Hyper-minimisation of weighted finite automata

Daniel Quernheim and Andreas Maletti

Institute for Natural Language Processing, University of Stuttgart
<first.last>@ims.uni-stuttgart.de

August 18, 2011

Outline

Unweighted case

Weighted case

Conclusion

Outline

Unweighted case

Weighted case

Conclusion

Minimisation

Problem

given DFA, return

- equivalent DFA such that
- all equivalent DFA are larger

Theorem (Hopcroft 1971)

DFA minimisation can be done in time $O(n \log n)$

- n*: number of states*

Minimisation

Problem

given DFA, return

- equivalent DFA
- minimal

Theorem (Hopcroft 1971)

DFA minimisation can be done in time $O(n \log n)$

- n*: number of states*

Hyper-minimisation

Definition

DFA A, B almost equivalent if $L(A)$ and $L(B)$ have finite difference

Problem [Badr et al. 2009]

given DFA, return

- almost equivalent DFA such that
- all almost equivalent DFA are larger

Theorem (Holzer, $\sim$ 2009, Gawrychowsky, Jeż 2009)

DFA hyper-minimisation can be done in time $O(n \log n)$

Hyper-minimisation

Definition

DFA A, B almost equivalent if $L(A)$ and $L(B)$ have finite difference

Problem [Badr et al. 2009]

given DFA, return

- almost equivalent DFA
- hyper-minimal

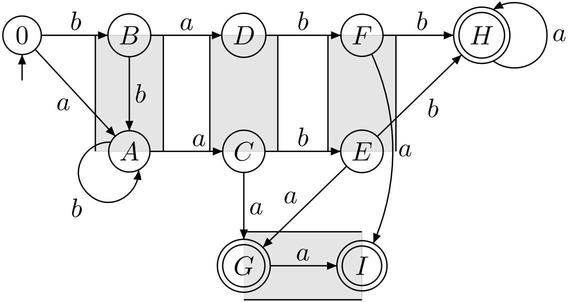
Theorem (Holzer, $\sim$ 2009, Gawrychowsky, Jeż 2009)

DFA hyper-minimisation can be done in time $O(n \log n)$

Almost equivalence

- States are almost equivalent if their right languages differ finitely

$$\overrightarrow{q} = \{w \in \Sigma^* \mid \delta(q, w) \in F\}$$

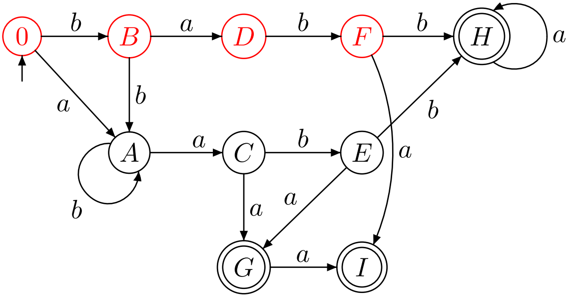


Kernel and preamble states

Definition

- preamble state: finite left language
- kernel state: infinite left language

$$\overleftarrow{q} = \{w \in \Sigma^* \mid \delta(q_0, w) = q\}$$



Structural characterisation

Theorem (Badr et al. 2009)

DFA is hyper-minimal iff

- *minimal*
- *no preamble state is almost equivalent to another state*

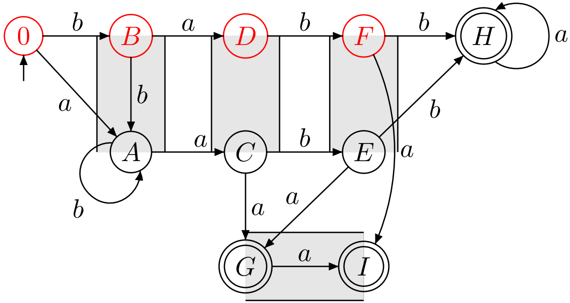
Structural characterisation

Theorem (Badr et al. 2009)

