Incremental construction and maintenance of morphological analysers based on augmented letter transducers[*]

Alicia Garrido-Alenda, *Mikel L. Forcada* and Rafael C. Carrasco

www.*interNOSTRUM*.com

Departament de Llenguatges i Sistemes Informàtics

Universitat d'Alacant

E-03071 Alacant, Spain

[*]Funded by Caja de Ahorros del Mediterráneo, Universitat d'Alacant and CICyT (project TIC2000-1599-C02-02).

Index

Morphological analysers/1 : An important component of MT systems

Morphological analysers: an important component of machine translation systems.

They are the first to deal with the source text . . .

. . . and identify and classify lexically relevant units.

Morphological analysers/2 : A wishlist

Ideally, a morphological analyser:

Reads text only once, left-to-right

Divides text into (context-sensitive) lexical tokens (**surface forms**)

Incrementally outputs the corresponding **lexical forms** as it reads text

Is **fast**. . . and **compact!**

And may be **easily updated**. . . while **keeping it compact**!

Finite-state letter transducers/1

Have a finite set of states Q,

with an **initial state** q_I and a set of **acceptance states** F.

State-to-state arrows have input–output labels (σ, γ).

Input σ can be an input symbol from Σ or nothing (ϵ)

Output γ can be an output symbol from Γ or nothing (ϵ)

Clearly, (ϵ, ϵ) arrows do nothing may be avoided.

Finite-state letter transducers/2

Formally, they are basically finite-state automata

$$T = (Q, L, \delta, q_I, F),$$

with a pair alphabet $L = (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\})$

or, removing (ϵ, ϵ), $L = ((\Sigma \cup \{\epsilon\}) \times \Gamma) \cup (\Sigma \times (\Gamma \cup \{\epsilon\}))$

which may be easily be made deterministic with respect to L so that

$$\delta : Q \times L \rightarrow Q.$$

as deterministic finite-state automata (DFA)*.

*They are in general nondeterministic with respect to Σ.

Finite-state letter transducers/3

Using letter transducers for **morphological analysis**:

If there is a state-by-state arrow **path***. . .

. . . going from the initial state q_I. . .

. . . to an acceptance state in F. . .

. . . so that the σ's spell the corresponding *surface form*, . . .

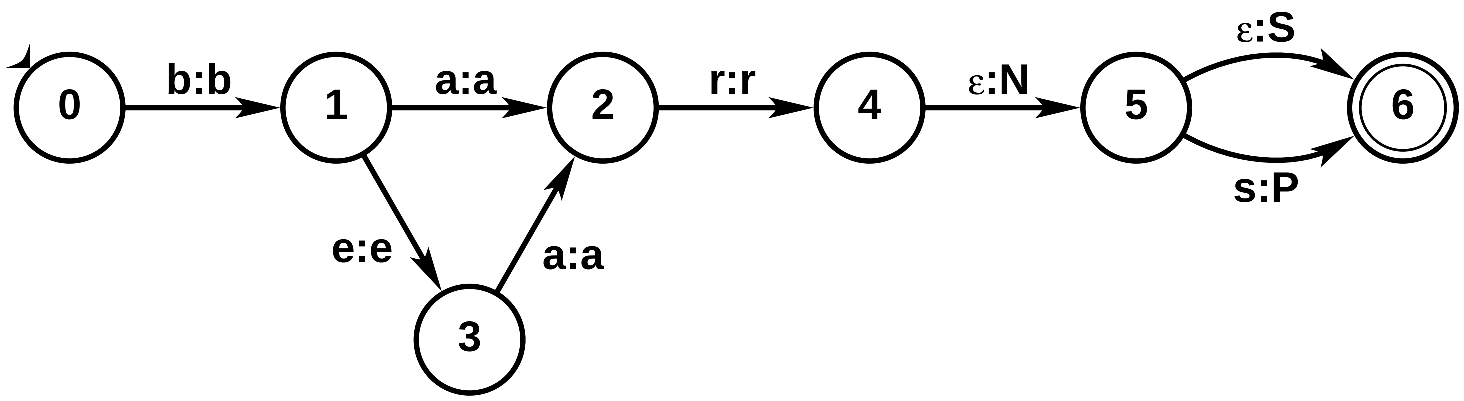
. . . then the γ's spell a *lexical form*.

