Równania nad zbiorami liczb naturalnych

Artur Jeż

UWr

19.04.2008

Model

- zmienne $X_i \subseteq \mathbb{N}$

Model

- zmienne $X_i \subseteq \mathbb{N}$
- operacje $\cap, \cup, +, \{n\}$

$$X + Y = \{x + y \mid x \in X, y \in Y\}$$

Model

- zmienne $X_i \subseteq \mathbb{N}$
- operacje $\cap, \cup, +, \{n\}$

$$X + Y = \{x + y \mid x \in X, y \in Y\}$$

- Równania postaci

$$\psi_i(X_1, \ldots, X_n) \quad = \quad \varphi_i(X_1, \ldots, X_n) \quad \text{dla } i = 1, \ldots, k$$

Model

- zmienne $X_i \subseteq \mathbb{N}$
- operacje $\cap, \cup, +, \{n\}$

$$X + Y = \{x + y \mid x \in X, y \in Y\}$$

- Równania postaci

$$\begin{aligned} \psi_i(X_1, \ldots, X_n) &= \varphi_i(X_1, \ldots, X_n) \quad \text{dla } i = 1, \ldots, k \\ X_i &= \varphi_i(X_1, \ldots, X_n) \quad \text{dla } i = 1, \ldots, n \end{aligned}$$

Model

- zmienne $X_i \subseteq \mathbb{N}$
- operacje $\cap, \cup, +, \{n\}$

$$X + Y = \{x + y \mid x \in X, y \in Y\}$$

- Równania postaci

$$\begin{aligned} \psi_i(X_1, \ldots, X_n) &= \varphi_i(X_1, \ldots, X_n) \quad \text{dla } i = 1, \ldots, k \\ X_i &= \varphi_i(X_1, \ldots, X_n) \quad \text{dla } i = 1, \ldots, n \end{aligned}$$

- rozwiązania
  1. najmniejsze, największe, jedyne
  2. najmniejsze (*EQ*)

Przykład

$$X = X + \{2\} \cup \{0\}$$

$$\{2n \mid n \geq 0\}$$

Przykład

$$X = X + \{2\} \cup \{0\}$$
$$\{2n \mid n \geq 0\}$$

Motywacja

- gramatyki koniunkcyjne unarne

Przykład

$$X = X + \{2\} \cup \{0\}$$
$$\{2n \mid n \geq 0\}$$

Motywacja

- gramatyki koniunkcyjne unarne
- obwody arytmetyczne

Notacja i techniki

- narzędzie: notacja k-pozycyjna

Notacja i techniki

- narzędzie: notacja k-pozycyjna
- liczba $n \Leftrightarrow$ jej notacja $(w)_k$

Notacja i techniki

- narzędzie: notacja k-pozycyjna
- liczba $n \Leftrightarrow$ jej notacja $(w)_k$
- zbiór liczb $\Leftrightarrow$ zbiór notacji liczb

Notacja i techniki

- narzędzie: notacja k-pozycyjna
- liczba $n \Leftrightarrow$ jej notacja $(w)_k$
- zbiór liczb $\Leftrightarrow$ zbiór notacji liczb
- podstawa konstrukcji i dowodu

Notacja i techniki

- narzędzie: notacja k-pozycyjna
- liczba $n \Leftrightarrow$ jej notacja $(w)_k$
- zbiór liczb $\Leftrightarrow$ zbiór notacji liczb
- podstawa konstrukcji i dowodu

Przykład

$$\{4^n \mid n \geq 0\} \Leftrightarrow (10^*)_4$$

Wyniki

Stare

Twierdzenie

Dla każdego k naturalnego i każdego R — regularnego nad $\{0, \ldots k-1\}$ zbiór liczb $(R)_k$ jest w EQ.

Wyniki

Stare

Twierdzenie

Dla każdego k naturalnego i każdego R — regularnego nad $\{0, \ldots k-1\}$ zbiór liczb $(R)_k$ jest w EQ.

Nowe

1. szersza klasa zbiorów (trellis automata)

Wyniki

Stare

Twierdzenie

Dla każdego k naturalnego i każdego R — regularnego nad $\{0, \ldots k-1\}$ zbiór liczb $(R)_k$ jest w EQ.

