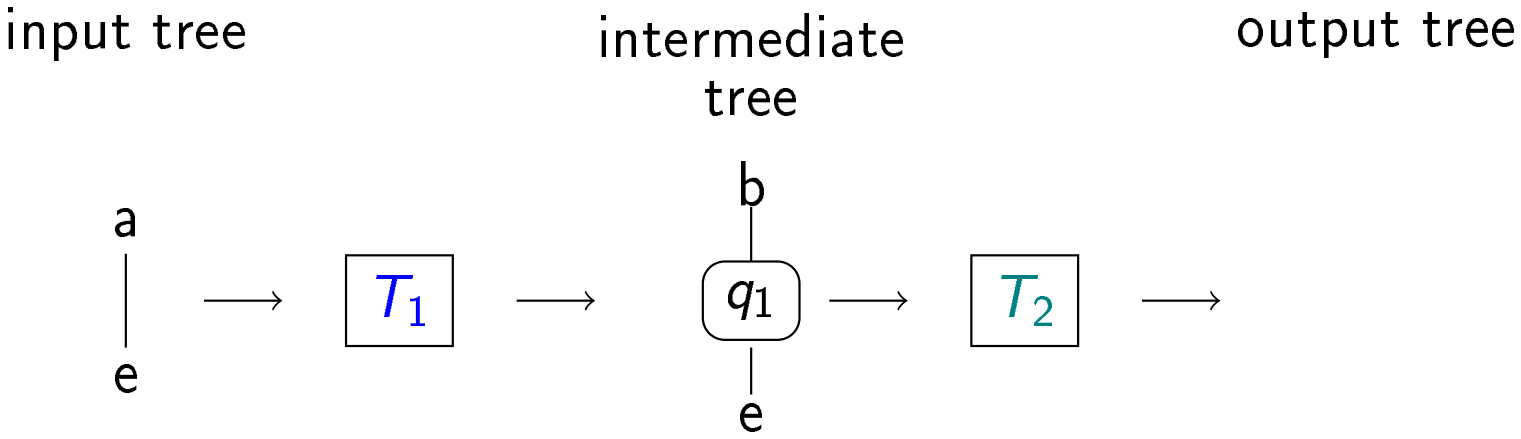
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1)) \qquad q_2'(e_j) \rightarrow e \mid j = 1, 2$$
$$q_2''(e_3) \rightarrow e' \qquad q_2''(e_j) \rightarrow e \mid j = 1, 2.$$

Transduction:



Copying Rules

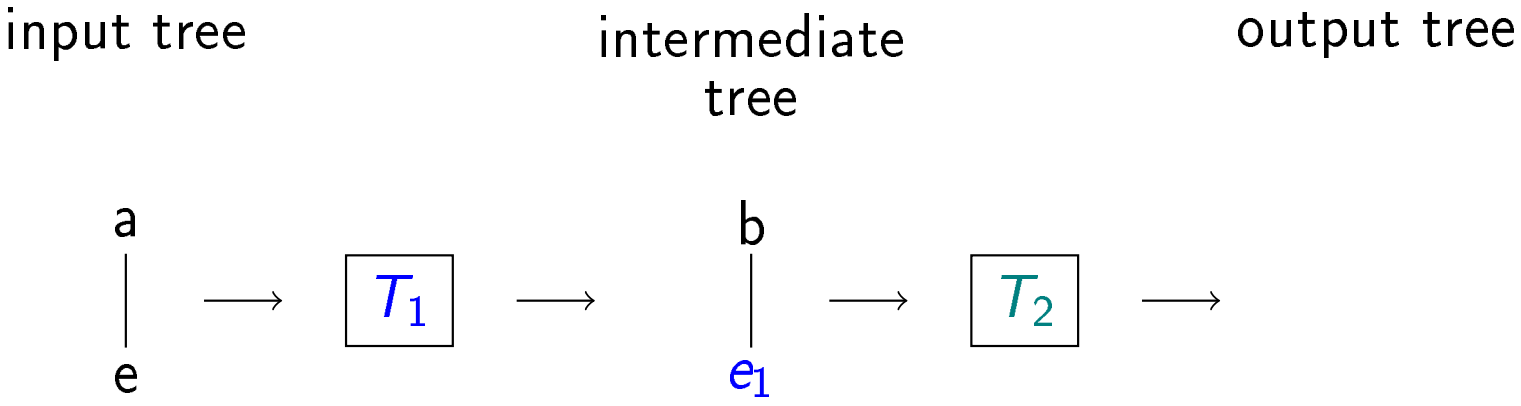
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$\begin{array}{lll} q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1)) & & q_2'(e_j) \rightarrow e \mid j = 1, 2 \\ q_2''(e_3) \rightarrow e' & & q_2''(e_j) \rightarrow e \mid j = 1, 2. \end{array}$$

Transduction:



Copying Rules

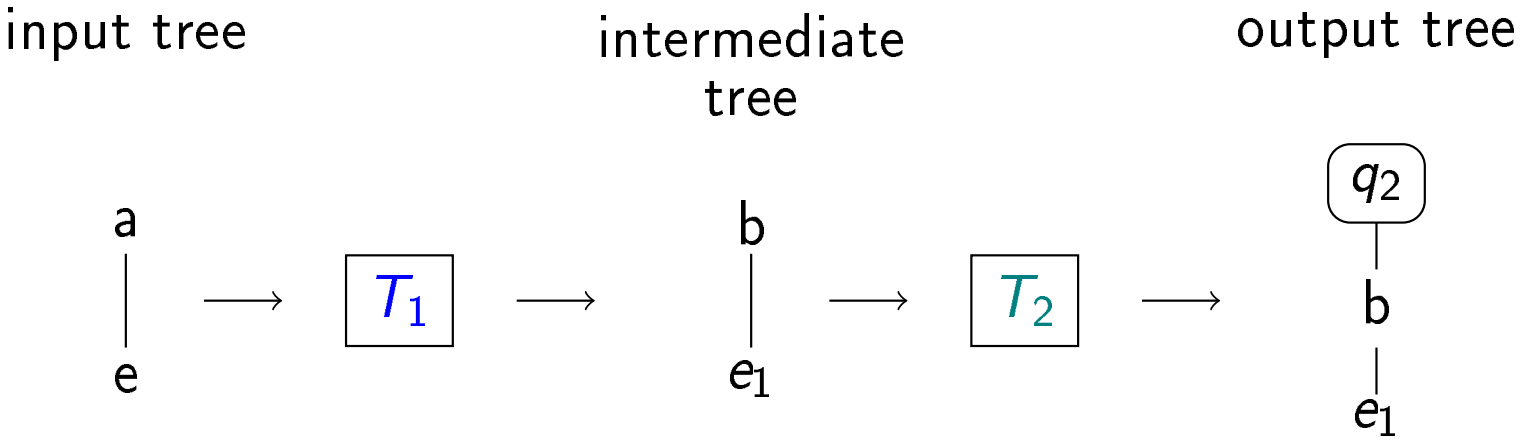
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1)) \qquad q_2'(e_j) \rightarrow e \mid j = 1, 2$$
$$q_2''(e_3) \rightarrow e' \qquad q_2''(e_j) \rightarrow e \mid j = 1, 2.$$

Transduction:



Copying Rules

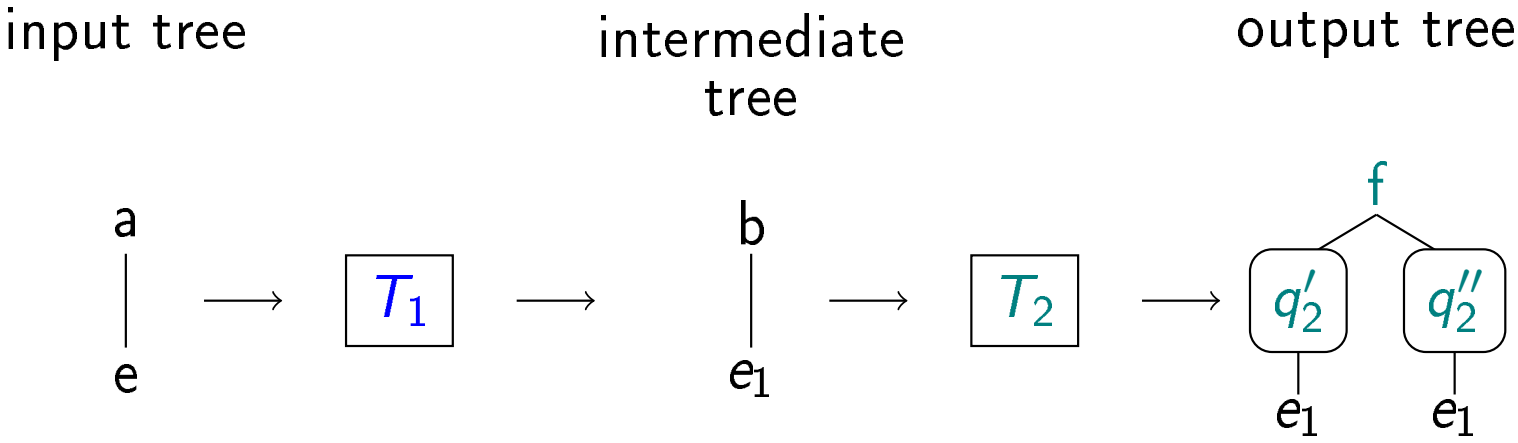
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1)) \qquad q_2'(e_j) \rightarrow e \mid j = 1, 2$$
$$q_2''(e_3) \rightarrow e' \qquad q_2''(e_j) \rightarrow e \mid j = 1, 2.$$

