Pushing for weighted tree automata

Thomas Hanneforth Andreas Maletti Daniel Quernheim

Institute for Natural Language Processing
University of Stuttgart, Germany

maletti@ims.uni-stuttgart.de

QuantLA

Dresden, 2013

Outline

Motivation

Minimization

Equivalence testing

Motivation

Toolkit

- for unweighted and weighted tree automata
- support for all standard operations
- here: *minimization* and *equivalence testing*

Motivation

Toolkit

- for unweighted and weighted tree automata
- support for all standard operations
- here: *minimization* and *equivalence testing*

Notes

- mainly developed by THOMAS HANNEFORTH (U Potsdam)
- project name: TALib (tree automata library)

Motivation

Toolkit

- for unweighted and weighted tree automata
- support for all standard operations
- here: *minimization* and *equivalence testing*

Notes

- mainly developed by THOMAS HANNEFORTH (U Potsdam)
- project name: TALib (tree automata library)
- fitting *talib* (Arabic, “student”), but plural is *taliban*

Motivation

Toolkit

- for unweighted and weighted tree automata
- support for all standard operations
- here: *minimization* and *equivalence testing*

Why those?

- difference between unweighted and weighted
- *minimization* essential for large automata
- *equivalence testing* important for sanity checks

Motivation — Minimization

Typical automata

- English BERKELEY parser grammar 153 MB
  (1,133 states and 4,267,277 transitions)
- English EGRET parser grammar 107 MB
- Chinese EGRET parser grammar 98 MB

Semirings

Definition (commutative semiring $(S, +, \cdot, 0, 1)$)

- commutative monoids $(S, +, 0)$ and $(S, \cdot, 1)$
- absorption $s \cdot 0 = 0$ $\qquad s \in S$
- distributivity $s_1 \cdot (s_2 + s_3) = (s_1 \cdot s_2) + (s_1 \cdot s_3)$ $\qquad s_1, s_2, s_3 \in S$

Semirings

Definition (commutative semiring $(S, +, \cdot, 0, 1)$)

- commutative monoids $(S, +, 0)$ and $(S, \cdot, 1)$
- absorption $s \cdot 0 = 0$ $\qquad s \in S$
- distributivity $s_1 \cdot (s_2 + s_3) = (s_1 \cdot s_2) + (s_1 \cdot s_3)$ $\qquad s_1, s_2, s_3 \in S$

Definition (commutative semifield $(S, +, \cdot, 0, 1)$)

- commutative semiring $(S, +, \cdot, 0, 1)$
- for every $s \in S \setminus \{0\}$ there exists $s^{-1} \in S$ such that $s \cdot s^{-1} = 1$

Semirings

Definition (commutative semiring $(S, +, \cdot, 0, 1)$)

- commutative monoids $(S, +, 0)$ and $(S, \cdot, 1)$
- absorption $s \cdot 0 = 0$ $\qquad s \in S$
- distributivity $s_1 \cdot (s_2 + s_3) = (s_1 \cdot s_2) + (s_1 \cdot s_3)$ $\qquad s_1, s_2, s_3 \in S$

Definition (commutative semifield $(S, +, \cdot, 0, 1)$)

- commutative semiring $(S, +, \cdot, 0, 1)$
- for every $s \in S \setminus \{0\}$ there exists $s^{-1} \in S$ such that $s \cdot s^{-1} = 1$

Definition (commutative field $(S, +, \cdot, 0, 1)$)

- commutative semifield $(S, +, \cdot, 0, 1)$
- for every $s \in S$ there exists $(-s) \in S$ such that $s + (-s) = 0$

Weighted tree automaton

Definition (wta)

Weighted tree automaton (Q, Σ, μ, F)

- Q finite set of *states*
- Σ ranked alphabet of *input symbols*
- $\mu = (\mu_k)_{k \in \mathbb{N}}$ with $\mu_k \colon \Sigma_k \to S^{Q^k \times Q}$ *transition weight assignment*
- $F \subseteq Q$ *final states*

Weighted tree automaton

Definition (wta)

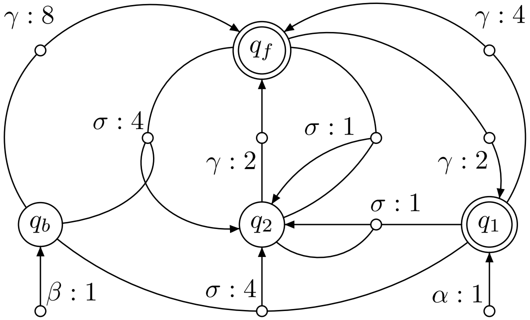
Weighted tree automaton (Q, Σ, μ, F)