Nowe

1. szersza klasa zbiorów (trellis automata)
2. EXPTIME-zupełność dla *EQ*

Wyniki

Stare

Twierdzenie

Dla każdego k naturalnego i każdego R — regularnego nad $\{0, \ldots k-1\}$ zbiór liczb $(R)_k$ jest w EQ.

Nowe

1. szersza klasa zbiorów (trellis automata)
2. EXPTIME-zupełność dla *EQ*
3. uniwersalność w ogólnym przypadku

Wyniki

Stare

Twierdzenie

Dla każdego k naturalnego i każdego R — regularnego nad $\{0, \ldots k-1\}$ zbiór liczb $(R)_k$ jest w EQ.

Nowe

1. szersza klasa zbiorów (trellis automata)
2. EXPTIME-zupełność dla *EQ*
3. uniwersalność w ogólnym przypadku
4. jedna zmienna i jedno równanie

Poprzedni wynik

Przykład

Zbiory $(10^*)_4, (20^*)_4, (30^*)_4, (120^*)_4$
Konstruujemy jednocześnie

Poprzedni wynik

Przykład

Zbiory $(10^*)_4, (20^*)_4, (30^*)_4, (120^*)_4$
Konstruujemy jednocześnie Np. $(10^*)_4$

$$\big((10^*)_4 + (30^*)\big) \cap \big((20^*)_4 + (20^*)\big) =$$
$$\big((10^+)_4 \cup (10^*30^*)_4 \cup (30^*10^*)_4\big) \cap \big((10^+)_4 \cup (20^*20^*)_4\big) =$$
$$(10^+)_4$$

Poprzedni wynik

Przykład

Zbiory $(10^*)_4, (20^*)_4, (30^*)_4, (120^*)_4$
Konstruujemy jednocześnie Np. $(10^*)_4$

$$\big((10^*)_4 + (30^*)\big) \cap \big((20^*)_4 + (20^*)\big) =$$
$$\big((10^+)_4 \cup (10^*30^*)_4 \cup (30^*10^*)_4\big) \cap \big((10^+)_4 \cup (20^*20^*)_4\big) =$$
$$(10^+)_4$$

- Pozostałe zbiory analogicznie

Poprzedni wynik

Przykład

Zbiory $(10^*)_4, (20^*)_4, (30^*)_4, (120^*)_4$
Konstruujemy jednocześnie Np. $(10^*)_4$

$$\big((10^*)_4 + (30^*)\big) \cap \big((20^*)_4 + (20^*)\big) =$$
$$\big((10^+)_4 \cup (10^*30^*)_4 \cup (30^*10^*)_4\big) \cap \big((10^+)_4 \cup (20^*20^*)_4\big) =$$
$$(10^+)_4$$

- Pozostałe zbiory analogicznie
- Języki regularne — więcej pracy

Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;

Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

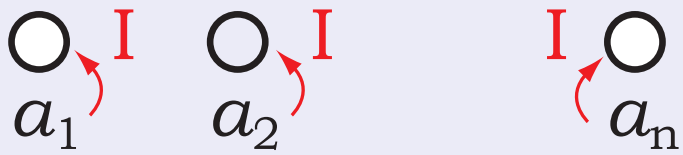
- Σ: alfabet wejściowy;
- Q: zbiór stanów;

Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;
- Q: zbiór stanów;
- $I : \Sigma \to Q$ ustawia stan początkowy;

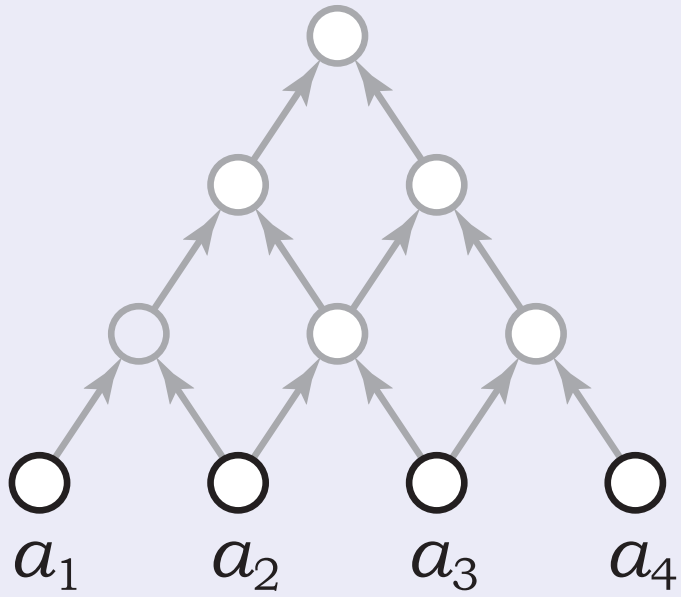


Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;
- Q: zbiór stanów;
- $I : \Sigma \to Q$ ustawia stan początkowy;