DFA is hyper-minimal iff

- *minimal*
- *no preamble state is almost equivalent to another state*

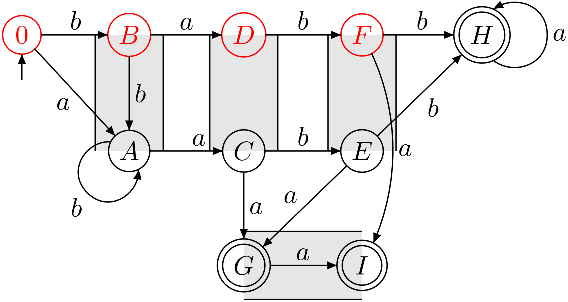
Example (Not hyper-minimal)



Hyper-minimisation

preamble	almost equivalent
B	A
D	C
F	E

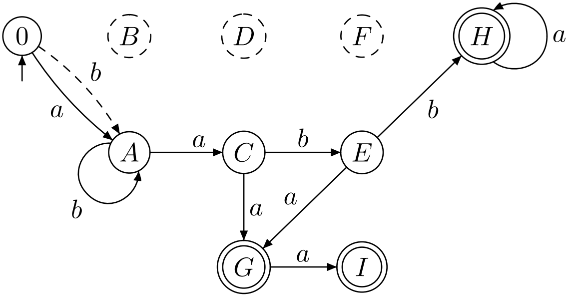
Table: Merges



Hyper-minimisation

preamble	almost equivalent
B	A
D	C
F	E

Table: Merges



Motivation

Application

In natural language processing

- Automata often deterministic for efficient evaluation
- DFA tend to be very big $\rightarrow$ good reduction potential
- Often approximative data $\rightarrow$ lossy compression ok

Motivation

Application

In natural language processing

- Automata often deterministic for efficient evaluation
- DFA tend to be very big $\rightarrow$ good reduction potential
- Often approximative data $\rightarrow$ lossy compression ok
- but DFA are weighted

Conclusion

Let's do hyper-minimisation for weighted DFA!

Outline

Unweighted case

Weighted case

Conclusion

Weight structure

General approach

works for semifields (semirings with mult. inverses)

Weight structure

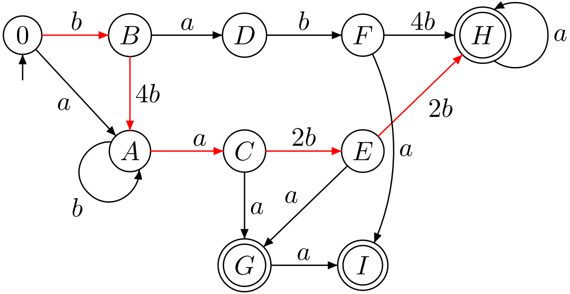
General approach

works for semifields (semirings with mult. inverses)

Presentation

here we use the field of real numbers

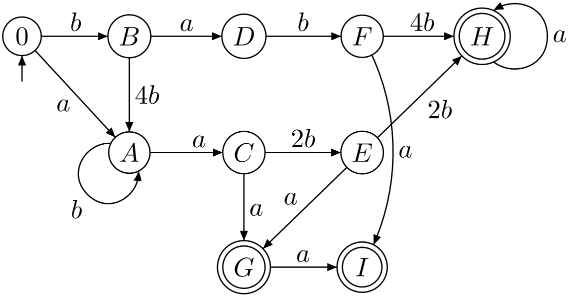
Example (WDFA)