- Q finite set of *states*
- Σ ranked alphabet of *input symbols*
- $\mu = (\mu_k)_{k \in \mathbb{N}}$ with $\mu_k \colon \Sigma_k \to S^{Q^k \times Q}$ *transition weight assignment*
- $F \subseteq Q$ *final states*

Definition (dwta)

A wta (Q, Σ, μ, F) is deterministic if for all $\sigma \in \Sigma_k$, $q, q', q_1, \dots, q_k \in Q$
$\mu_k(\sigma)_{q_1 \cdots q_k, q} \neq 0 \neq \mu_k(\sigma)_{q_1 \cdots q_k, q'}$ implies $q = q'$

Deterministic weighted tree automaton



Weighted tree automaton

Definition

The semantics of a wta $M = (Q, \Sigma, \mu, F)$ is $M \colon T_\Sigma \to S$

$$M(t) = \sum_{q \in F} h_\mu(t)_q \qquad t \in T_\Sigma$$

with $h_\mu \colon T_\Sigma(Q) \to S^Q$ $\qquad q, q' \in Q$, $\sigma \in \Sigma_k$, $t_1, \dots, t_k \in T_\Sigma(Q)$

$$h_\mu\big(q'\big)_q = \begin{cases} 1 & \text{if } q = q' \\ 0 & \text{otherwise} \end{cases}$$

$$h_\mu\big(\sigma(t_1, \dots, t_k)\big)_q = \sum_{q_1, \dots, q_k \in Q} \mu_k(\sigma)_{q_1 \cdots q_k, q} \cdot \prod_{i=1}^{k} h_\mu(t_i)_{q_i}$$

Outline

Motivation

Minimization

Equivalence testing

Minimization

State-of-the-art (tree / string)

	unweighted	weighted (field)
dwta	$\mathcal{O}(m \log n)$	$\mathcal{O}(mn)$ / $\mathcal{O}(m \log n)$
wta	PSPACE-complete	P / $\mathcal{O}(mn^2)$

Notes

- unweighted = weighted over $(\{0,1\}, \max, \min, 0, 1)$
- m = size of the transition table
- n = number of states

Minimization

State-of-the-art (tree / string)

	unweighted	weighted (field)
dwta	$\mathcal{O}(m \log n)$	$\mathcal{O}(mn)$ / $\mathcal{O}(m \log n)$
wta	PSPACE-complete	P / $\mathcal{O}(mn^2)$

Notes

- unweighted = weighted over $(\{0, 1\}, \max, \min, 0, 1)$
- m = size of the transition table
- n = number of states

Minimization

Let $(S, +, \cdot, 0, 1)$ be a semifield

Definition (Borchardt 2003)

States $q_1, q_2 \in Q$ in dwta (Q, Σ, μ, F) are equivalent ($q_1 \equiv q_2$) if there exists $s \in S \setminus \{0\}$ such that

$$\sum_{q \in F} h_\mu(c[q_1])_q = s \cdot \sum_{q \in F} h_\mu(c[q_2])_q \qquad c \in C_\Sigma(Q)$$

Minimization

Let $(S, +, \cdot, 0, 1)$ be a semifield

Definition (Borchardt 2003)

States $q_1, q_2 \in Q$ in dwta (Q, Σ, μ, F) are equivalent ($q_1 \equiv q_2$) if there exists $s \in S \setminus \{0\}$ such that

$$\sum_{q \in F} h_\mu(c[q_1])_q = s \cdot \sum_{q \in F} h_\mu(c[q_2])_q \qquad c \in C_\Sigma(Q)$$

Notes

- $q_1 \equiv q_2$ if they behave equally in all contexts (up to a constant invertable scaling factor)
- $\equiv$ is a congruence
- finer than the classical (unweighted) state equivalence

Minimization

Theorem (M. 2009)

Dwta over semifields can be minimized in time $\mathcal{O}(mn)$

Approach

1. Compute sign of life for each state
2. Compute equivalence pairwise with scaling factor
3. Merge equivalent states

Minimization

Theorem (M. 2009)

Dwta over semifields can be minimized in time $\mathcal{O}(mn)$

Approach

1. Compute sign of life for each state
2. Compute equivalence pairwise with scaling factor
3. Merge equivalent states

Definition

- $c \in C_\Sigma(Q)$ is sign of life for $q \in Q$ if $\sum_{q' \in F} h_\mu(c[q])_{q'} \neq 0$

Minimization

Theorem (M. 2009)