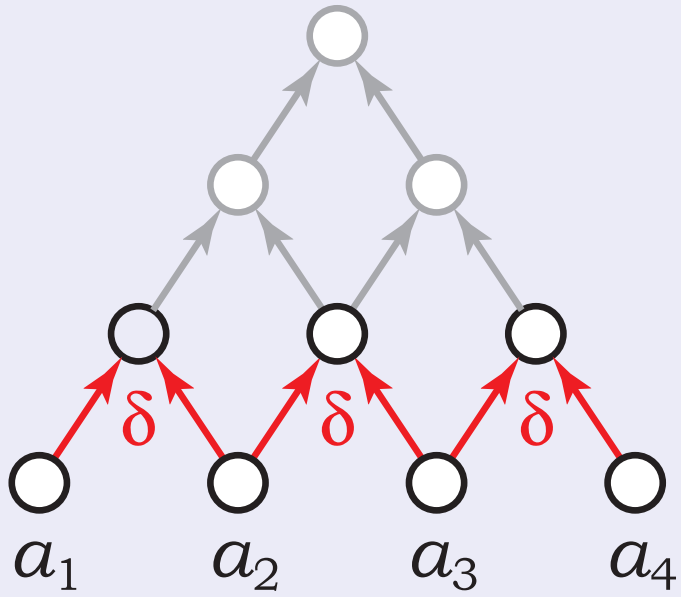


Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;
- Q: zbiór stanów;
- $I : \Sigma \to Q$ ustawia stan początkowy;
- $\delta : Q \times Q \to Q$, funkcja przejścia

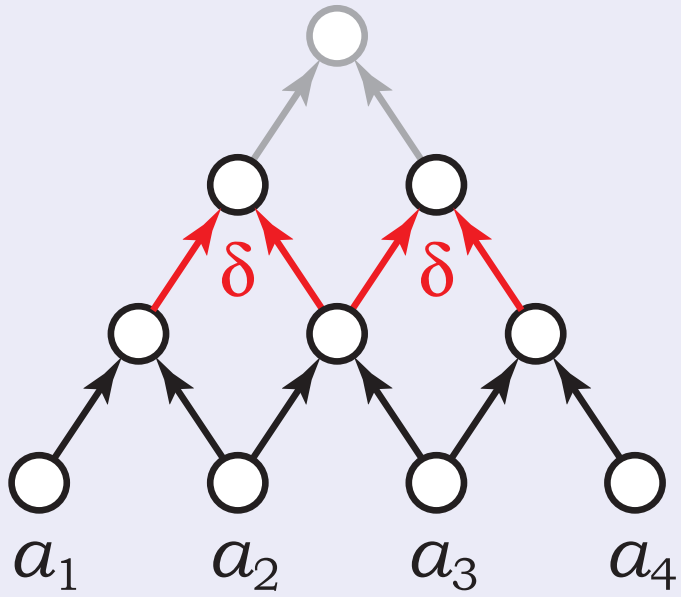


Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;
- Q: zbiór stanów;
- $I : \Sigma \to Q$ ustawia stan początkowy;
- $\delta : Q \times Q \to Q$, funkcja przejścia