weight of $bbabb$
is $1 \cdot 4 \cdot 1 \cdot 2 \cdot 2 = 16$

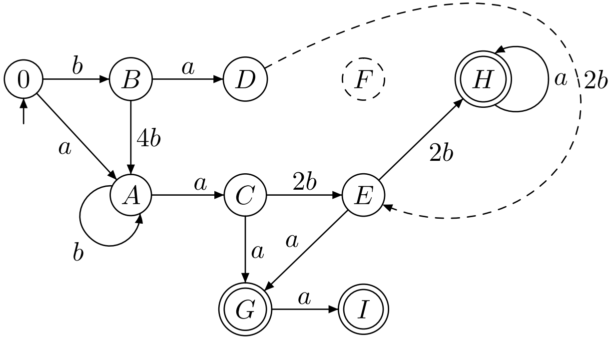
weight of ab is 0

Weighted merge



Weighted merge of F into E with factor 2

Weighted merge



Weighted merge of F into E with factor 2

Almost equivalence

Definition

Two WDFA A, B are almost equivalent if

$$A(w) \neq B(w) \quad \text{for only finitely many } w \in \Sigma^*$$

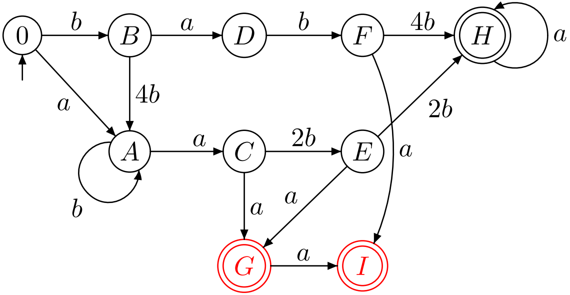
Definition

Two states p, q are almost equivalent if there is $k \in \mathbb{R} \setminus \{0\}$ such that

$$\overrightarrow{p}(w) \neq k \cdot \overrightarrow{q}(w) \quad \text{for only finitely many } w \in \Sigma^*$$

[Borchardt: The Myhill-Nerode theorem for recognizable tree series. DLT 2003]

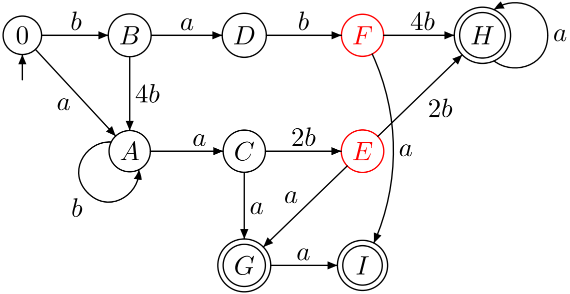
Example



	G	I
ϵ	1	1
a	1	0

- G and I almost equivalent

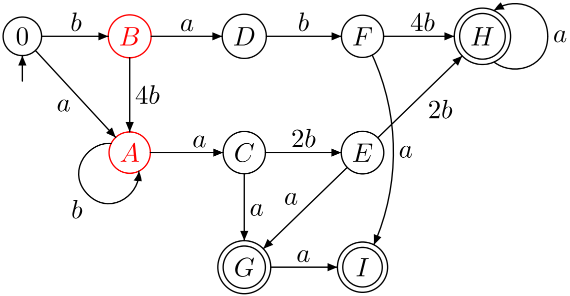
Example



	E	F
a	1	1
aa	1	0
b	2	4
ba	2	4
baa	2	4
$baaa$	2	4
$\ldots$		

- G and I almost equivalent
- E and F almost equivalent (with factor 2)

Example



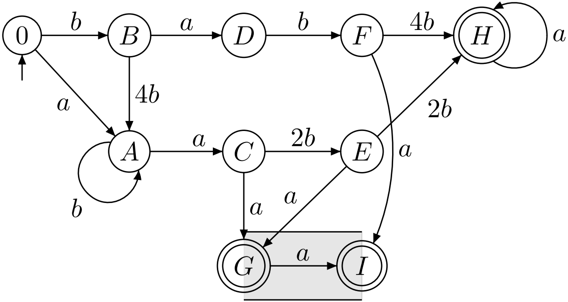
	A	B
abb	4	4
$abba$	4	4
$abbaa$	4	4
...		
$babb$	4	16
$babba$	4	16
$babbaa$	4	16
...		

- G and I almost equivalent
- E and F almost equivalent (with factor 2)
- A and B not almost equivalent!