Dwta over semifields can be minimized in time $\mathcal{O}(mn)$

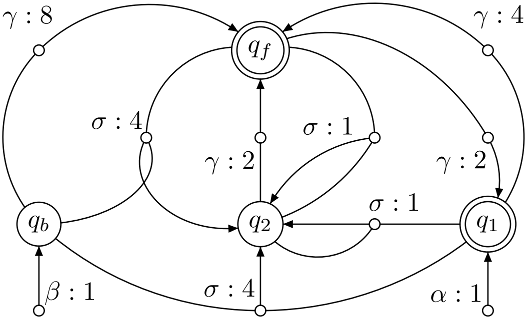
Approach

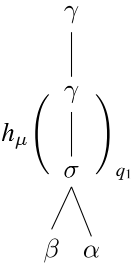
1. Compute sign of life for each state
2. Compute equivalence pairwise with scaling factor
3. Merge equivalent states

Definition

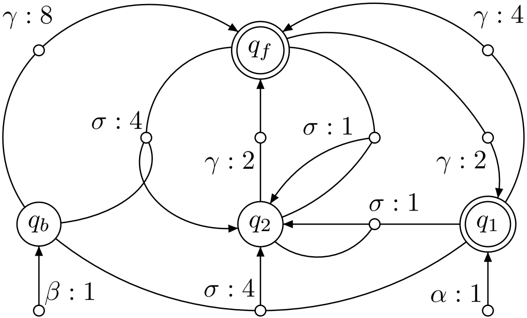
- $c \in C_\Sigma(Q)$ is sign of life for $q \in Q$ if $\sum_{q' \in F} h_\mu(c[q])_{q'} \neq 0$
- state that has a sign of life is live
- state without a sign of life is dead

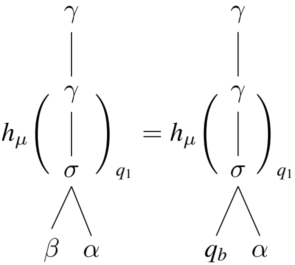
Sign of life


$\gamma : 8$
$\gamma : 4$
q_f
$\sigma : 4$
$\sigma : 1$
$\gamma : 2$
$\gamma : 2$
$\sigma : 1$
q_b
q_2
q_1
$\beta : 1$
$\sigma : 4$
$\alpha : 1$

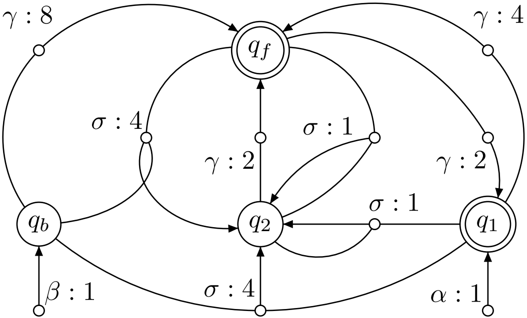

γ
h_μ
γ
σ
q_1
β
α

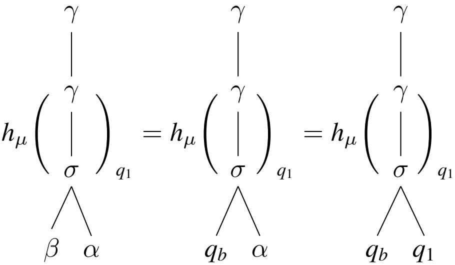
Sign of life


q_f
q_b
q_2
q_1
$\gamma : 8$
$\gamma : 4$
$\sigma : 4$
$\sigma : 1$
$\gamma : 2$
$\gamma : 2$
$\sigma : 1$
$\beta : 1$
$\sigma : 4$
$\alpha : 1$

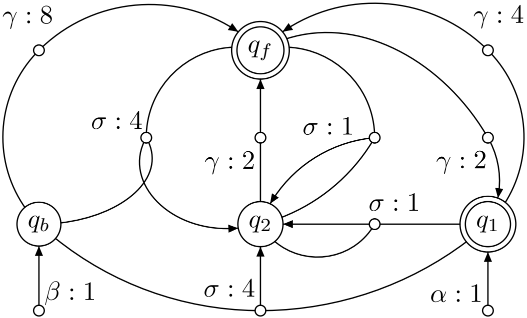

h_μ
γ
γ
σ
β α
q_1
$= h_\mu$
γ
γ
σ
q_b α
q_1

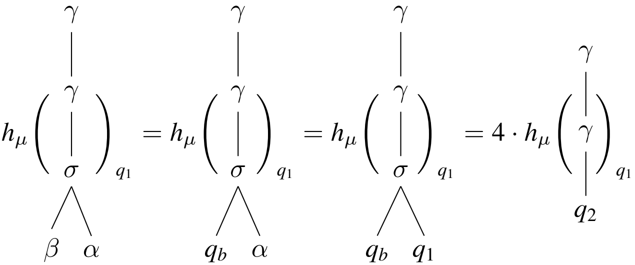
Sign of life


$\gamma : 8$
$\gamma : 4$
q_f
$\sigma : 4$
$\sigma : 1$
$\gamma : 2$
$\gamma : 2$
$\sigma : 1$
q_b
q_2
q_1
$\beta : 1$
$\sigma : 4$
$\alpha : 1$

