DFA hyper-minimisation

Paweł Gawrychowski \(^1\) Artur Jeż \(^1\)

Institute of Computer Science, University of Wrocław

November 24, 2009
DFA minimisation

Definition
DFA: \( \langle Q, \Sigma, \delta, q_0, F \rangle \), where \( \delta : Q \times \Sigma \rightarrow Q \). DFA is minimal, if it has the minimal number of states among automata recognising \( L(M) \).
DFA minimisation

Definition

DFA: \(\langle Q, \Sigma, \delta, q_0, F \rangle\), where \(\delta : Q \times \Sigma \rightarrow Q\). DFA is **minimal**, if it has the minimal number of states among automata recognising \(L(M)\).

- unique with this property
- calculated using \(\equiv_L\):

  \[
  w \equiv w' \text{ if and only if } \forall w'' \ ww'' \in L \iff w'w'' \in L
  \]

- equivalence classes correspond to
  - states of the minimal automaton
  - partition of states of \(M\)
DFA minimisation

Definition

DFA: \( \langle Q, \Sigma, \delta, q_0, F \rangle \), where \( \delta : Q \times \Sigma \rightarrow Q \). DFA is minimal, if it has the minimal number of states among automata recognising \( L(M) \).

- unique with this property
- calculated using \( \equiv_L \):
  \[
  w \equiv w' \text{ if and only if } \forall w'' \quad ww'' \in L \iff w'w'' \in L
  \]
- equivalence classes correspond to
  - states of the minimal automaton
  - partition of states of \( M \)
- Hopcroft’s algorithm: \( \mathcal{O}(n \log n) \); refines the partition of states
$f$-equivalence and hyper-minimisation

**Definition ($f$-equivalent)**

$L \sim L' \iff$ they differ on finite amount of words.

Extend the definition to automata.
Definition (f-equivalent)

$L \sim L' \iff$ they differ on finite amount of words.

Extend the definition to automata.

Definition (A. Badr, V. Geffert, I. Shipman)

$M$ is hyper-minimal, if it has the minimal number of states among the $f$-equivalent automata. (Not unique)
Definition ($f$-equivalent)

$L \sim L' \iff$ they differ on finite amount of words.

Extend the definition to automata.

Definition (A. Badr, V. Geffert, I. Shipman)

$M$ is hyper-minimal, if it has the minimal number of states among the $f$-equivalent automata. (Not unique)

Remark

For fixed $L$ we extend $\sim$ to words: $w \sim w' \iff w^{-1}L \sim w'{-1}L$

For fixed automata $M$ we extend $\sim$ to states: $q \sim q' \iff L(q) \sim L(q')$

(where $L(q)$ is the language recognised starting from $q$).
Approach

Idea

We want a relation on words, such that equivalence classes are states of a hyper-minimal automaton, $\sim$ is a natural candidate.
Approach

Idea

*We want a relation on words, such that equivalence classes are states of a hyper-minimal automaton, \( \sim \) is a natural candidate.*

- Classes of \( \sim \) are groups of classes of \( \equiv \).
- We cannot greedily merge those groups: \( w : \delta(q_0, w) = q_1 : wL(q_1) \) changes to \( wL(q_3) \neq wL(q_1) \). Infinitely many such \( w \) — problem!

- No problem occurs if there are only finitely many such \( w \).
Approach

Idea

We want a relation on words, such that equivalence classes are states of a hyper-minimal automaton, $\sim$ is a natural candidate.

- Classes of $\sim$ are groups of classes of $\equiv$.
- We cannot greedily merge those groups: $w : \delta(q_0, w) = q_1 : wL(q_1)$ changes to $wL(q_3) \neq wL(q_1)$. Infinitely many such $w$ — problem!

No problem occurs if there are only finitely many such $w$.

Definition

State $q$ is in preamble if $\{w : \delta(q_0, w) = q\}$ is finite. In kernel otherwise.
Heuristic

Definition (state merging)
Heuristic

Greedily merge q to p whenever
- \( q \equiv p \) or
- \( q \sim p \) and q is in the preamble
Heuristic

Definition (state merging)

Heuristic

Greedily merge $q$ to $p$ whenever

- $q \equiv p$ or
- $q \sim p$ and $q$ is in the preamble and there is no path from $p$ to $q$
Heuristic

Definition (state merging)

Heuristic

Greedily merge $q$ to $p$ whenever
- $q \equiv p$ or
- $q \sim p$ and $q$ is in the preamble and there is no path from $p$ to $q$

Theorem (A. Badr, V. Geffert, I. Shipman)

The heuristic is proper, i.e. it results in hyper-minimal automaton $f$-equivalent to the input one.
Data structures

**Definition (Operational definition of ~)**

- $D^M(q, q')$ if $q = q'$ or,
- $D^M(q, q')$ if for all $a \in \Sigma$ $D^M(\delta_M(q, a), \delta_M(q', a))$.