Transduction:



Copying Rules

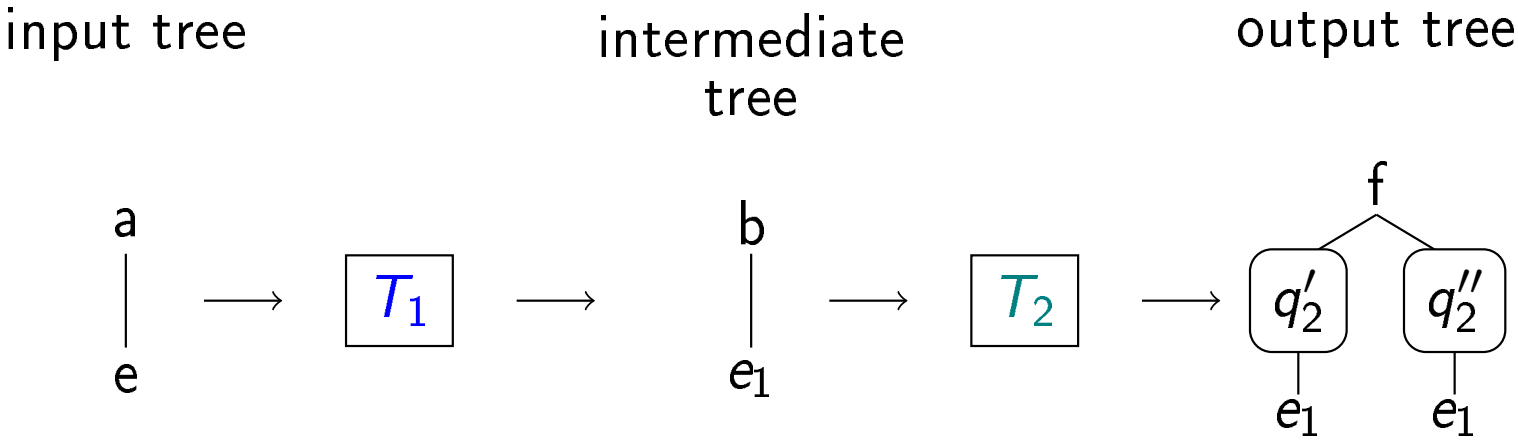
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1)) \qquad q_2'(e_j) \rightarrow e \mid j = 1, 2$$
$$q_2''(e_3) \rightarrow e' \qquad q_2''(e_j) \rightarrow e \mid j = 1, 2.$$

Transduction:



Copying Rules

Transducer T_1:

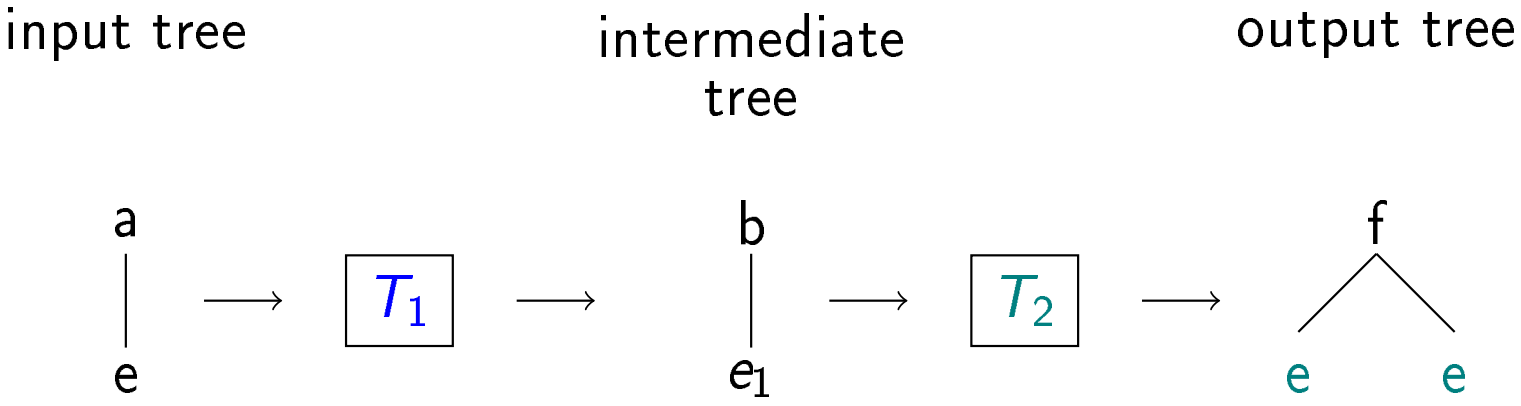
$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1))$$
$$q_2''(e_3) \rightarrow e'$$

$$q_2'(e_j) \rightarrow e \mid j = 1, 2$$
$$q_2''(e_j) \rightarrow e \mid j = 1, 2.$$

Transduction:



Copying Rules

Transducer T_1:

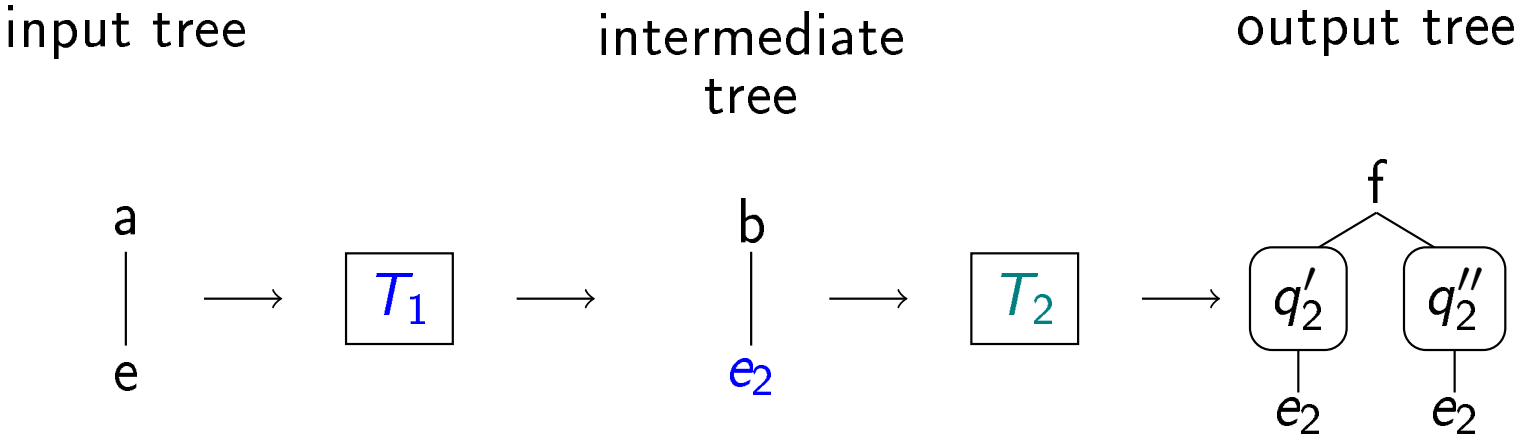
$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1))$$
$$q_2''(e_3) \rightarrow e'$$

$$q_2'(e_j) \rightarrow e \mid j = 1, 2$$
$$q_2''(e_j) \rightarrow e \mid j = 1, 2.$$

Transduction:



Copying Rules

Transducer T_1:

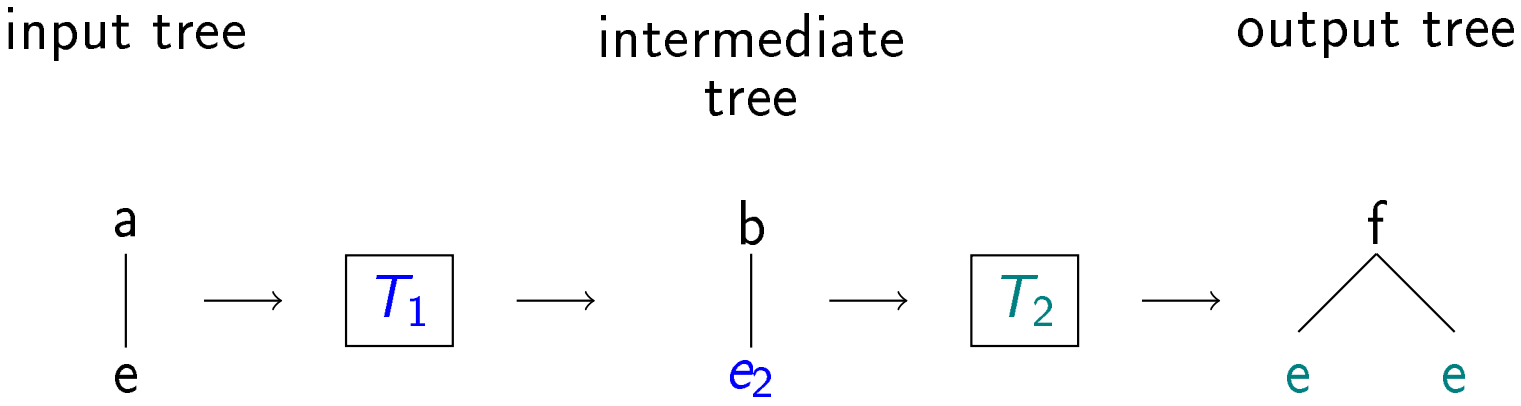
$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$q_2(b(x_1)) \rightarrow f(q_2'(x_1), q_2''(x_1))$$
$$q_2''(e_3) \rightarrow e'$$

$$q_2'(e_j) \rightarrow e \mid j = 1, 2$$
$$q_2''(e_j) \rightarrow e \mid j = 1, 2.$$