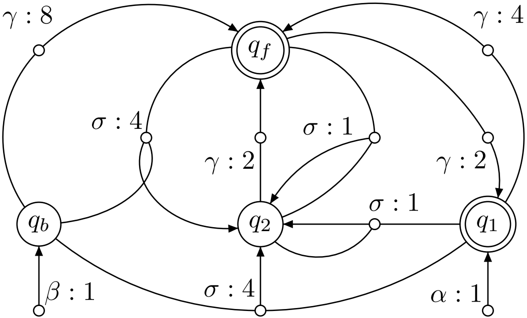

h_μ
γ
γ
σ
β α
q_1
$= h_\mu$
γ
γ
σ
q_b α
q_1
$= h_\mu$
γ
γ
σ
q_b q_1
q_1

Sign of life


q_f
q_b
q_2
q_1
$\gamma : 8$
$\gamma : 4$
$\sigma : 4$
$\sigma : 1$
$\gamma : 2$
$\gamma : 2$
$\sigma : 1$
$\beta : 1$
$\sigma : 4$
$\alpha : 1$

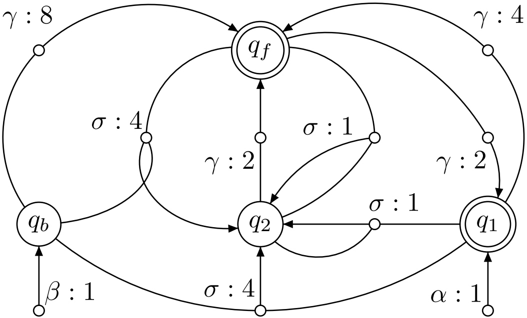


Sign of life



$$h_\mu\big(\gamma(\gamma(\sigma(\beta,\alpha)))\big)_{q_1} = h_\mu\big(\gamma(\gamma(\sigma(q_b,\alpha)))\big)_{q_1} = h_\mu\big(\gamma(\gamma(\sigma(q_b,q_1)))\big)_{q_1} = 4 \cdot h_\mu\big(\gamma(\gamma(q_2))\big)_{q_1} = 8 \cdot h_\mu\big(\gamma(q_f)\big)_{q_1}$$

Sign of life



$$h_\mu\big(\gamma(\gamma(\sigma(\beta, \alpha)))\big)_{q_1} = h_\mu\big(\gamma(\gamma(\sigma(q_b, \alpha)))\big)_{q_1} = h_\mu\big(\gamma(\gamma(\sigma(q_b, q_1)))\big)_{q_1} = 4 \cdot h_\mu\big(\gamma(\gamma(q_2))\big)_{q_1} = 8 \cdot h_\mu\big(\gamma(q_f)\big)_{q_1} = 16$$

Minimization

Our approach

Compute the classical unweighted equivalence

Minimization

Our approach

Compute the classical unweighted equivalence

Choose *one* sign of life for every equivalence class

Minimization

Our approach

Compute	the classical unweighted equivalence
Choose	*one* sign of life for every equivalence class
Normalize	transition weights according to signs of life (“pushing”)

Minimization

Our approach

Compute	the classical unweighted equivalence
Choose	*one* sign of life for every equivalence class
Normalize	transition weights according to signs of life (“pushing”)
Join	transition labels and weights

Minimization

Our approach

Compute	the classical unweighted equivalence
Choose	*one* sign of life for every equivalence class
Normalize	transition weights according to signs of life (“pushing”)
Join	transition labels and weights
Minimize	the obtained unweighted dwta

Minimization

Our approach

Compute	the classical unweighted equivalence
Choose	*one* sign of life for every equivalence class
Normalize	transition weights according to signs of life (“pushing”)
Join	transition labels and weights
Minimize	the obtained unweighted dwta
Expand	the labels again

Minimization

Our approach

- **Compute** the classical unweighted equivalence
- **Choose** *one* sign of life for every equivalence class
- **Normalize** transition weights according to signs of life (“pushing”)
- **Join** transition labels and weights
- **Minimize** the obtained unweighted dwta
- **Expand** the labels again