Finding almost equivalent states

Definition

- co-kernel state: infinite right language
- co-preamble state: finite right language



Signature standardisation

Definition

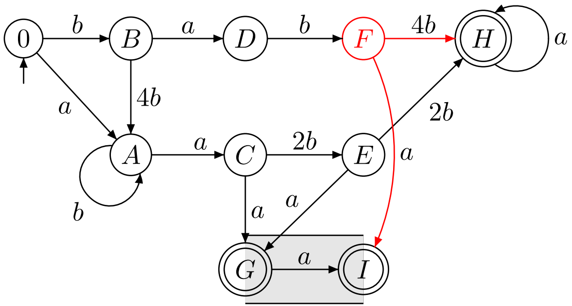
signature of q: $\big(\langle \delta(q,\sigma), c(q,\sigma)\rangle\big)_{\sigma\in\Sigma}$

Standardisation

- select minimal $\sigma_0 \in \Sigma$ such that $\delta(q,\sigma_0)$ is co-kernel
- adjust transition weights:

$$c'(q,\sigma) = \begin{cases} \frac{c(q,\sigma)}{c(q,\sigma_0)} & \text{if } \delta(q,\sigma) \text{ is co-kernel} \\ 1 & \text{otherwise} \end{cases}$$

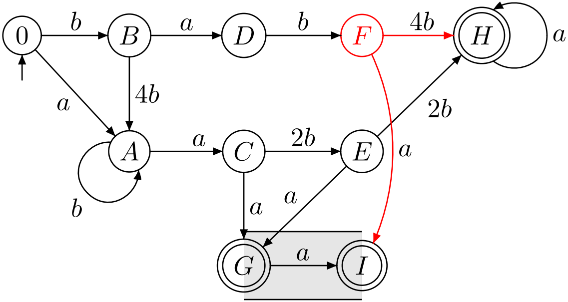
Signature standardisation



Standardised signature of F

- $F \xrightarrow{a} I$ leads to co-preamble

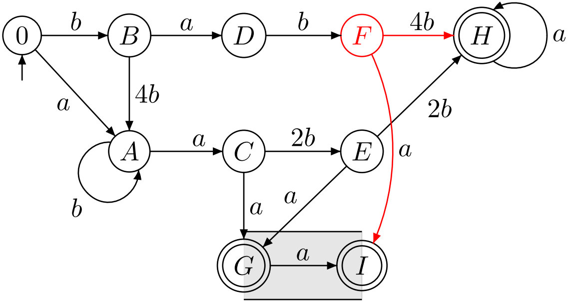
Signature standardisation



Standardised signature of F

- $F \stackrel{a}{\to} I$ leads to co-preamble $\quad \to \quad c'(F, a) = 1$

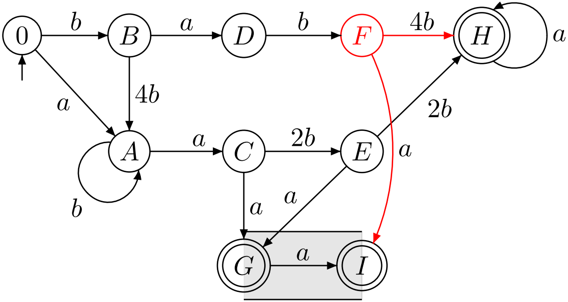
Signature standardisation



Standardised signature of F

- $F \xrightarrow{a} I$ leads to co-preamble $\quad \rightarrow \quad c'(F, a) = 1$
- $F \xrightarrow{4b} H$ leads to co-kernel