Transduction:



Copying Rules

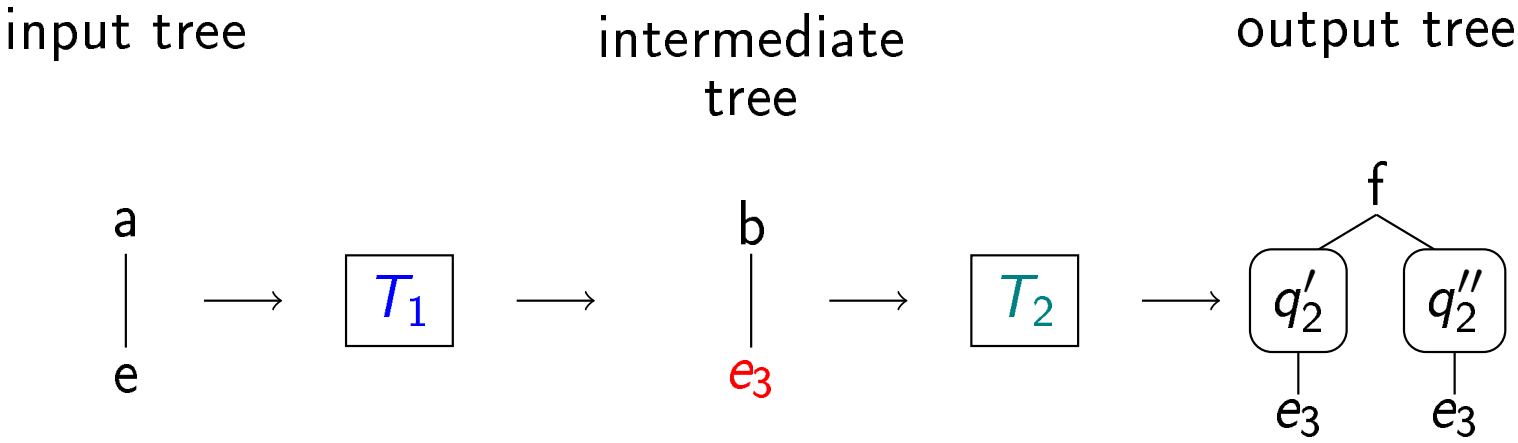
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$\begin{array}{llll} q_2(b(x_1)) & \rightarrow & f(q_2'(x_1), q_2''(x_1)) & q_2'(e_j) \rightarrow e \mid j = 1, 2 \\ q_2''(e_3) & \rightarrow & e' & q_2''(e_j) \rightarrow e \mid j = 1, 2. \end{array}$$

Transduction:



Copying Rules

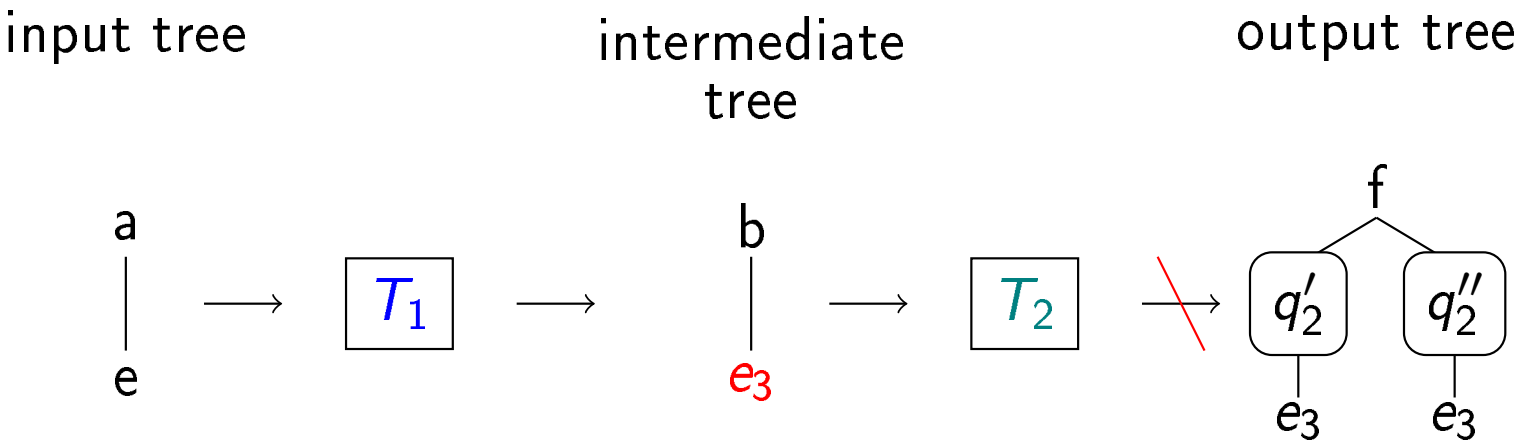
Transducer T_1:

$$q_1(a(x_1)) \rightarrow b(q_1(x_1)) \qquad q_1(e) \rightarrow e_i \mid i = 1, 2, 3$$

Transducer T_2:

$$\begin{array}{llll} q_2(b(x_1)) & \rightarrow & f(q_2'(x_1), q_2''(x_1)) & q_2'(e_j) \rightarrow e \mid j = 1, 2 \\ q_2''(e_3) & \rightarrow & e' & q_2''(e_j) \rightarrow e \mid j = 1, 2. \end{array}$$

Transduction:



Naive Construction of N

(q_1, q_2)

Naive Construction of N

$$q_1(a(x_1)) \to b(q_1(x_1))$$

$$(q_1, q_2)(a(x_1))$$

Naive Construction of N

$(q_1, q_2)(a(x_1))$

$q_1(a(x_1)) \to b(q_1(x_1))$

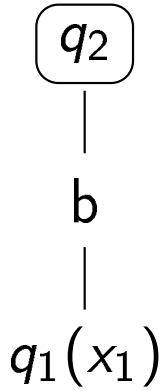
$q_2(b(q_1(x_1)))$

Naive Construction of N

$(q_1, q_2)(a(x_1))$

$q_1(a(x_1)) \to b(q_1(x_1))$

$q_2(b(q_1(x_1)))$



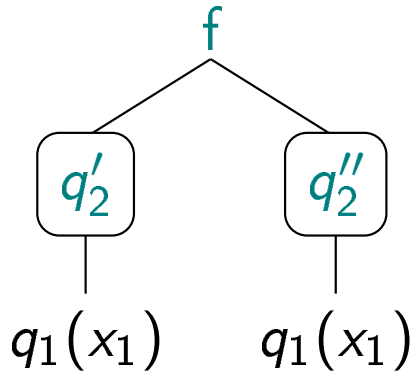
$q_2(b(x_1)) \to f(q_2'(x_1), q_2''(x_1))$

Naive Construction of N

$(q_1, q_2)(a(x_1))$

$q_1(a(x_1)) \to b(q_1(x_1))$

$q_2(b(q_1(x_1)))$



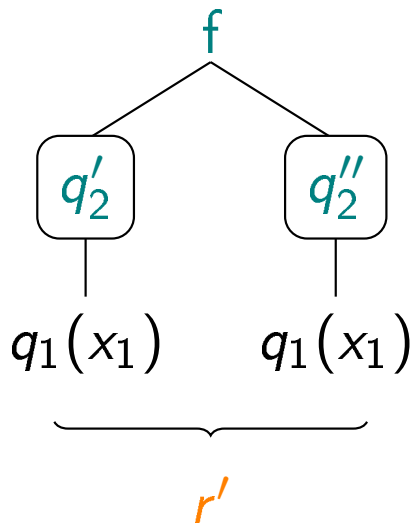
$q_2(b(x_1)) \to f(q_2'(x_1), q_2''(x_1))$