$\Rightarrow$ the resulting dwta is minimal

Signs of life

Computation

1. Compute (unweighted) MYHILL-NERODE equivalence $\cong$

Signs of life

Computation

1. Compute (unweighted) MYHILL-NERODE equivalence $\cong$
2. Compute $\text{sol}\colon (Q/{\cong}) \to C_\Sigma(Q)$ such that $\text{sol}([q])$ is a sign of life for q live $q \in Q$

Signs of life

Computation

1. Compute (unweighted) MYHILL-NERODE equivalence $\cong$
2. Compute $\mathrm{sol}\colon (Q/\cong) \to C_\Sigma(Q)$ such that
   $\mathrm{sol}([q])$ is a sign of life for q — live $q \in Q$
3. Compute $\lambda\colon Q \to (S \setminus \{0\})$ such that
   - $\lambda(q) = \sum_{q' \in F} h_\mu(c[q])_{q'}$ with $c = \mathrm{sol}([q])$ — live $q \in Q$
   - $\lambda(q) = 1$ — dead $q \in Q$

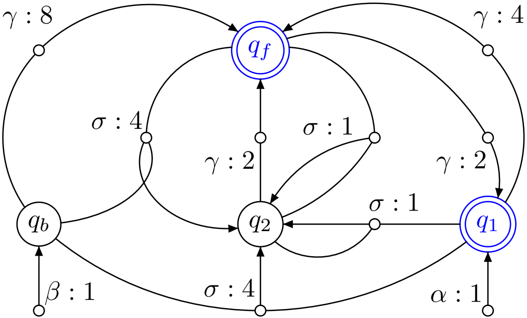
Signs of life

Computation

1. Compute (unweighted) MYHILL-NERODE equivalence $\cong$
2. Compute $\mathrm{sol}\colon (Q/{\cong}) \to C_\Sigma(Q)$ such that $\mathrm{sol}([q])$ is a sign of life for q — live $q \in Q$
3. Compute $\lambda\colon Q \to (S \setminus \{0\})$ such that
   - $\lambda(q) = \sum_{q' \in F} h_\mu(c[q])_{q'}$ with $c = \mathrm{sol}([q])$ — live $q \in Q$
   - $\lambda(q) = 1$ — dead $q \in Q$

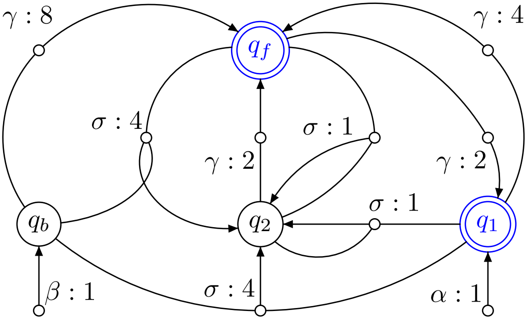
Complexity: $\mathcal{O}(m \log n)$

Signs of life



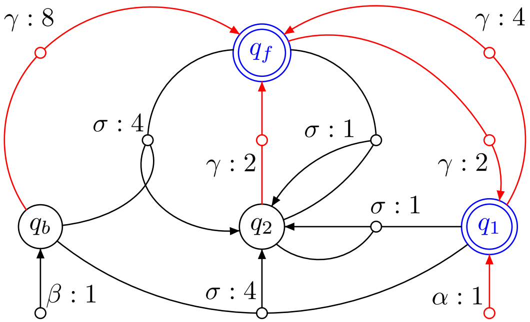
- Start with the equivalence $\cong = \{\{q_1, q_f\}, \{q_2, q_b\}\}$

Signs of life



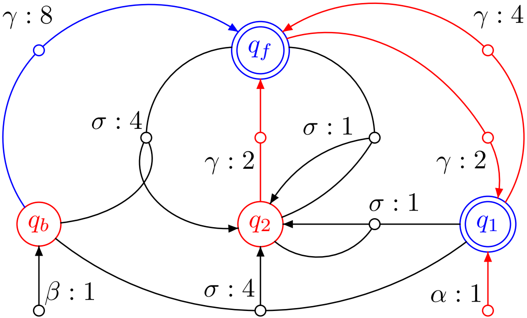
- Start with the equivalence $\cong = \{\{q_1, q_f\}, \{q_2, q_b\}\}$
- q_1 and q_f are trivially live with $\mathrm{sol}(\{q_1, q_f\}) = \square$
- $\lambda(q_1) = \lambda(q_f) = 1$

Signs of life



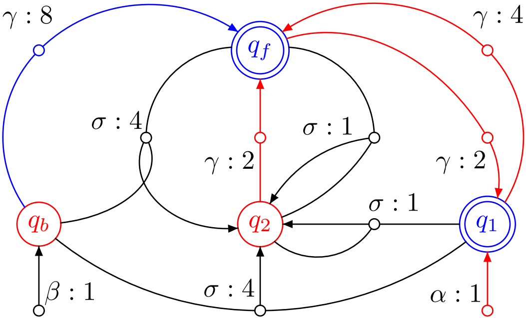
- Consider all transitions leading to q_1 or q_f