State paths easily computed as the surface form is read.

*There may be more than one!!

Finite-state letter transducers/4

An example of a transducer:



This transducer accepts (bar··:barNS), (bar·s:barNP), (bear··:bearNS), (bear·s:bearNP).

Finite-state letter transducers/5

Finite-state letter transducers may also be used for other lexical mapping tasks in machine translation:

- Lexical transfer (bilingual dictionary)
- Target-language morphological generation

Finite-state letter transducers/6

But *in morphological analysis* surface forms have to be “cut” (**tokenization**) from text

How long is a surface form?

One would not cut . . .

. . . `international` in “internationalization. . .” (letters follow),
. . . `The` in “The Hague. . .” (multiword unit).

Finite-state letter transducers/7

How long is a surface form? Some problems...

A period (.) . . .

. . . may be part of an abbreviation, as in “Ph.D.” . . .

. . . or a full stop, as in “He left. Then. . . ” . . .

. . . even if one sloppily writes “He left.Then . . . ”.

Cutting benefits from previous context.

Finite-state letter transducers/7

We need a means to *divide input in surface forms*.

Longest match, left to right: start at the initial state,

eat input until no path is possible,

cut the longest surface form (the last one to visit an acceptance state).

But... is `bar` the correct SF when reading `barqzaxx`?

Not really! We also need *lookahead*!

Augmented finite-state letter transducers/1

Finite-state letter transducers with lookahead and two acceptance levels:

$$(Q, L, \delta, q_I, \xi_s, \xi_w).$$

Lookahead functions specify validation lookahead symbols at each state

$$\xi_s : Q \to \Sigma \cup \{\$\} \quad \text{(strong)}$$
$$\xi_w : Q \to \Sigma \cup \{\$\} \quad \text{(weak)}$$

Where $\{\$\}$ = end of the input file.

Acceptance states are those with $\xi_w(q) \cup \xi_s(q) \neq \emptyset$.

Augmented finite-state letter transducers/2

Surface forms are strongly validated if:

- they reach a state q with $\xi_s(q) \neq \emptyset$;
- they are followed by a valid symbol $\sigma \in \xi_s(q)$

Weak validation (e.g., for out-of-dictionary words) uses $\xi_w(q)$ instead.

Strongly validated analyses therefore prevail.

Minimality...

[For technicalities, see paper]

DALT (deterministic augmented letter transducers) may be minimized as DFA. . .

. . . just note: not all acceptance states are equivalent (different lookahead sets).

Minimal DALT are equally fast but *more compact*!

The challenge: *adding and removing entries* while preserving *minimality*.

Adding an entry /1

Lookahead management ignored in discussion for clarity (see paper).

Adding an entry = adding a path...

1. *Clone* states visited by the new entry;

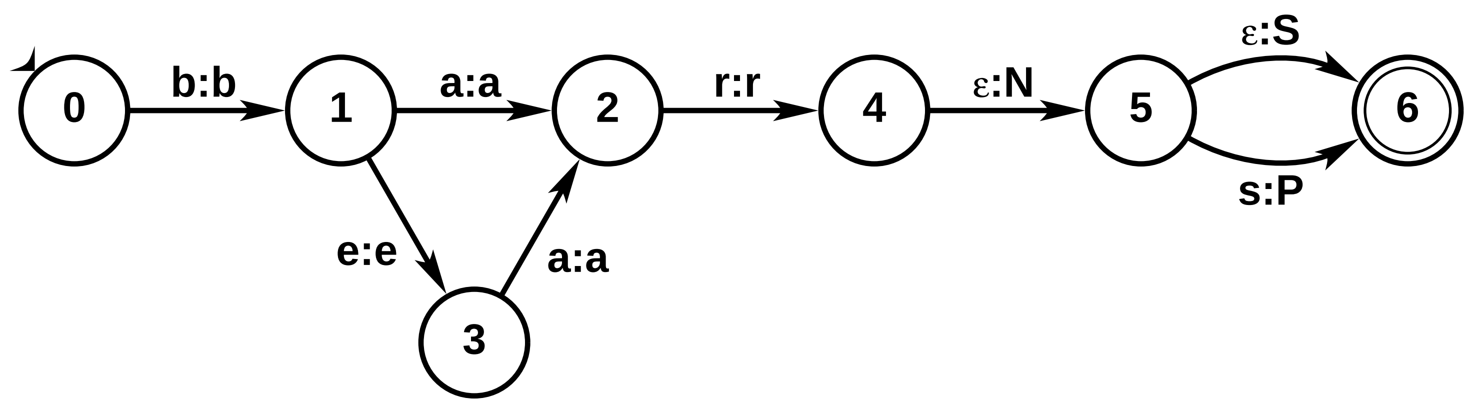
2. *Form a queue* with the rest of the path