Naive Construction of N

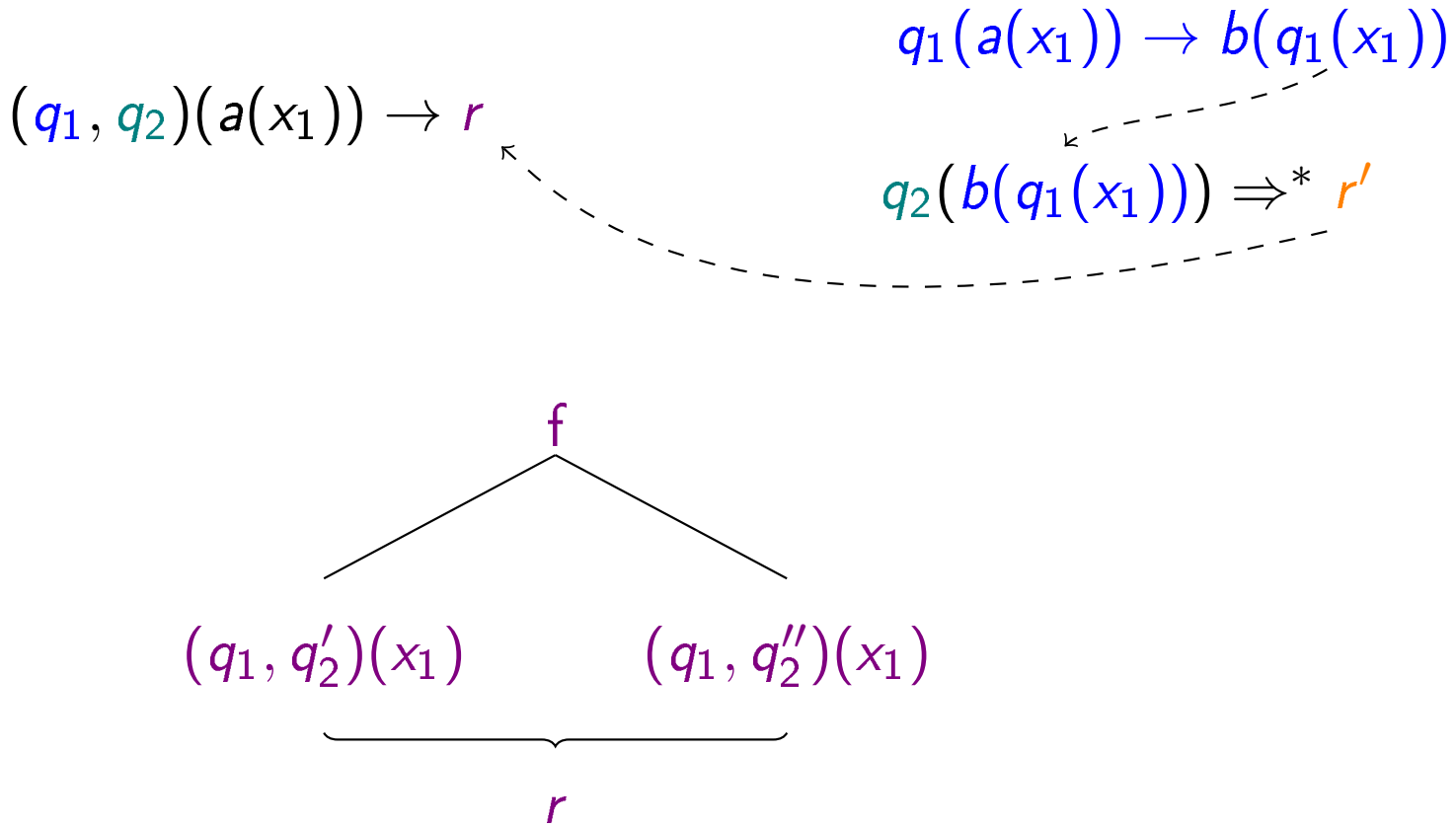
$(q_1, q_2)(a(x_1))$

$q_1(a(x_1)) \to b(q_1(x_1))$

$q_2(b(q_1(x_1))) \Rightarrow^* r'$



Naive Construction of N



Naive Construction of N

Transducer N:

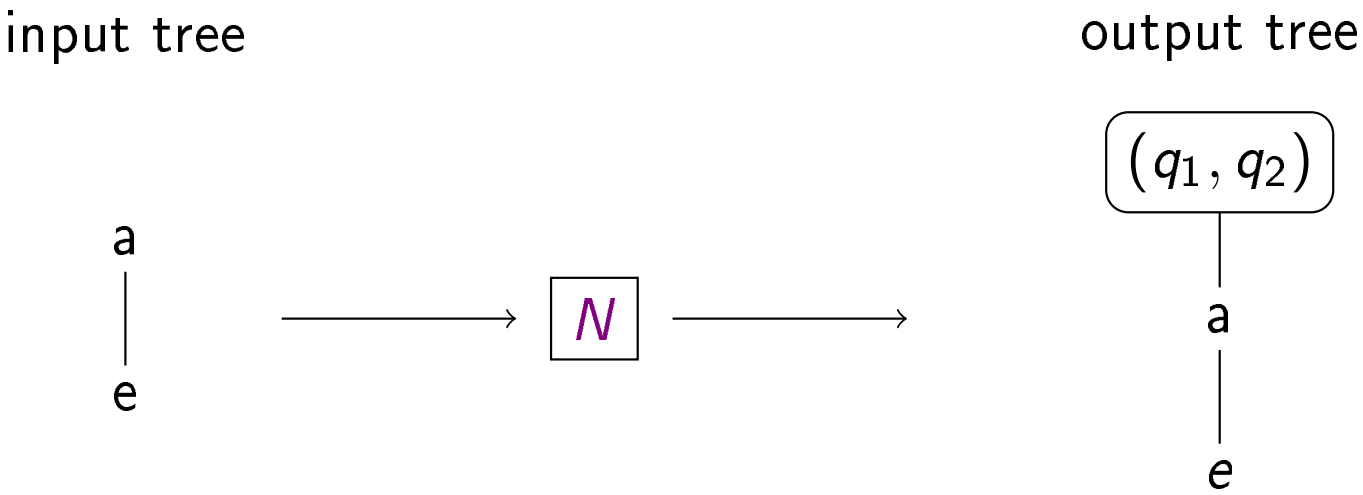
$$
\begin{array}{lcl}
(q_1, q_2)(a(x_1)) & \to & f\ ((q_1, q_2')(x_1), (q_1, q_2'')(x_1)) \\
(q_1, q_2'')(e) & \to & e
\end{array}
\qquad
\begin{array}{lcl}
(q_1, q_2')(e) & \to & e \\
(q_1, q_2'')(e) & \to & e'
\end{array}
$$

Naive Construction of N

Transducer N:

$$\begin{array}{llll} (q_1, q_2)(a(x_1)) & \to & f\ ((q_1, q_2')(x_1), (q_1, q_2'')(x_1)) & \\ (q_1, q_2'')(e) & \to & e & \end{array} \qquad \begin{array}{llll} (q_1, q_2')(e) & \to & e & \\ (q_1, q_2'')(e) & \to & e' & \end{array}$$

Transduction:

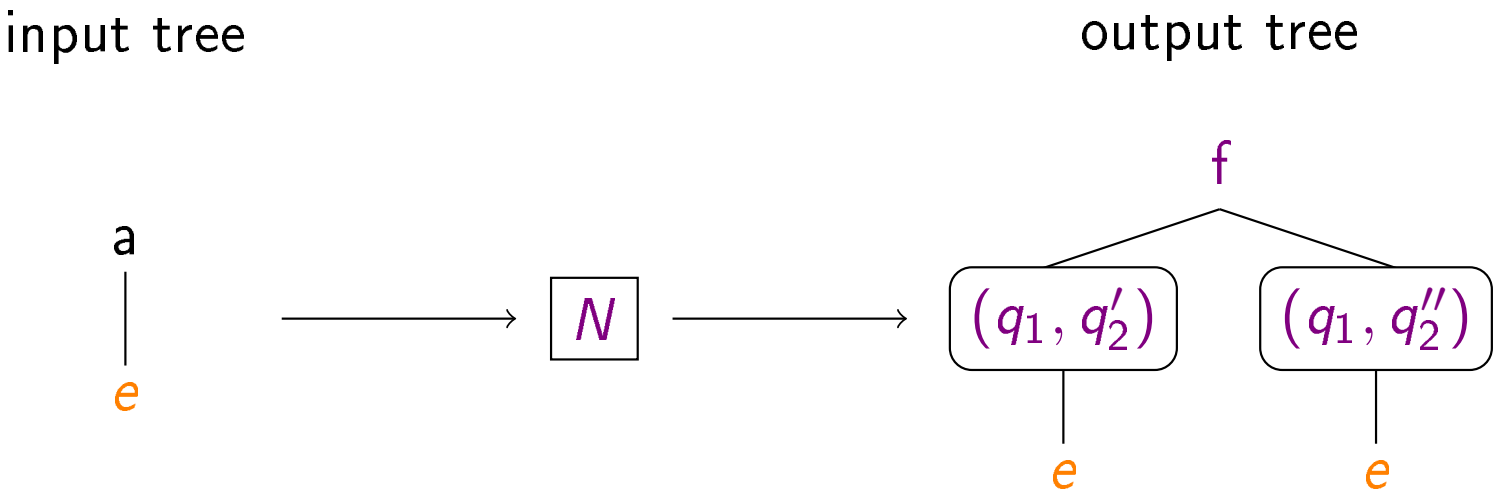


Naive Construction of N

Transducer N:

$$\begin{array}{lcl} (q_1, q_2)(a(x_1)) & \to & f\ ((q_1, q_2')(x_1), (q_1, q_2'')(x_1)) \\ (q_1, q_2'')(e) & \to & e \end{array} \qquad \begin{array}{lcl} (q_1, q_2')(e) & \to & e \\ (q_1, q_2'')(e) & \to & e' \end{array}$$

Transduction:



Naive Construction of N

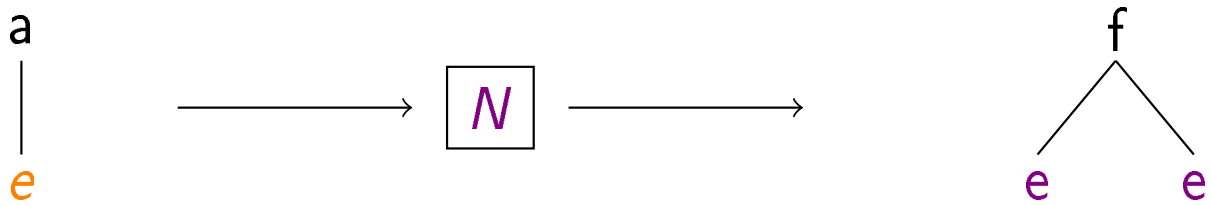
Transducer N:

$$(q_1, q_2)(a(x_1)) \to f\ ((q_1, q_2')(x_1), (q_1, q_2'')(x_1))$$
$$(q_1, q_2'')(e) \to e$$
$$(q_1, q_2')(e) \to e$$
$$(q_1, q_2'')(e) \to e'$$