Signs of life



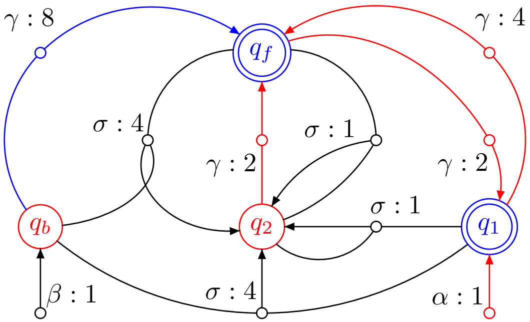
- Consider all transitions leading to q_1 or q_f
- Pick $\gamma(q_b)$ because q_b unexplored

Signs of life



- Consider all transitions leading to q_1 or q_f
- Pick $\gamma(q_b)$ because q_b unexplored
- Set $\mathrm{sol}(\{q_b, q_2\}) = \gamma(\square)$

Signs of life



- Consider all transitions leading to q_1 or q_f
- Pick $\gamma(q_b)$ because q_b unexplored
- Set $\mathrm{sol}(\{q_b, q_2\}) = \gamma(\square)$
- Set $\lambda(q_b) = \lambda(q_f) \cdot 8 = 8$ and $\lambda(q_2) = \lambda(q_f) \cdot 2 = 2$

Pushing

Definition

Given $\lambda\colon Q \to (S \setminus \{0\})$ such that $\lambda(q) = 1$ for all $q \in F$

$$\mathrm{push}_\lambda(M) = (Q, \Sigma, \mu', F)$$

$$\mu'_k(\sigma)_{q_1 \cdots q_k, q} = \prod_{i=1}^{k} \lambda(q_i)^{-1} \cdot \mu_k(\sigma)_{q_1 \cdots q_k, q} \cdot \lambda(q)$$

Pushing

Definition

Given $\lambda\colon Q \to (S \setminus \{0\})$ such that $\lambda(q) = 1$ for all $q \in F$

$$\text{push}_\lambda(M) = (Q, \Sigma, \mu', F)$$

$$\mu'_k(\sigma)_{q_1 \cdots q_k, q} = \prod_{i=1}^{k} \lambda(q_i)^{-1} \cdot \mu_k(\sigma)_{q_1 \cdots q_k, q} \cdot \lambda(q)$$

Notes

- Transitions to $q \in Q$ charge additional weight $\lambda(q)$

Pushing

Definition

Given $\lambda\colon Q \to (S \setminus \{0\})$ such that $\lambda(q) = 1$ for all $q \in F$

$$\text{push}_\lambda(M) = (Q, \Sigma, \mu', F)$$

$$\mu'_k(\sigma)_{q_1 \cdots q_k, q} = \prod_{i=1}^{k} \lambda(q_i)^{-1} \cdot \mu_k(\sigma)_{q_1 \cdots q_k, q} \cdot \lambda(q)$$

Notes

- Transitions to $q \in Q$ charge additional weight $\lambda(q)$
- Transitions leaving q_i compensate by charging the weight $\lambda(q_i)^{-1}$

Pushing

Definition

Given $\lambda\colon Q \to (S \setminus \{0\})$ such that $\lambda(q) = 1$ for all $q \in F$

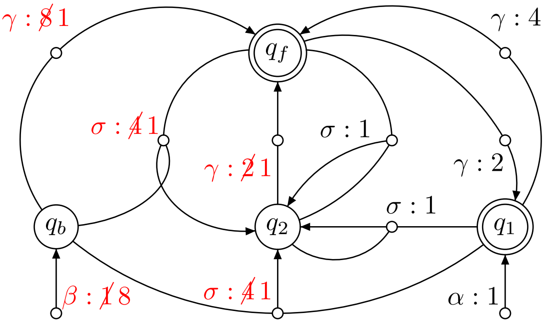
$$\mathrm{push}_\lambda(M) = (Q, \Sigma, \mu', F)$$

$$\mu'_k(\sigma)_{q_1 \cdots q_k, q} = \prod_{i=1}^{k} \lambda(q_i)^{-1} \cdot \mu_k(\sigma)_{q_1 \cdots q_k, q} \cdot \lambda(q)$$