3. *Remove* unreachable states;

4. *Check (backwards)* clones and queues against intact states.

We'll see this with an example.

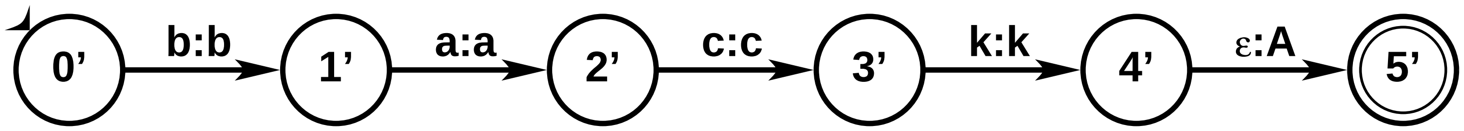
Adding an entry /2



The initial transducer accepts (bar··:barNS), (bar·s:barNP), (bear··:bearNS), (bear·s:bearNP).

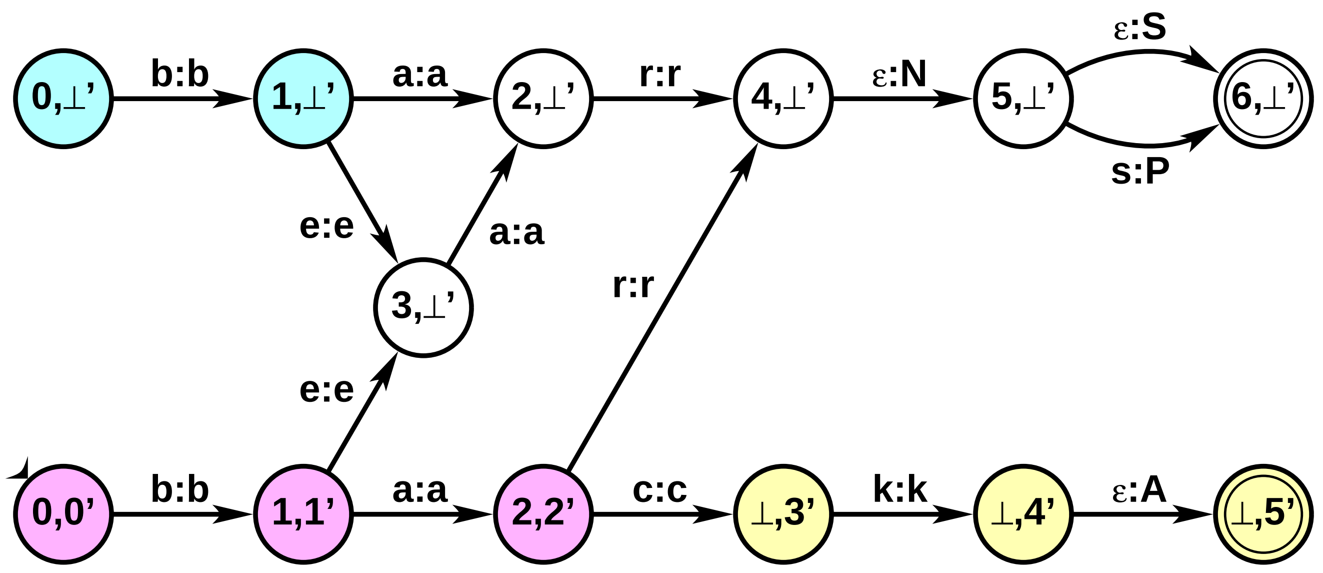
Adding an entry /3

We'll add the entry (`back·:backA`), an adverb.