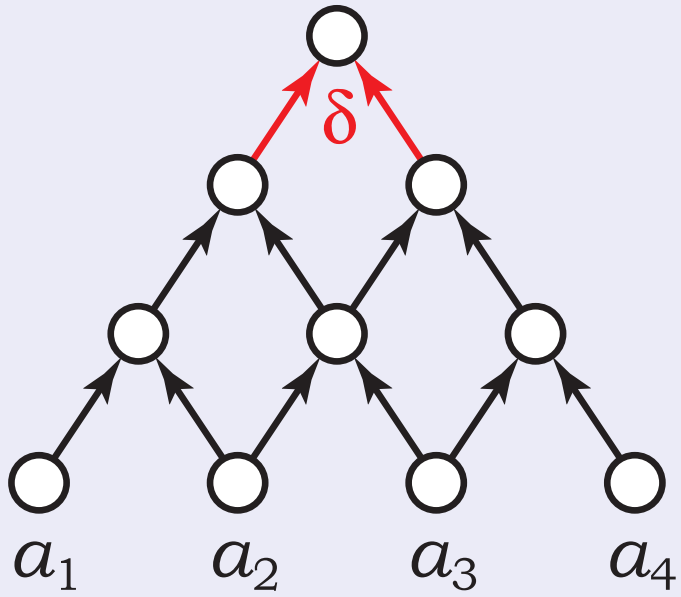


Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;
- Q: zbiór stanów;
- $I : \Sigma \to Q$ ustawia stan początkowy;
- $\delta : Q \times Q \to Q$, funkcja przejścia

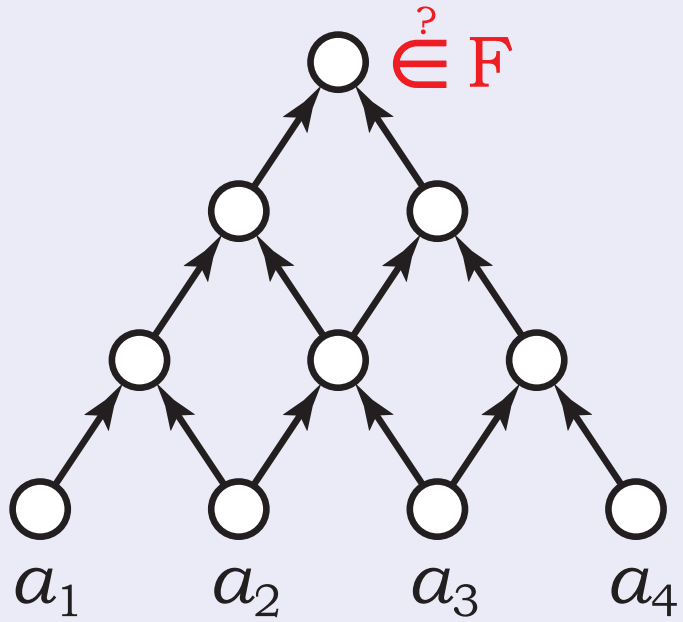


Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;
- Q: zbiór stanów;
- $I : \Sigma \to Q$ ustawia stan początkowy;
- $\delta : Q \times Q \to Q$, funkcja przejścia
- $F \subset Q$: stany akceptujące.

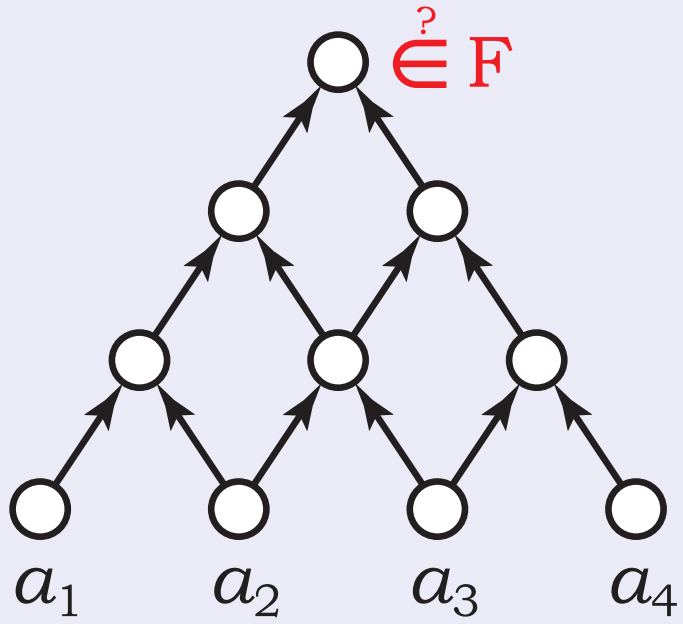


Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;
- Q: zbiór stanów;
- $I : \Sigma \to Q$ ustawia stan początkowy;
- $\delta : Q \times Q \to Q$, funkcja przejścia
- $F \subset Q$: stany akceptujące.