Theorem

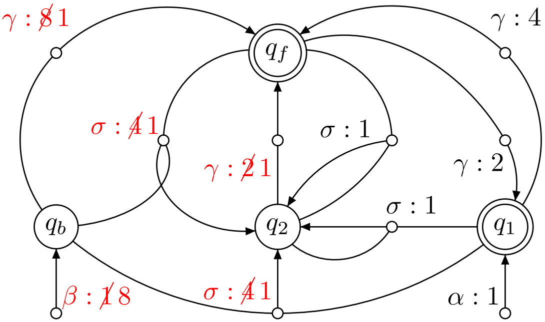
$\mathrm{push}_\lambda(M)$ *and* M *are equivalent*

Pushing



- $\lambda(q_1) = \lambda(q_f) = 1$
- $\lambda(q_2) = 2$
- $\lambda(q_b) = 8$

Pushing

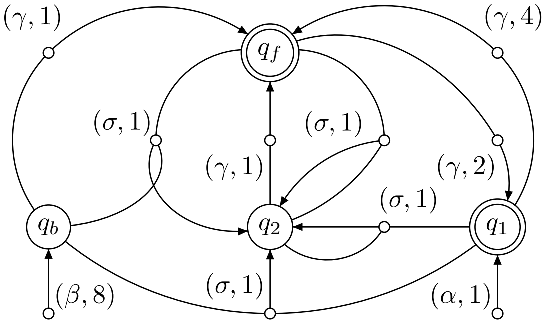


Theorem

Given suitable $\lambda \colon Q \to (S \setminus \{0\})$ *and* $\mathrm{push}_\lambda(M) = (Q, \Sigma, \mu', F)$

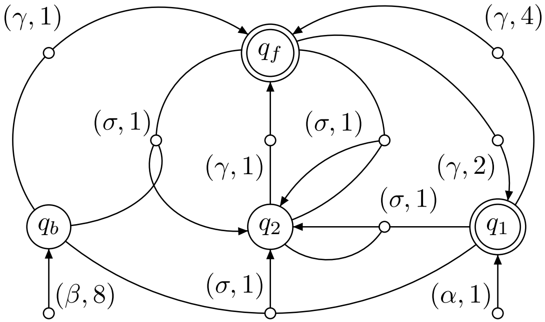
$$\mu'_k(\sigma)_{q_1 \cdots q_k, q} = \mu'_k(\sigma)_{q'_1 \cdots q'_k, q'} \qquad \sigma \in \Sigma_k,\ q_i \equiv q'_i, \text{ and } q \equiv q'$$

Minimization



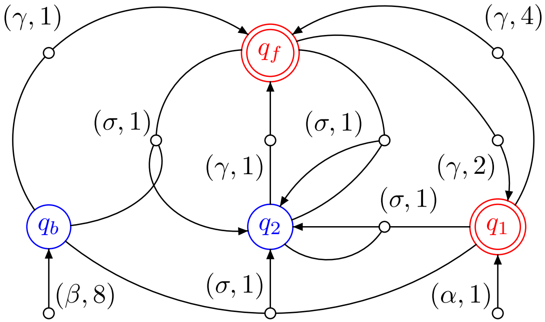
1. Make weight part of the label

Minimization



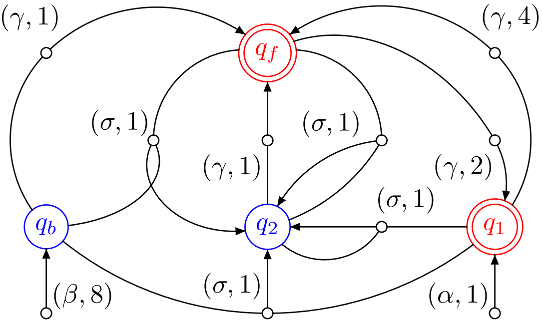
1. Make weight part of the label
2. minimize as (unweighted) dwta

Minimization



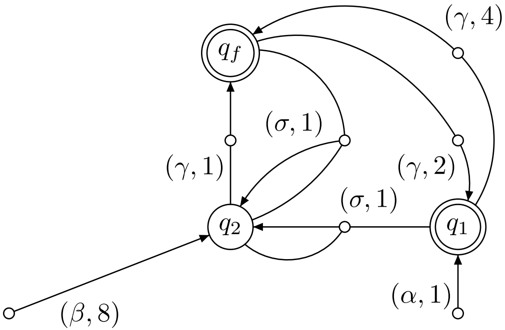
1. Make weight part of the label
2. minimize as (unweighted) dwta
   - q_b and q_2 equivalent (will be merged)

Minimization



1. Make weight part of the label
2. minimize as (unweighted) dwta
   - q_b and q_2 equivalent (will be merged)
   - q_1 and q_f not equivalent