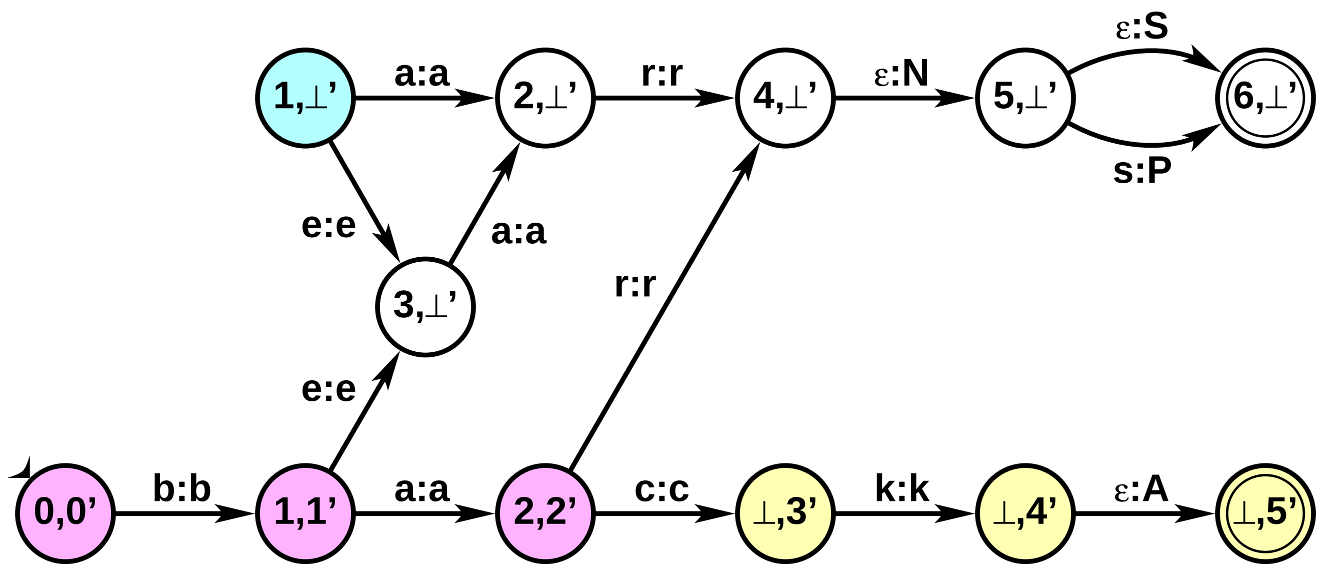
Adding an entry /4

The result after creating clones and queues: new initial state is cloned.