Transduction:

input tree

output tree



Naive Construction of N

Transducer N:

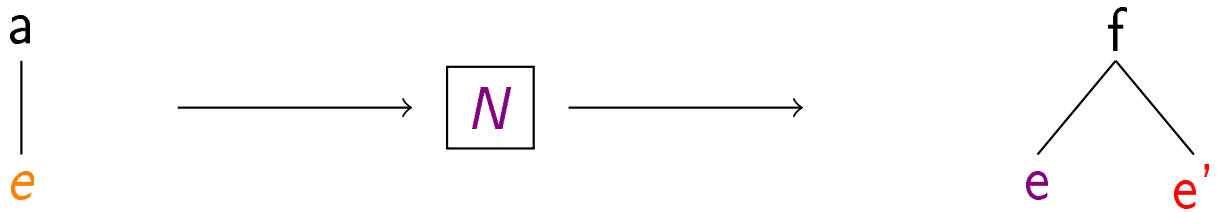
$$\begin{array}{lcl} (q_1, q_2)(a(x_1)) & \to & f\ ((q_1, q_2')(x_1), (q_1, q_2'')(x_1)) \\ (q_1, q_2'')(e) & \to & e \end{array}$$

$$\begin{array}{lcl} (q_1, q_2')(e) & \to & e \\ (q_1, q_2'')(e) & \to & e' \end{array}$$

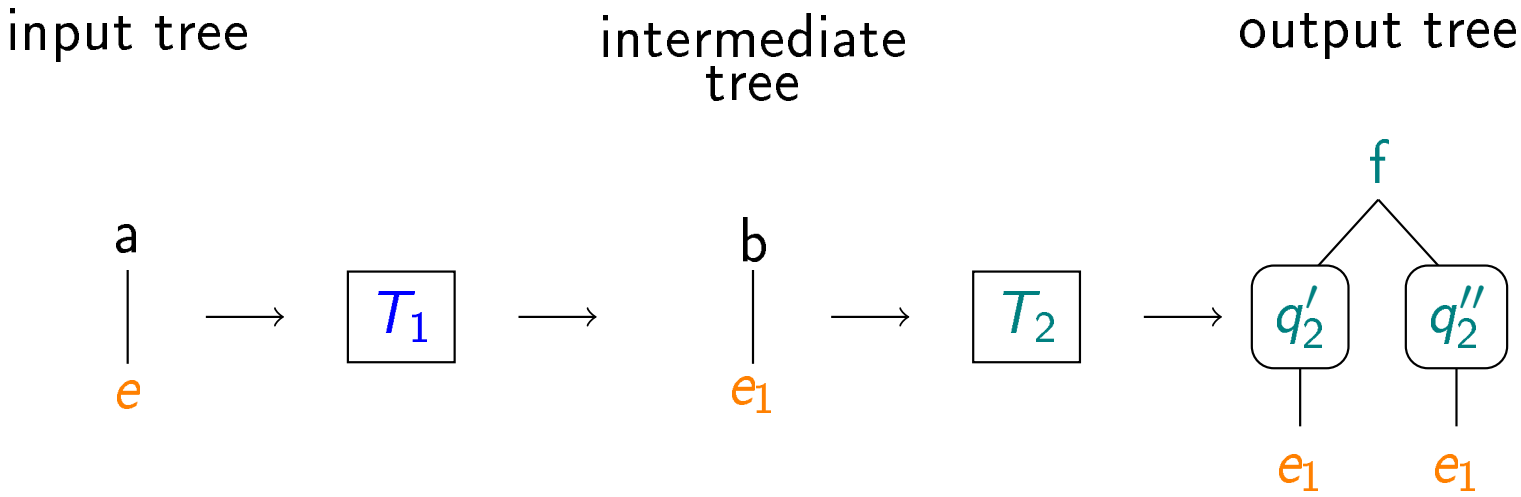
Transduction:

input tree

output tree

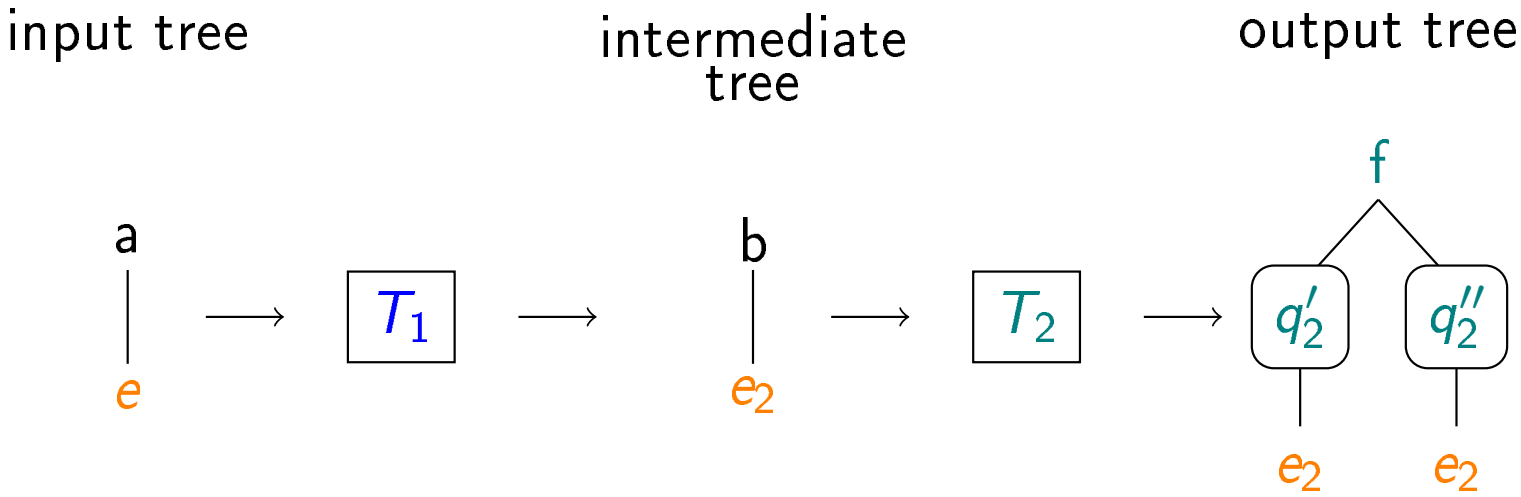


Problems of the naive Construction



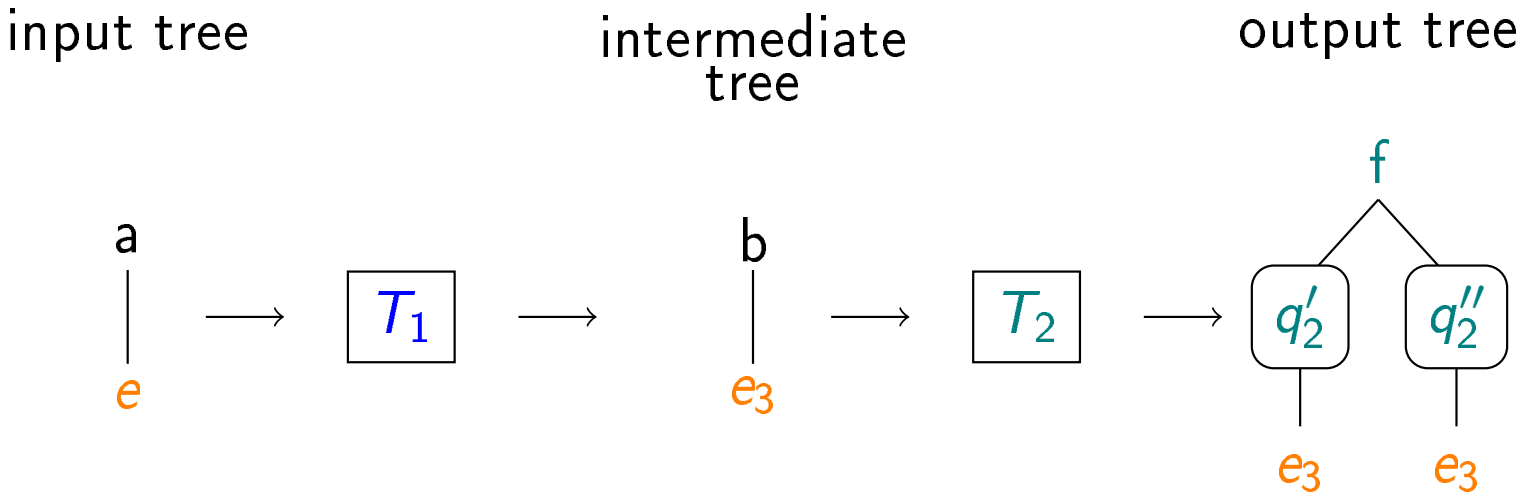
q_2' and q_2'' process the tree e_1 produced by q_1 on input e