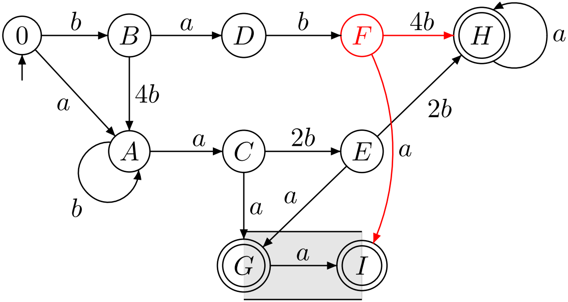
Signature standardisation



Standardised signature of F

- $F \stackrel{a}{\to} I$ leads to co-preamble $\quad \to \quad c'(F, a) = 1$
- $F \stackrel{4b}{\to} H$ leads to co-kernel $\quad \to \quad c'(F, b) = \frac{4}{4} = 1$

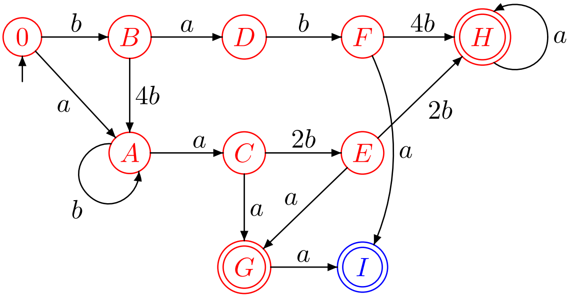
Signature standardisation



Standardised signature of F

- $F \xrightarrow{a} I$ leads to co-preamble $\rightarrow$ $c'(F, a) = 1$
- $F \xrightarrow{4b} H$ leads to co-kernel $\rightarrow$ $c'(F, b) = \frac{4}{4} = 1$
- Standardised signature $(\langle I, 1\rangle, \langle H, 1\rangle)$

Finding almost equivalent states



Signature map:

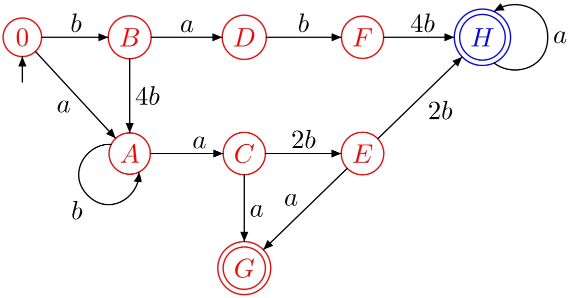
$\langle \bot, 1 \rangle, \langle \bot, 1 \rangle$	$\bot$

Blocks:

- $\{\bot, I\}$

- $\mathrm{sig}(I) = (\langle \bot, 1 \rangle, \langle \bot, 1 \rangle)$ in map!
- add I to block of $\bot$ (and merge)

Finding almost equivalent states



Signature map:

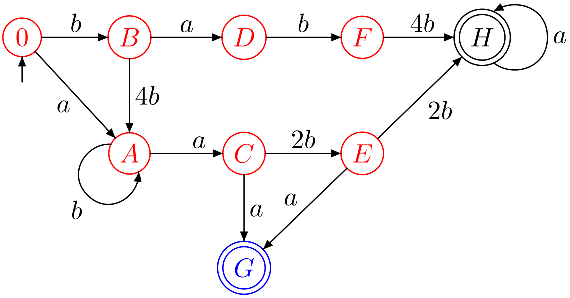
$\langle \bot, 1\rangle, \langle \bot, 1\rangle$	$\bot$
$\langle H, 1\rangle, \langle \bot, 1\rangle$	H

Blocks:

- $\{\bot, I\}$

- $\text{sig}(H) = (\langle H, 1\rangle, \langle \bot, 1\rangle)$
- add to map

Finding almost equivalent states



Signature map:

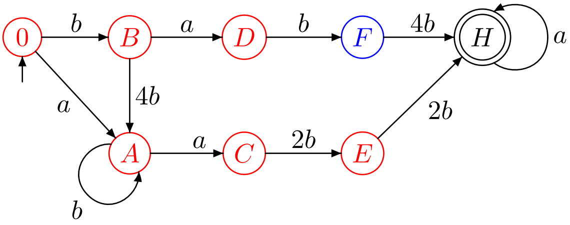
$\langle \bot, 1\rangle, \langle \bot, 1\rangle$	$\bot$
$\langle H, 1\rangle, \langle \bot, 1\rangle$	H