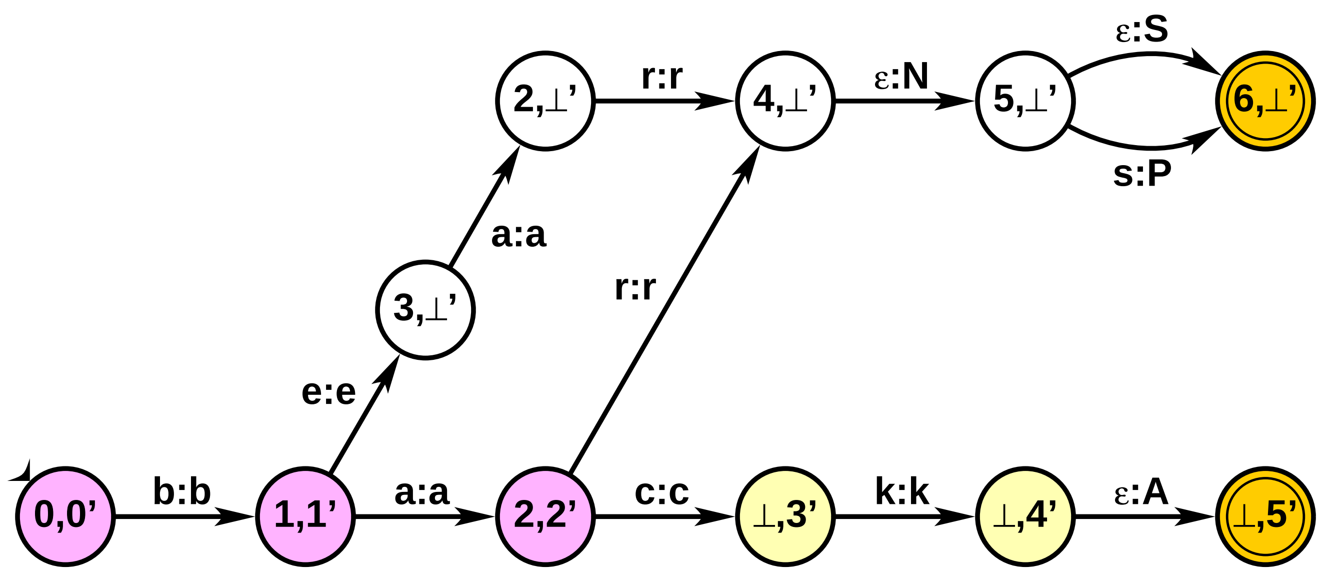
Adding an entry /4

After removal of unreachable state $(0, \perp')$.



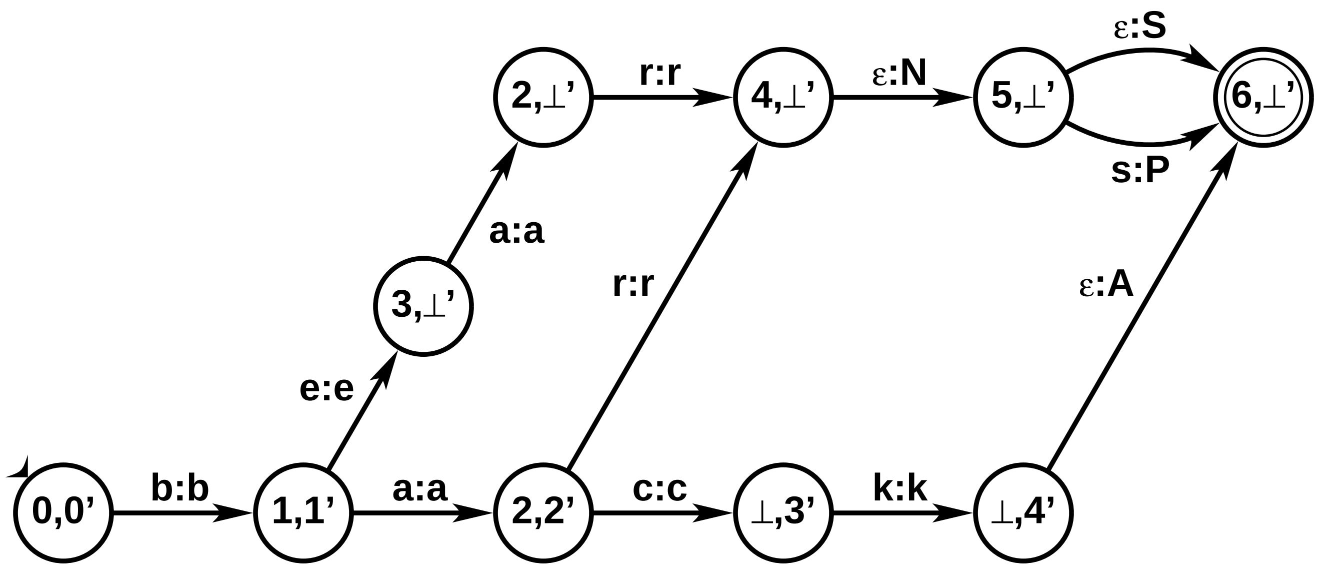
Adding an entry /5

After removal of unreachable state $(1, \perp')$.