Problems of the naive Construction



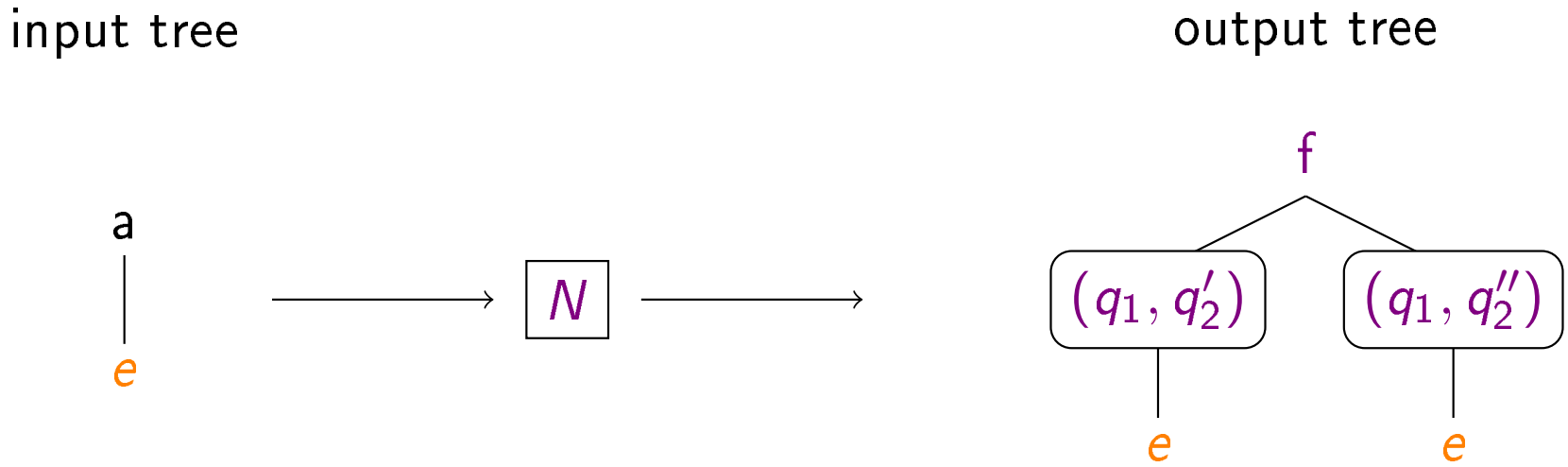
q_2' and q_2'' process the tree e_2 produced by q_1 on input e

Problems of the naive Construction



q_2' cannot process the tree e_3 produced by q_1 on input e

Problems of the naive Construction

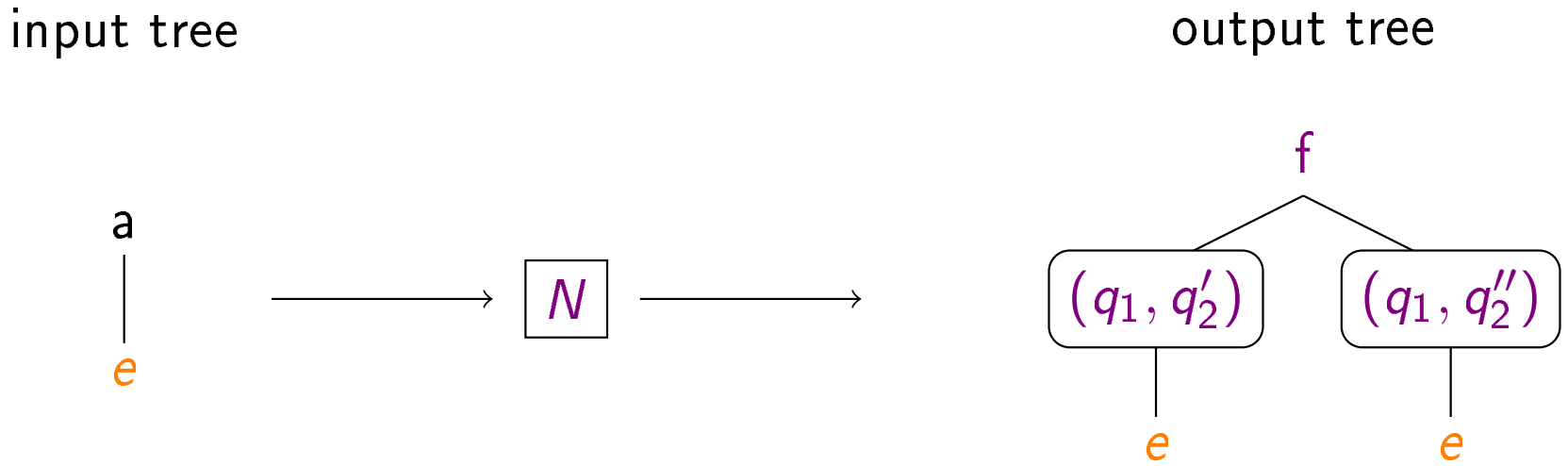


$$(q_1, q_2')(e) \to e \begin{cases} q_1(e) \to e_1 \\ q_2'(e_1) \to e \end{cases} \qquad (q_1, q_2'')(e) \to e \begin{cases} q_1(e) \to e_1 \\ q_2''(e_1) \to e \end{cases}$$

$$(q_1, q_2')(e) \to e \begin{cases} q_1(e) \to e_2 \\ q_2'(e_2) \to e \end{cases} \qquad (q_1, q_2'')(e) \to e \begin{cases} q_1(e) \to e_2 \\ q_2''(e_2) \to e \end{cases}$$

$$(q_1, q_2'')(e) \to e' \begin{cases} q_1(e) \to e_3 \\ q_2''(e_3) \to e' \end{cases}$$

Problems of the naive Construction



unfortunately, synchronization is not possible...

Observation

$$(q_1, q_2')(e) \to e \begin{cases} q_1(e) \to e_1 \\ q_2'(e_1) \to e \end{cases} \qquad (q_1, q_2'')(e) \to e \begin{cases} q_1(e) \to e_1 \\ q_2''(e_1) \to e \end{cases}$$

$$(q_1, q_2')(e) \to e \begin{cases} q_1(e) \to e_2 \\ q_2'(e_2) \to e \end{cases} \qquad (q_1, q_2'')(e) \to e \begin{cases} q_1(e) \to e_2 \\ q_2''(e_2) \to e \end{cases}$$

Observation

$$(q_1, q_2')(e) \to e \begin{cases} q_1(e) \to e_1 \\ q_2'(e_1) \to e \end{cases} \qquad (q_1, q_2'')(e) \to e \begin{cases} q_1(e) \to e_1 \\ q_2''(e_1) \to e \end{cases}$$

$$(q_1, q_2')(e) \to e \begin{cases} q_1(e) \to e_2 \\ q_2'(e_2) \to e \end{cases} \qquad (q_1, q_2'')(e) \to e \begin{cases} q_1(e) \to e_2 \\ q_2''(e_2) \to e \end{cases}$$

Observation: It is sufficient that

1. (q_1, q_2') only guesses trees in $\mathrm{dom}(q_2') \cap \mathrm{dom}(q_2'')$
2. (q_1, q_2'') only guesses trees in $\mathrm{dom}(q_2') \cap \mathrm{dom}(q_2'')$

Idea:

Modification of T_1:

Idea:

Modification of T_1:

- new states are of the form $(q_1, \{q_2', q_2''\})$

Idea:

Modification of T_1:

- new states are of the form $(q_1, \{q_2', q_2''\})$
- $(q_1, \{q_2', q_2''\})$ only produces trees in $\mathrm{dom}(q_2') \cap \mathrm{dom}(q_2'')$

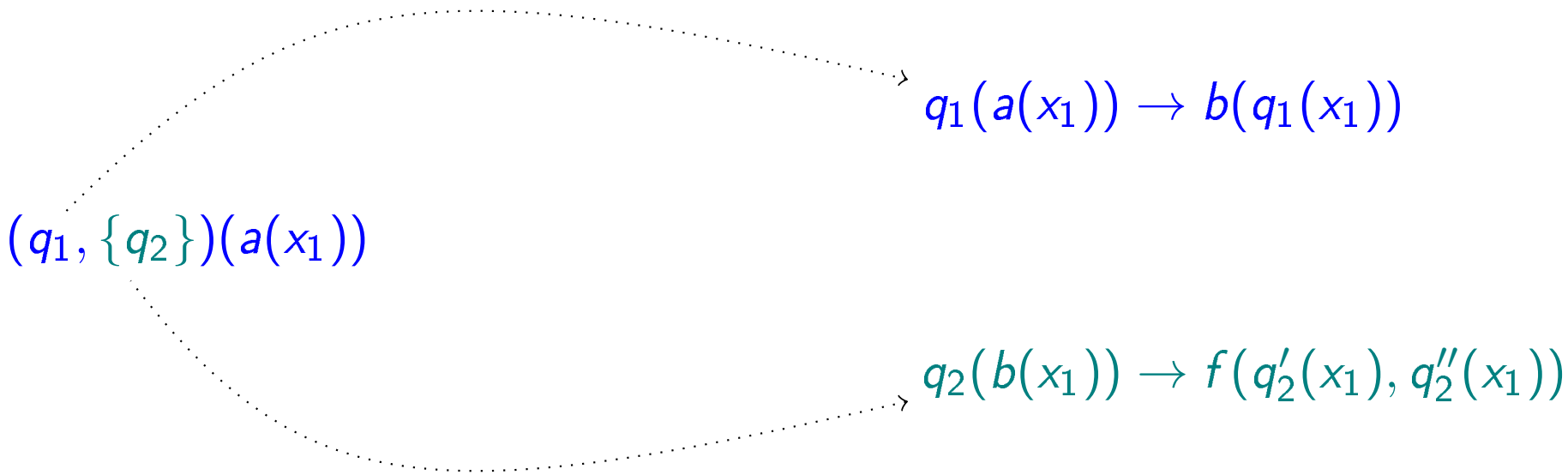
Idea:

Modification of T_1:

- new states are of the form $(q_1, \{q_2', q_2''\})$
- $(q_1, \{q_2', q_2''\})$ only produces trees in $\mathrm{dom}(q_2') \cap \mathrm{dom}(q_2'')$

How to obtain the modified T_1?

idea: T_1 simulates T_2 on its output



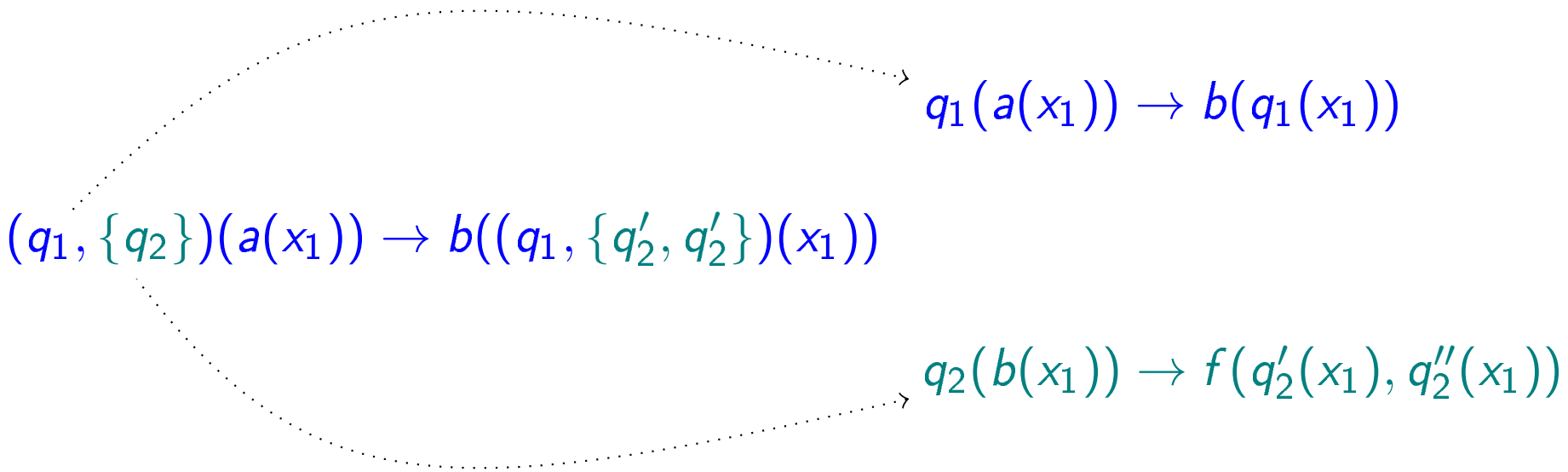
Idea:

Modification of T_1:

- new states are of the form $(q_1, \{q_2', q_2''\})$
- $(q_1, \{q_2', q_2''\})$ only produces trees in $\mathrm{dom}(q_2') \cap \mathrm{dom}(q_2'')$

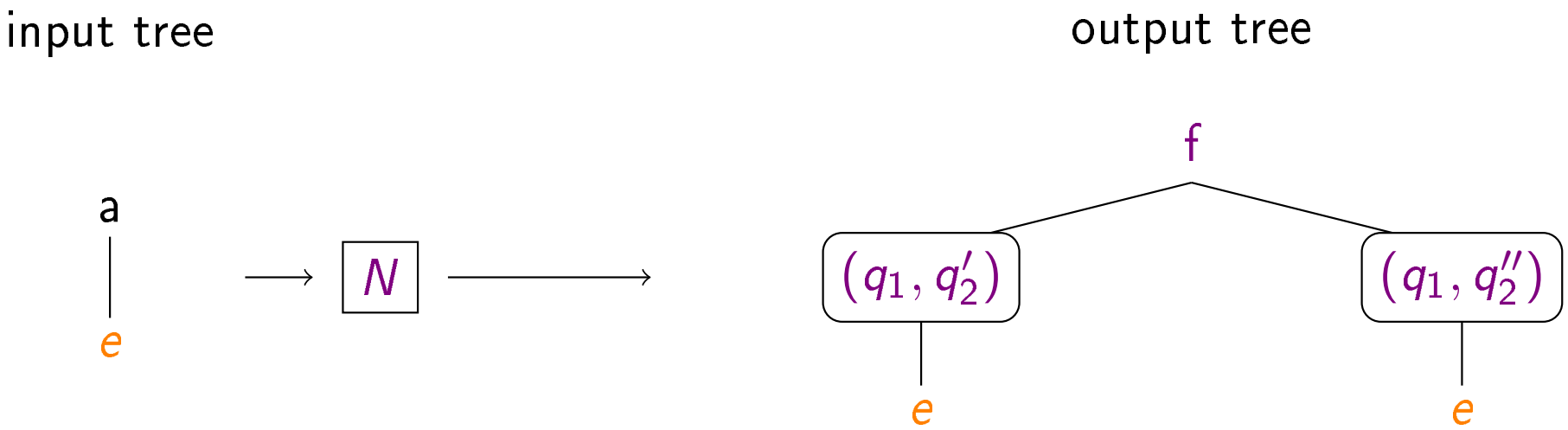
How to obtain the modified T_1?

idea: T_1 simulates T_2 on its output



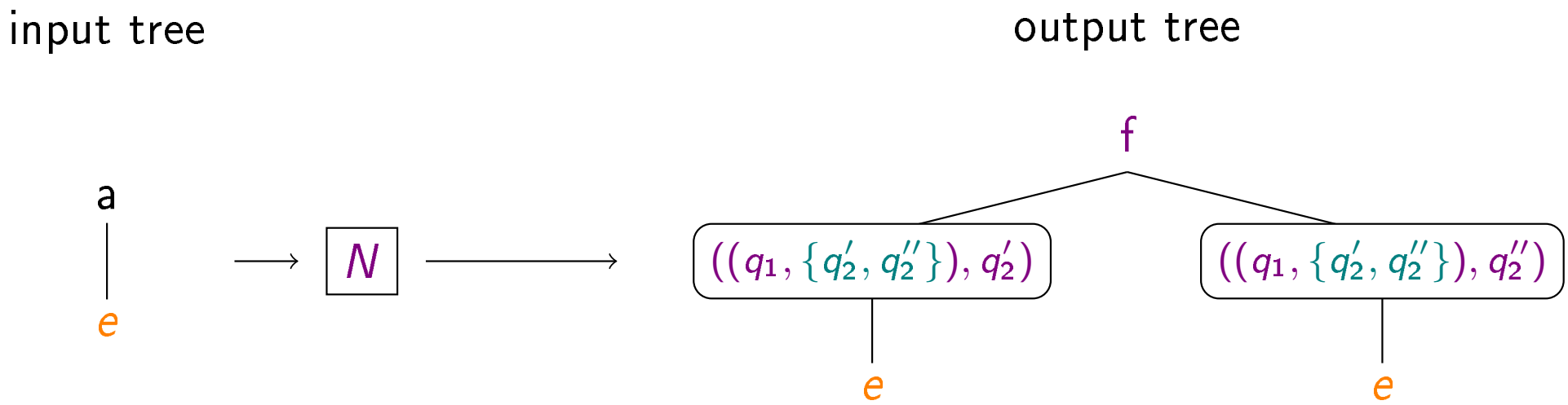
Idea:

Naive construction with the modified T_1:



Idea:

Naive construction with the modified T_1:



Deleting Rules

Transducer T_1:

$$\begin{array}{lcl} q_1(a(x_1, x_2)) & \to & b(q_1'(x_1), q_1''(x_2), q_1'''(x_2)) \\ q_1'(e) & \to & e \end{array}$$

$\mathrm{dom}(q_1')$ consists of trees whose leftmost leaf is e
$\mathrm{dom}(q_1'')$ consists of trees whose leftmost leaf is c

Transducer T_2:

$$\begin{array}{lcll} q_2(b(x_1, x_2, x_3)) & \to & q_2(x_1) & \\ q_2(e) & \to & e_j & \mid j = 1, 2. \end{array}$$

Deleting Rules

Transducer T_1:

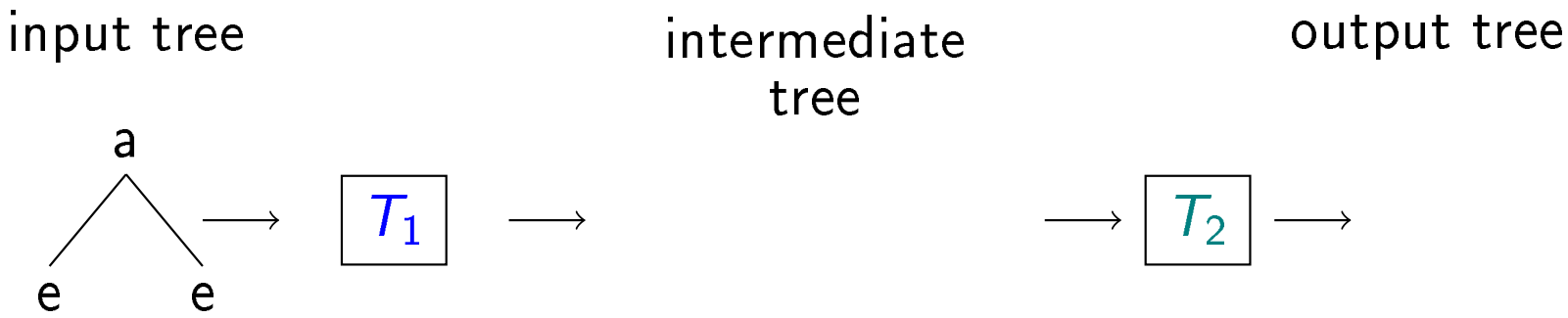
$$\begin{array}{lcl} q_1(a(x_1, x_2)) & \to & b(q_1'(x_1), q_1''(x_2), q_1'''(x_2)) \\ q_1'(e) & \to & e \end{array}$$