**Lemma**

*If the automaton $M$ is minimised the $D$ coincides with ~.*
Data structures

Definition (Operational definition of $\sim$)

- $D^M(q, q')$ if $q = q'$ or,
- $D^M(q, q')$ if for all $a \in \Sigma$ $D^M(\delta_M(q, a), \delta_M(q', a))$.

Lemma

If the automaton $M$ is minimised the $D$ coincides with $\sim$.

We need a dictionary structure supporting

- query, if there are $q, q'$ such that $(\delta(q, 0), \delta(q, 1)) = (\delta(q', 0), \delta(q', 1))$
- when $q$ is merged to $q'$, fast update of $\delta$
Data structures

**Definition (Operational definition of ⊳)**
- \( D^M(q, q') \) if \( q = q' \) or,
- \( D^M(q, q') \) if for all \( a \in \Sigma \) \( D^M(\delta_M(q, a), \delta_M(q', a)) \).

**Lemma**

*If the automaton \( M \) is minimised the \( D \) coincides with \( ⊳ \).*

We need a dictionary structure supporting

- query, if there are \( q, q' \) such that
  \((\delta(q, 0), \delta(q, 1)) = (\delta(q', 0), \delta(q', 1))\)
- when \( q \) is merged to \( q' \), fast update of \( \delta \)

- Deterministic — tree: the path from root to the leave is \((\delta(q, 0), \delta(q, 1))\)
- Randomised — hashing
Algorithm

Calculating relation $D$ over states

- identify $q, q'$ with the same successors
- delete the one with less predecessors
- update the predecessors

Using $D$ greedily merge states.
Calculating relation $D$ over states

- identify $q, q'$ with the same successors
- delete the one with less predecessors
- update the predecessors

Using $D$ greedily merge states.

Running time: $\mathcal{O}(n \log n)$ times insertion time

- insertion time:
  - deterministic: $\mathcal{O}(\log n)$
  - randomised $\mathcal{O}(1)$
Remarks and Questions

- $|\Sigma|$ has linear impact on the running time
- for partial $\delta$, running time $O(|\delta| \log^2 n)$ can be obtained
Remarks and Questions

- $|\Sigma|$ has linear impact on the running time
- for partial $\delta$, running time $O(|\delta| \log^2 n)$ can be obtained
- Done independantly by Markus Holzer and Andreas Maletti, CIAA 2009.
Remarks and Questions

- $|\Sigma|$ has linear impact on the running time
- for partial $\delta$, running time $O(|\delta| \log^2 n)$ can be obtained
- Done independantly by Markus Holzer and Andreas Maletti, CIAA 2009.

- Deterministic running time $O(n \log n)$?
- Checking the $f$-equivalence of two automata is faster?
Refinement

Definition (distance between languages)

\[ d(L, L') = \begin{cases} 
\max\{|u| : u \in L(w) \Delta L(w')\} + 1 & \text{if } L \neq L' \\
0 & \text{if } L = L' 
\end{cases} \]

Definition (\(k\)-f-equivalence)

\( L \sim_k L' \iff d(L, L') \leq k \)

Definition

\( M \) is \( k \)-minimal if it has the least number of states among the \( \sim_k \) automata.
Refinement

Definition (distance between languages)

\[ d(L, L') = \begin{cases} \max \{|u| : u \in L(w) \Delta L(w')\} + 1 & \text{if } L \neq L' , \\ 0 & \text{if } L = L' . \end{cases} \]

Definition (k-f-equivalence)

\[ L \sim_k L' \iff d(L, L') \leq k \]

Definition

\( M \) is \( k \)-minimal if it has the least number of states among the \( \sim_k \) automata.