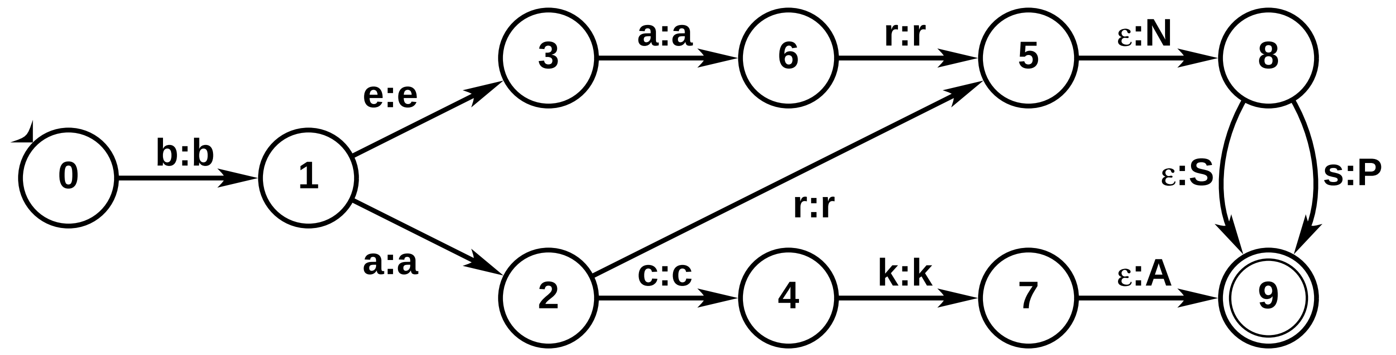
Adding an entry /6

After merging queue state $(\perp, 5')$ with original state $(6, \perp')$ no more equivalent clones and queues remain.



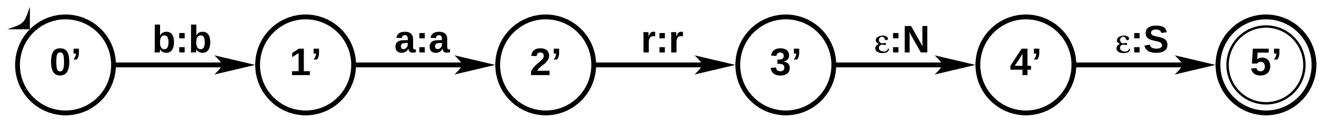
Adding an entry /7

We get a minimal transducer (let me rearrange it a bit).



Removing an entry /1

Now, we will remove entry (`bar··:barNS`). Here's the entry:



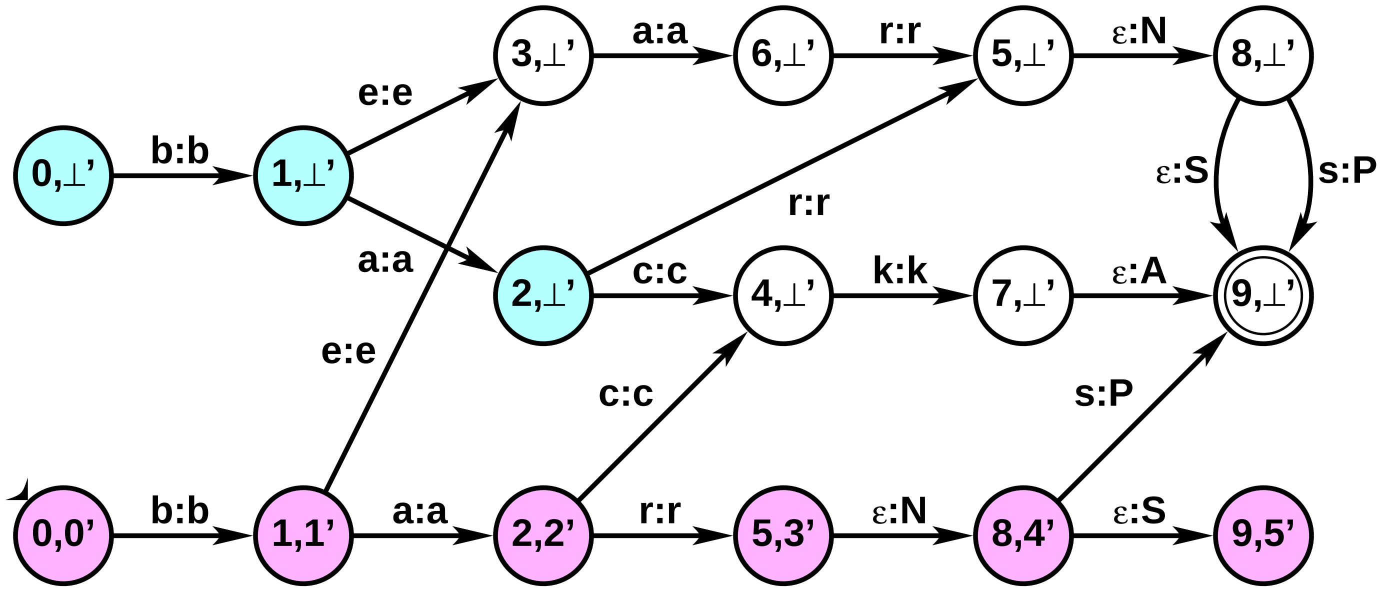
Small changes in the algorithm:

No queue states form

The last cloned state is made nonaccepting.

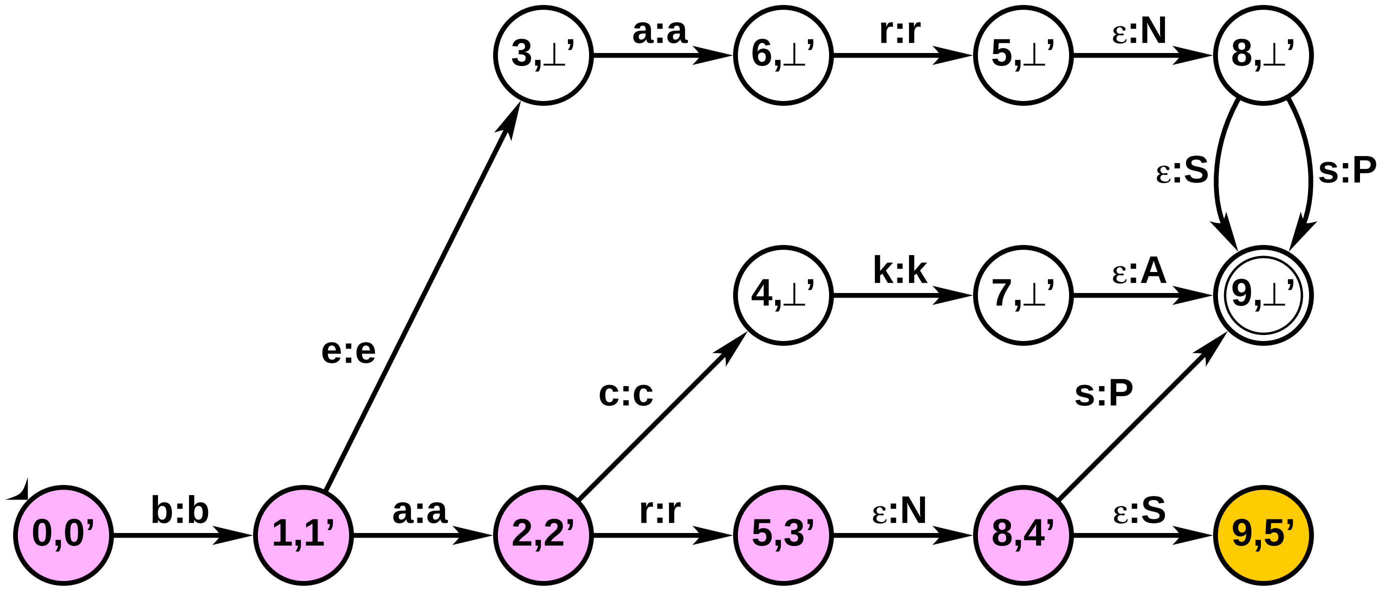
Removing an entry /2

The result after creating clones: new initial state is cloned. States $(0, \perp')$, $(1, \perp')$, and $(2, \perp')$ are unreachable.