Blocks:

- $\{\bot, I, G\}$

- $\text{sig}(G) = (\langle \bot, 1\rangle, \langle \bot, 1\rangle)$ in map!
- add G to $\bot$ (and merge)

Finding almost equivalent states



Signature map:

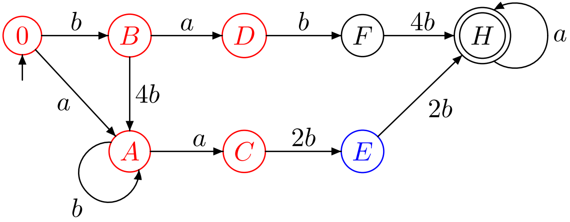
$\langle\bot, 1\rangle, \langle\bot, 1\rangle$	$\bot$
$\langle H, 1\rangle, \langle\bot, 1\rangle$	H
$\langle\bot, 1\rangle, \langle H, 1\rangle$	F

Blocks:

- $\{\bot, I, G\}$

- $\mathrm{sig}(F) = (\langle\bot, 1\rangle, \langle H, 1\rangle)$
- add to map

Finding almost equivalent states



Signature map:

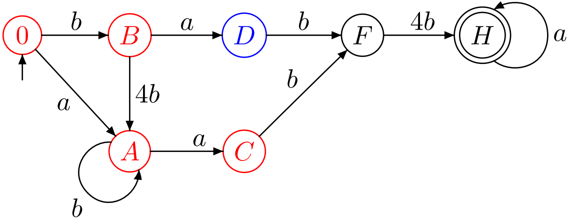
$\langle \bot, 1\rangle, \langle \bot, 1\rangle$	$\bot$
$\langle H, 1\rangle, \langle \bot, 1\rangle$	H
$\langle \bot, 1\rangle, \langle H, 1\rangle$	F

Blocks:

- $\{\bot, I, G\}$
- $\{F, E\}$

- $\mathrm{sig}(E) = (\langle \bot, 1\rangle, \langle H, 1\rangle)$ in map!
- add E to F (and merge with factor $\frac{1}{2}$)

Finding almost equivalent states



Signature map:

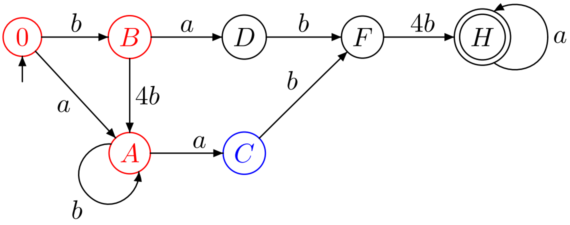
$\langle\bot,1\rangle,\langle\bot,1\rangle$	$\bot$
$\langle H,1\rangle,\langle\bot,1\rangle$	H
$\langle\bot,1\rangle,\langle H,1\rangle$	F
$\langle\bot,1\rangle,\langle F,1\rangle$	D

Blocks:

- $\{\bot, I, G\}$
- $\{F, E\}$

- $\text{sig}(D) = (\langle\bot,1\rangle,\langle F,1\rangle)$
- add to map

Finding almost equivalent states



Signature map:

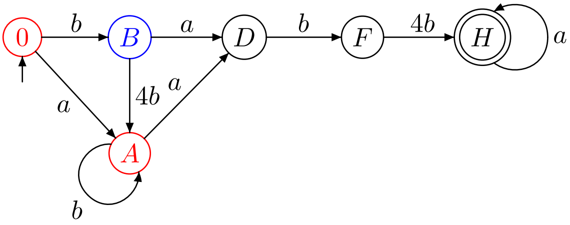
$\langle\bot, 1\rangle, \langle\bot, 1\rangle$	$\bot$
$\langle H, 1\rangle, \langle\bot, 1\rangle$	H
$\langle\bot, 1\rangle, \langle H, 1\rangle$	F
$\langle\bot, 1\rangle, \langle F, 1\rangle$	D