Remark

Algorithm is similar, but some theoretical work is to be done.
Approach

Idea

- Suppose there are $w_1, w_2$ with respective $q_1, q_2$ and $L(w_1), L(w_2)$.
- We merge state $q_1$ to $q_2$
- Intuitively, $w_1 L(w_1)$ changes to $w_1 L(w_2)$
- If $L(w_1) \neq L(w_2)$ we want
  \[ k \geq d(w_1 L(w_1); w_1 L(w_2)) = |w_1| + d(L(w_1), L(w_2)) \]
Approach

Idea

- Suppose there are $w_1, w_2$ with respective $q_1, q_2$ and $L(w_1), L(w_2)$.
- We merge state $q_1$ to $q_2$
- Intuitively, $w_1 L(w_1)$ changes to $w_1 L(w_2)$
- If $L(w_1) \neq L(w_2)$ we want
  \[ k \geq d(w_1 L(w_1); w_1 L(w_2)) = |w_1| + d(L(w_1), L(w_2)) \]

Definition

\[ w_1 \sim_k w_2 \iff L(w_1) = L(w_2) \text{ or } \min(|w_1|, |w_2|) + d(L(w_1), L(w_2)) \leq k \]

Remark

This is not an equivalence relation: it is not transitive.
Approach

Idea

- Suppose there are \( w_1, w_2 \) with respective \( q_1, q_2 \) and \( L(w_1), L(w_2) \).
- We merge state \( q_1 \) to \( q_2 \)
- Intuitively, \( w_1L(w_1) \) changes to \( w_1L(w_2) \)
- If \( L(w_1) \neq L(w_2) \) we want
  \[ k \geq d(w_1L(w_1); w_1L(w_2)) = |w_1| + d(L(w_1), L(w_2)) \]

Definition

\[ w_1\sim_k w_2 \iff L(w_1) = L(w_2) \text{ or } \min(|w_1|, |w_2|) + d(L(w_1), L(w_2)) \leq k \]

Remark

This is not an equivalence relation: it is not transitive.

Lemma

If \( \{w_i\}_{i=1}^\ell \) satisfy \( w_i \not\sim_k w_j \) then every automaton \( k\text{-f-equivalent to } M \) has at least \( \ell \) states.
Adjusting the relation

**Definition (Expanding for states)**

For $q$ define its **representative word** $\text{word}(w)$: the longest word $w$ such that $\delta(q_0, w) = q$. (take any word of length $k + 1$ if this is badly defined).

$q \sim_k q' \iff \text{word}(q) \sim_k \text{word}(q')$
Adjusting the relation

**Definition (Expanding for states)**

For \( q \) define its **representative word** \( \text{word}(w) \): the longest word \( w \) such that \( \delta(q_0, w) = q \). (take any word of length \( k + 1 \) if this is badly defined).

\[ q \sim_k q' \iff \text{word}(q) \sim_k \text{word}(q') \]

Improving \( \sim_k \) to an equivalence relation \( \approx_k \) satisfying:

- \( w \approx_k w' \) implies \( w \sim_k w' \)
- equivalence class of \( \approx_k \) has a representative \( \text{Rep} \)
- \( w \not\approx_k w' \) implies \( \text{Rep}(w) \not\sim_k \text{Rep}(w') \)
Adjusting the relation

**Definition (Expanding for states)**

For $q$ define its representative word $\text{word}(w)$: the longest word $w$ such that $\delta(q_0, w) = q$. (take any word of length $k + 1$ if this is badly defined).

$q \sim_k q' \iff \text{word}(q) \sim_k \text{word}(q')$

**Improving $\sim_k$ to an equivalence relation $\approx_k$ satisfying:**

- $w \approx_k w'$ implies $w \sim_k w'$
- equivalence class of $\approx_k$ has a representative $\text{Rep}$
- $w \not\approx_k w'$ implies $\text{Rep}(w) \not\sim_k \text{Rep}(w')$

**Lemma**

$\approx_k$ can be calculated out of $\sim_k$ in a greedy fashion (using word)
$k$-minimal Automata

Definition ($k$-minimal automata $N$)

- $Q_N = \{ \langle w \rangle : w = \text{Rep}(w) \}$
- $\delta_N(\langle w \rangle, a) = \text{Rep}(wa)$
**k-minimal Automata**

**Definition (k-minimal automata \( N \))**

- \( Q_N = \{ \langle w \rangle : w = \text{Rep}(w) \} \)
- \( \delta_N(\langle w \rangle, a) = \text{Rep}(wa) \)

**Lemma**

\( N \sim_k M \)

**Proof.**

- for \( \text{Rep}(q) \) s.t. \(|\text{Rep}(q)| > k\) transition structure does not change.
- for other states by backward induction we show that \( d(L_M(q), L_N(\text{Rep}(q))) \leq k \)

It is \( k \)-minimal by previous lemma.

**Remark**

Algorithm — refinement of the previous one
Questions

- Deterministic running time $O(n \log n)$?
- Checking the $k$-$f$-equivalence of two automata is faster?