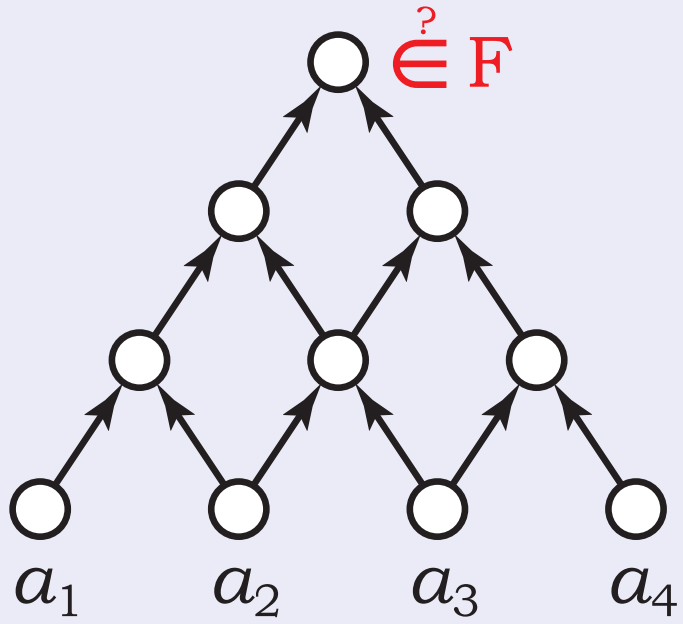
- Zamknięte na $\cup, \cap, \sim$, nie zamknięte na konkatenację.

Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;
- Q: zbiór stanów;
- $I : \Sigma \to Q$ ustawia stan początkowy;
- $\delta : Q \times Q \to Q$, funkcja przejścia
- $F \subset Q$: stany akceptujące.



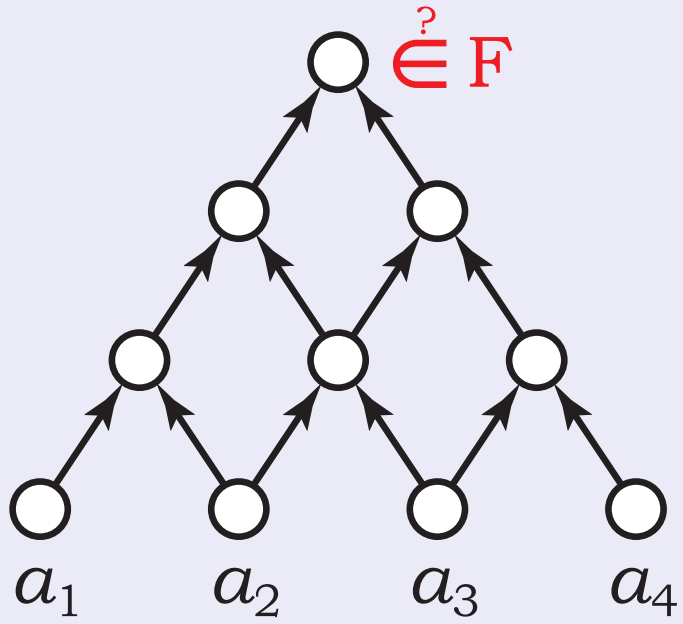
- Zamknięte na $\cup, \cap, \sim$, nie zamknięte na konkatenację.
- Rozpoznaje $\{wcw : w \in \{a, b\}^*\}$, $\{a^n b^n c^n\}$, $\{a^n b^{2^n}\}$, VALC.

Moc wyrażania

Definicja

Trellis automaton: $M = (\Sigma, Q, I, \delta, F)$

- Σ: alfabet wejściowy;
- Q: zbiór stanów;
- $I : \Sigma \to Q$ ustawia stan początkowy;
- $\delta : Q \times Q \to Q$, funkcja przejścia
- $F \subset Q$: stany akceptujące.



- Zamknięte na $\cup, \cap, \sim$, nie zamknięte na konkatenację.
- Rozpoznaje $\{wcw : w \in \{a, b\}^*\}$, $\{a^n b^n c^n\}$, $\{a^n b^{2^n}\}$, VALC.