$\mathrm{dom}(q_1')$ consists of trees whose leftmost leaf is e
$\mathrm{dom}(q_1'')$ consists of trees whose leftmost leaf is c

Transducer T_2:

$$\begin{array}{lcll} q_2(b(x_1, x_2, x_3)) & \to & q_2(x_1) & \\ q_2(e) & \to & e_j & \mid j = 1, 2. \end{array}$$

Transduction:



Deleting Rules

Transducer T_1:

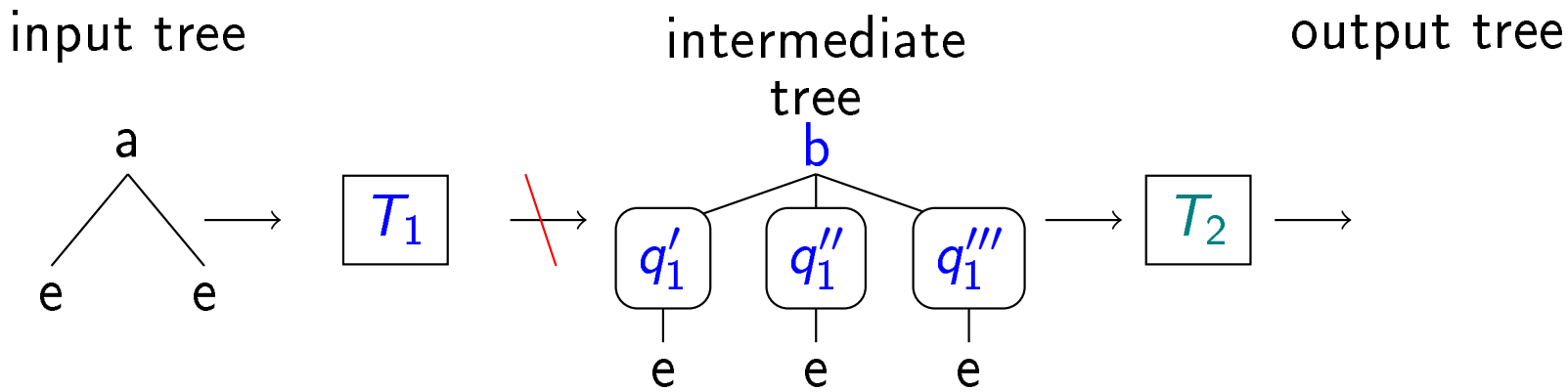
$$\begin{array}{lcl} q_1(a(x_1, x_2)) & \to & b(q_1'(x_1), q_1''(x_2), q_1'''(x_2)) \\ q_1'(e) & \to & e \end{array}$$

$\mathrm{dom}(q_1')$ consists of trees whose leftmost leaf is e
$\mathrm{dom}(q_1'')$ consists of trees whose leftmost leaf is c

Transducer T_2:

$$\begin{array}{lcll} q_2(b(x_1, x_2, x_3)) & \to & q_2(x_1) & \\ q_2(e) & \to & e_j & \mid j = 1, 2. \end{array}$$

Transduction:

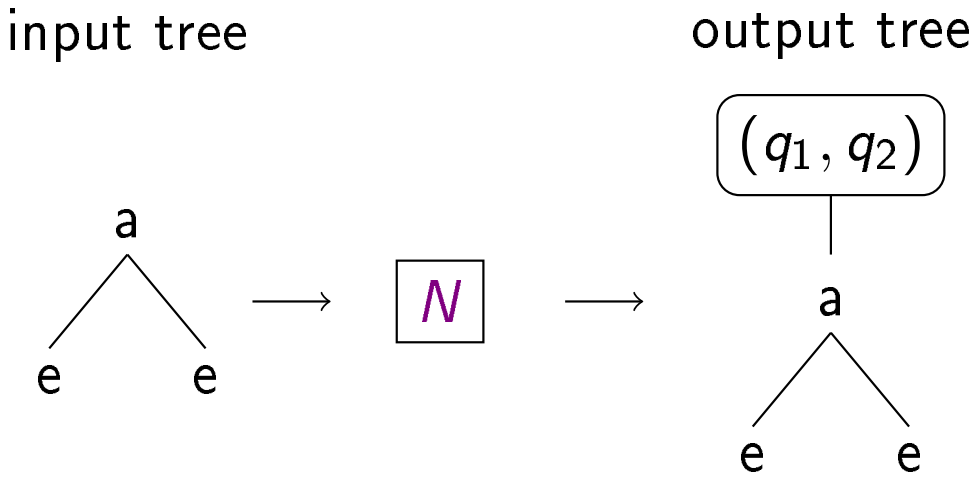


Naive Construction:

Transducer N:

$$(q_1, q_2)(a(x_1, x_2)) \rightarrow (q_1', q_2)(x_1) \qquad (q_1', q_2)(e) \rightarrow e_j \mid j = 1, 2$$

Transduction:

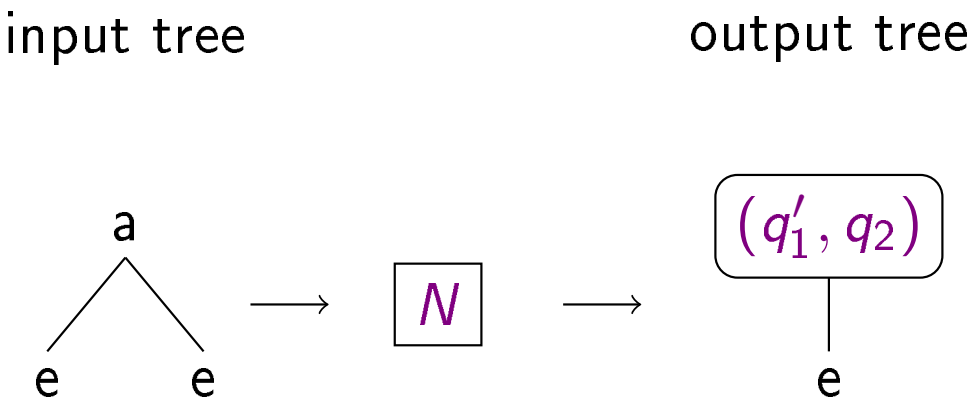


Naive Construction:

Transducer N:

$$(q_1, q_2)(a(x_1, x_2)) \rightarrow (q_1', q_2)(x_1) \qquad (q_1', q_2)(e) \rightarrow e_j \mid j = 1, 2$$

Transduction:



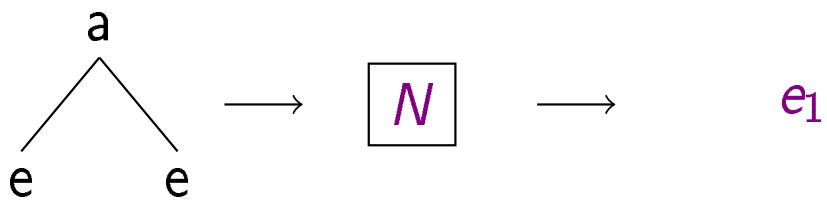
Naive Construction:

Transducer N:

$$(q_1, q_2)(a(x_1, x_2)) \rightarrow (q_1', q_2)(x_1) \qquad (q_1', q_2)(e) \rightarrow e_j \mid j = 1, 2$$

Transduction:

input tree output tree



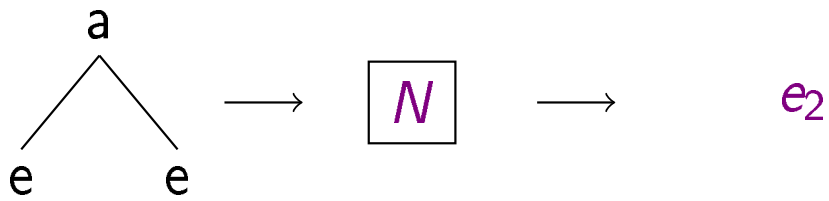
Naive Construction:

Transducer N:

$$(q_1, q_2)(a(x_1, x_2)) \rightarrow (q_1', q_2)(x_1) \qquad (q_1', q_2)(e) \rightarrow e_j \mid j = 1, 2$$

Transduction:

input tree output tree



Idea:

Before processing a tree $a(s_1, s_2)$ with th rule

$$(q_1, q_2)(a(x_1, x_2)) \to (q_1', q_2)(x_1) \begin{cases} q_1(a(x_1, x_2)) \to b(q_1'(x_1), q_1''(x_2), q_1'''(x_2)) \\ q_2(b(x_1, x_2, x_3)) \to q_2(x_1) \end{cases}$$

test whether $s_1 \in \mathrm{dom}(q_1')$ and $s_2 \in \mathrm{dom}(q_1'') \cap \mathrm{dom}(q_1''')$ using Look-Ahead

Result

Combining our ideas for copying and deleting rules yields:

Theorem

Functionality for arbitrary compositions of top-down and bottom-up tree transducers is decidable. In the affirmative case, an equivalent deterministic top-down tree transducer with look-ahead can be constructed.

Thank you for your Attention