Blocks:

- $\{\bot, I, G\}$
- $\{F, E\}$
- $\{D, C\}$

- $\mathrm{sig}(C) = (\langle\bot, 1\rangle, \langle F, 1\rangle)$ in map!
- Add C to D (and merge)

Finding almost equivalent states



Signature map:

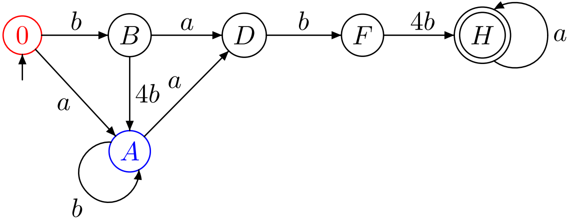
$\langle\bot,1\rangle,\langle\bot,1\rangle$	$\bot$
$\langle H,1\rangle,\langle\bot,1\rangle$	H
$\langle\bot,1\rangle,\langle H,1\rangle$	F
$\langle\bot,1\rangle,\langle F,1\rangle$	D
$\langle D,1\rangle,\langle A,4\rangle$	B

Blocks:

- $\{\bot, I, G\}$
- $\{F, E\}$
- $\{D, C\}$

- $\mathrm{sig}(B) = (\langle D,1\rangle,\langle A,4\rangle)$
- add to map

Finding almost equivalent states



Signature map:

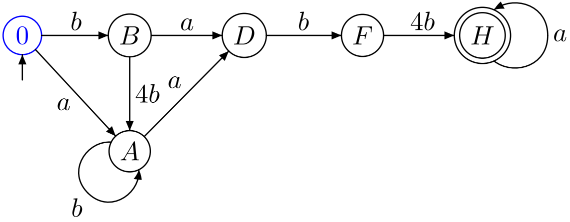
$\langle\bot,1\rangle,\langle\bot,1\rangle$	$\bot$
$\langle H,1\rangle,\langle\bot,1\rangle$	H
$\langle\bot,1\rangle,\langle H,1\rangle$	F
$\langle\bot,1\rangle,\langle F,1\rangle$	D
$\langle D,1\rangle,\langle A,4\rangle$	B
$\langle D,1\rangle,\langle A,1\rangle$	A

Blocks:

- $\{\bot, I, G\}$
- $\{F, E\}$
- $\{D, C\}$

- $\text{sig}(A) = (\langle D,1\rangle,\langle A,1\rangle)$
- add to map

Finding almost equivalent states



Signature map:

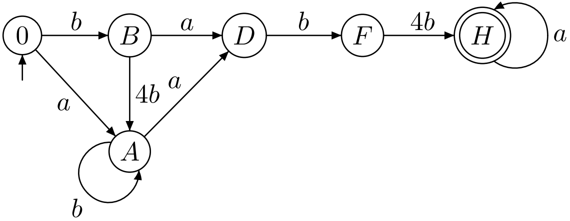
$\langle \bot, 1\rangle, \langle \bot, 1\rangle$	$\bot$
$\langle H, 1\rangle, \langle \bot, 1\rangle$	H
$\langle \bot, 1\rangle, \langle H, 1\rangle$	F
$\langle \bot, 1\rangle, \langle F, 1\rangle$	D
$\langle D, 1\rangle, \langle A, 4\rangle$	B
$\langle D, 1\rangle, \langle A, 1\rangle$	A
$\langle A, 1\rangle, \langle B, 1\rangle$	0

Blocks:

- $\{\bot, I, G\}$
- $\{F, E\}$
- $\{D, C\}$

- $\mathrm{sig}(0) = (\langle A, 1\rangle, \langle B, 1\rangle)$
- add to map

Finding almost equivalent states



Signature map:

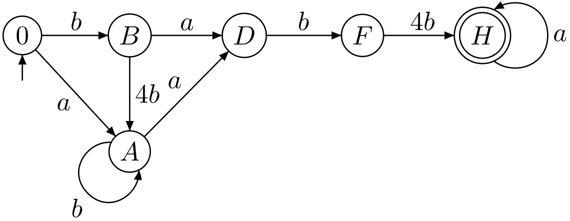
$\langle\bot, 1\rangle, \langle\bot, 1\rangle$	$\bot$
$\langle H, 1\rangle, \langle\bot, 1\rangle$	H
$\langle\bot, 1\rangle, \langle H, 1\rangle$	F
$\langle\bot, 1\rangle, \langle F, 1\rangle$	D
$\langle D, 1\rangle, \langle A, 4\rangle$	B
$\langle D, 1\rangle, \langle A, 1\rangle$	A
$\langle A, 1\rangle, \langle B, 1\rangle$	0

Blocks:

- $\{\bot, I, G\}$
- $\{F, E\}$
- $\{D, C\}$

- Blocks represent almost equivalence
- Scaling factors used in merges

Finding almost equivalent states



Signature map:

$\langle \bot, 1\rangle, \langle \bot, 1\rangle$	$\bot$
$\langle H, 1\rangle, \langle \bot, 1\rangle$	H
$\langle \bot, 1\rangle, \langle H, 1\rangle$	F
$\langle \bot, 1\rangle, \langle F, 1\rangle$	D
$\langle D, 1\rangle, \langle A, 4\rangle$	B
$\langle D, 1\rangle, \langle A, 1\rangle$	A
$\langle A, 1\rangle, \langle B, 1\rangle$	0

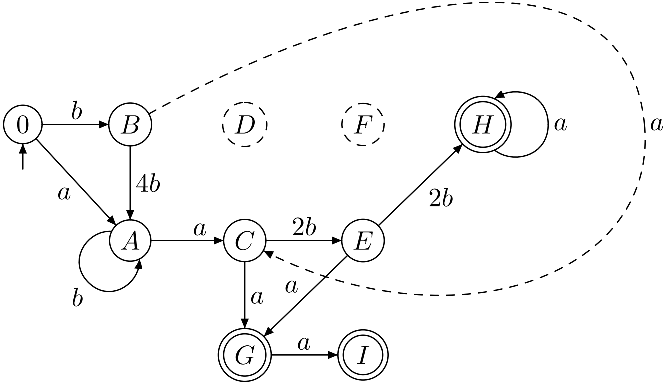
Blocks:

- $\{\bot, I, G\}$
- $\{F, E\}$
- $\{D, C\}$

- Blocks represent almost equivalence
- Scaling factors used in merges
- but we merged kernel states

Weighted merges

- merge of F into E with factor 2
- merge of D into C with factor 1



Conclusion

Solved

- hyper-minimisation for WDFA over semifields

Open

- Optimise number of errors
- Incremental construction
- Extensions to WNFA, more weight structures, etc.

Conclusion

Solved

- hyper-minimisation for WDFA over semifields

Open

- Optimise number of errors
- Incremental construction
- Extensions to WNFA, more weight structures, etc.

Thank you for your attention!

References

Andrew Badr, Viliam Geffert, and Ian Shipman.
Hyper-minimizing minimized deterministic finite state automata.
RAIRO Theor. Inf. Appl., 43(1), 2009.

Jason Eisner.
Simpler and more general minimization for weighted finite-state automata.
In *Proc. HLT-NAACL*, 2003.

Paweł Gawrychowski and Artur Jeż.
Hyper-minimisation made efficient.
In *Proc. MFCS*, volume 5734 of *LNCS*, 2009.

Markus Holzer and Andreas Maletti.
An $n \log n$ algorithm for hyper-minimizing a (minimized) deterministic automaton.
Theor. Comput. Sci., 411(38–39), 2010.

John E. Hopcroft.
An $n \log n$ Algorithm for Minimizing the States in a Finite Automaton.
In *The Theory of Machines and Computations*. Academic Press, 1971.