Twierdzenie

Dla każdego języka $L \subseteq \{0, \ldots k-1\}^*$ rozpoznawanego przez M — trellis automata zbiór liczb $(L)_k$ należy do klasy EQ.

Idea konstrukcji

- Zbiór zmiennych $\{X_q \mid q \in Q\}$, reprezentuje $\{(L_M(q))_k \mid q \in Q\}$.

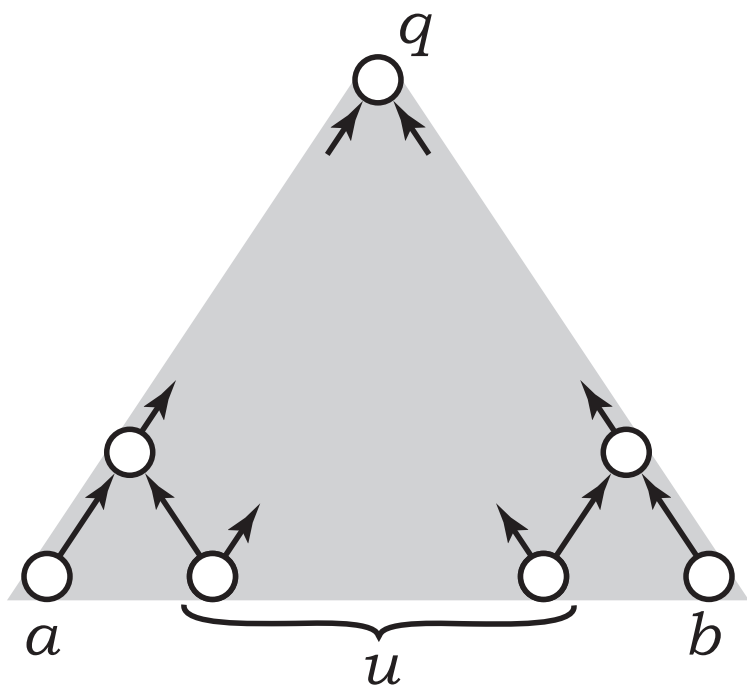
Idea konstrukcji

- Zbiór zmiennych $\{X_q \mid q \in Q\}$, reprezentuje $\{(L_M(q))_k \mid q \in Q\}$.
- Dokładniej $X_q = \{(1w10^*)_k \mid w \in L_M(q)\}$

Idea konstrukcji

- Zbiór zmiennych $\{X_q \mid q \in Q\}$, reprezentuje $\{(L_M(q))_k \mid q \in Q\}$.
- Dokładniej $X_q = \{(1w10^*)_k \mid w \in L_M(q)\}$

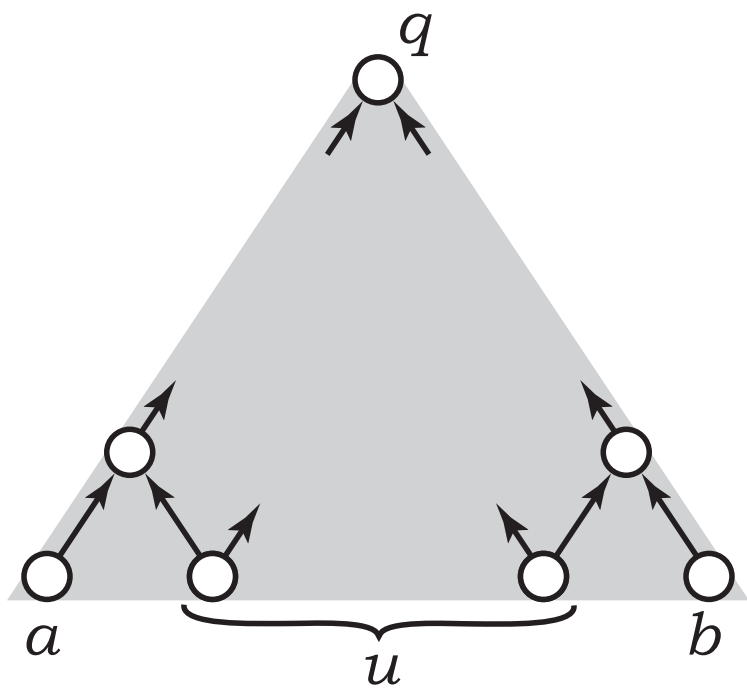
- $aub \in L_M(q)$



Idea konstrukcji

- Zbiór zmiennych $\{X_q \mid q \in Q\}$, reprezentuje $\{(L_M(q))_k \mid q \in Q\}$.
- Dokładniej $X_q = \{(1w10^*)_k \mid w \in L_M(q)\}$