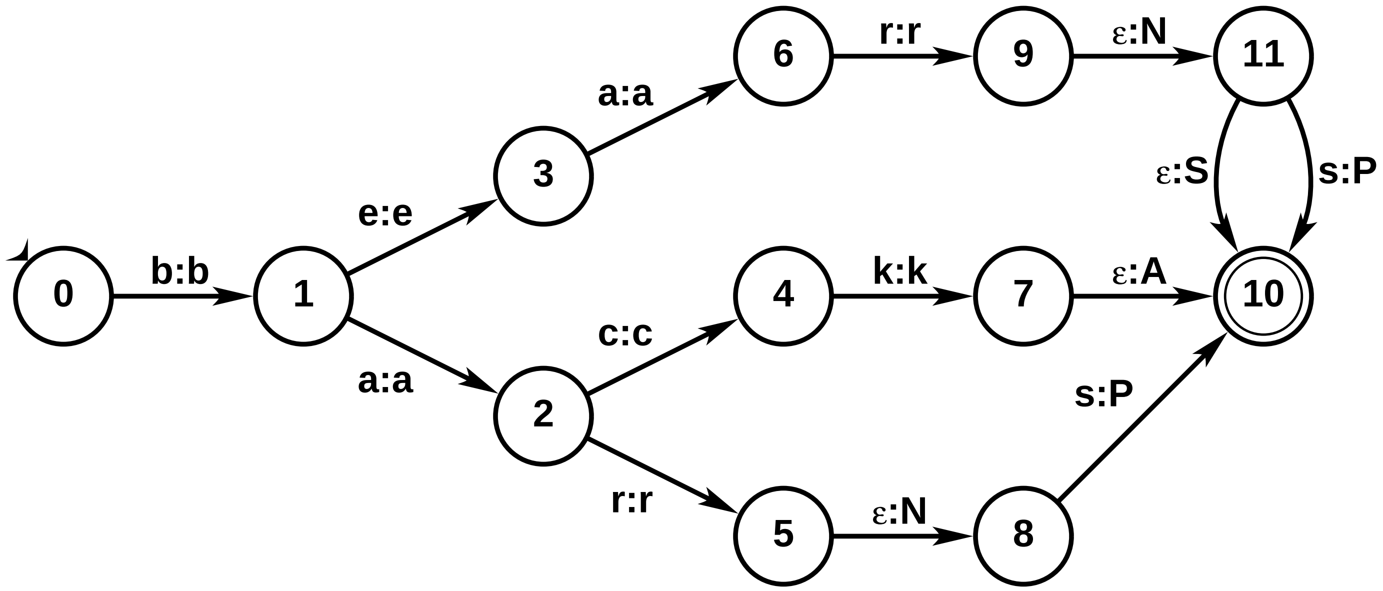
Removing an entry /3

The state $(9, 5')$ is a garbage state and may be removed. No cloned states are found to be equivalent to original states.



Removing an entry /4

The final transducer, rearranged and renumbered.



Comments on adding and removing

If original transducer is large, changes affect few states.

No need to recheck intact original states for minimality.

Main cost of adding a transition t: checking queue and clones against original states: $O(|Q||t|)$.

Works on transducers with cycles (Carrasco and Forcada 2002).

The original surface form–lexical form alignment is kept (linguistically motivated alignments may be shared $\rightarrow$ compactness).

Concluding remarks

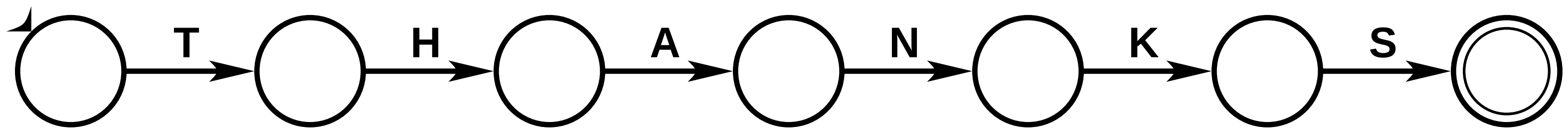
DALTs implement morphological analysers . . .

. . . which tokenize their input as they analyse it.

They use explicit SF–LF alignments.

May also be used for lexical transfer and generation (without lookahead)

An algorithm allows entries to be added to (and removed from) DALTs while keeping them minimal.


T
H
A
N
K
S