Minimal dwta



Minimization

Theorem

We can minimize dwta in time $\mathcal{O}(m \log n)$

State-of-the-art (tree / string)

	unweighted	weighted (field)
dwta	$\mathcal{O}(m \log n)$	$\mathcal{O}(m \log n)$
wta	PSPACE-complete	P / $\mathcal{O}(mn^2)$

Outline

Motivation

Minimization

Equivalence testing

Equivalence testing — Motivation

Determinization sanity checking

1. Sum (union) construction of dwta M_1 and M_2 yields wta M
2. Determinization yields dwta M'
3. Check equivalence between M' and the result M'' of the union product construction for M_1 and M_2

Equivalence testing — Motivation

Determinization sanity checking

1. Sum (union) construction of dwta M_1 and M_2 yields wta M
2. Determinization yields dwta M'
3. Check equivalence between M' and the result M'' of the union product construction for M_1 and M_2

Minimization sanity checking

1. Minimize dwta M to obtain M'
2. Check equivalence between M and M'

Equivalence testing — Motivation

Determinization sanity checking

1. Sum (union) construction of dwta M_1 and M_2 yields wta M
2. Determinization yields dwta M'
3. Check equivalence between M' and the result M'' of the union product construction for M_1 and M_2

~~Minimization sanity checking~~

1. ~~Minimize dwta M to obtain M'~~
2. ~~Check equivalence between M and M'~~

Equivalence testing

State-of-the-art (tree / string)

	unweighted	weighted (field)
dwta	$\mathcal{O}(m \log n)$ / $\mathcal{O}(m \log^* n)$	$\mathcal{O}(m_1 m_2)$ / $\mathcal{O}(m \log n)$
wta	EXPTIME-complete	P / $\mathcal{O}(mn^2)$

Notes

- $m = \max(m_1, m_2)$
- $n = \max(n_1, n_2)$

Equivalence testing

State-of-the-art (tree / string)

	unweighted	weighted (field)
dwta	$\mathcal{O}(m \log n)$ / $\mathcal{O}(m \log^* n)$	$\mathcal{O}(m_1 m_2)$ / $\mathcal{O}(m \log n)$
wta	EXPTIME-complete	P / $\mathcal{O}(mn^2)$

Notes

- $m = \max(m_1, m_2)$
- $n = \max(n_1, n_2)$

Equivalence testing

Definition

Dwta M and M' are push-isomorphic if there exists $\lambda \colon Q \to (S \setminus \{0\})$ with $\lambda(q) = 1$ for all $q \in F$ such that M' is isomorphic to $\mathrm{push}_\lambda(M)$

Equivalence testing

Definition

Dwta M and M' are push-isomorphic if there exists $\lambda\colon Q \to (S \setminus \{0\})$ with $\lambda(q) = 1$ for all $q \in F$ such that M' is isomorphic to $\mathrm{push}_\lambda(M)$

Theorem

All equivalent minimal dwta are push-isomorphic

Equivalence testing

Definition

Dwta M and M' are push-isomorphic if there exists $\lambda \colon Q \to (S \setminus \{0\})$ with $\lambda(q) = 1$ for all $q \in F$ such that M' is isomorphic to $\mathrm{push}_\lambda(M)$

Theorem

All equivalent minimal dwta are push-isomorphic

Approach

- Minimize both M_1 and M_2

Equivalence testing

Definition

Dwta M and M' are push-isomorphic if there exists $\lambda\colon Q \to (S \setminus \{0\})$ with $\lambda(q) = 1$ for all $q \in F$ such that M' is isomorphic to $\mathrm{push}_\lambda(M)$

Theorem

All equivalent minimal dwta are push-isomorphic

Approach

- Minimize both M_1 and M_2
- Check push-isomorphism (isomorphism after special pushing)

Equivalence testing

Theorem

We can test equivalence for dwta in time $\mathcal{O}(m \log n)$

State-of-the-art (tree / string)

	unweighted	weighted (field)
dwta	$\mathcal{O}(m \log n)$ / $\mathcal{O}(m \log^* n)$	$\mathcal{O}(m \log n)$
wta	EXPTIME-complete	P / $\mathcal{O}(mn^2)$

Summary

Minimization (tree / string)

	unweighted	weighted (field)
dwta	$\mathcal{O}(m \log n)$	$\mathcal{O}(m \log n)$
wta	PSPACE-complete	P / $\mathcal{O}(mn^2)$

Equivalence testing (tree / string)

	unweighted	weighted (field)
dwta	$\mathcal{O}(m \log n)$ / $\mathcal{O}(m \log^* n)$	$\mathcal{O}(m \log n)$
wta	EXPTIME-complete	P / $\mathcal{O}(mn^2)$