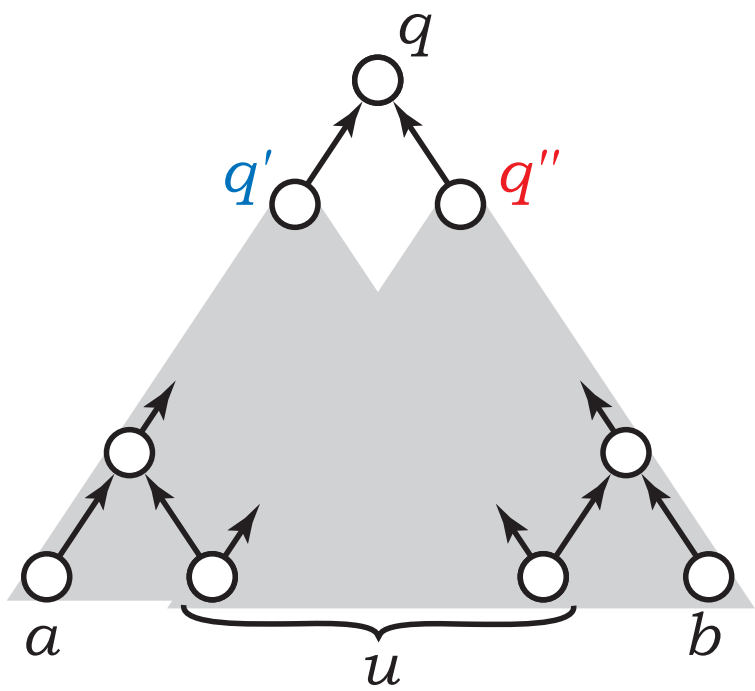
- $aub \in L_M(q)$ wtw



Idea konstrukcji

- Zbiór zmiennych $\{X_q \mid q \in Q\}$, reprezentuje $\{(L_M(q))_k \mid q \in Q\}$.
- Dokładniej $X_q = \{(1w10^*)_k \mid w \in L_M(q)\}$

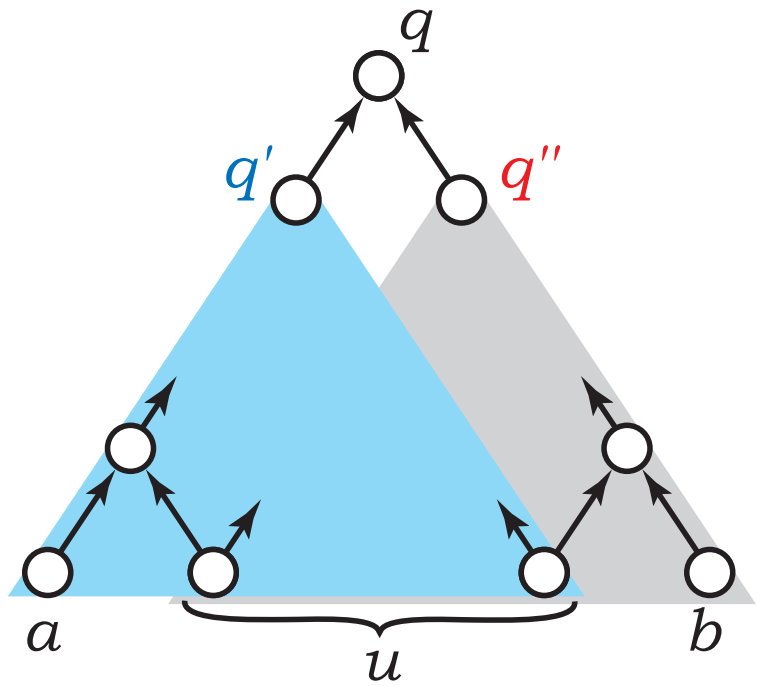
- $aub \in L_M(q)$ wtw
  $\exists q', q'' : \delta(q', q'') = q$,



Idea konstrukcji

- Zbiór zmiennych $\{X_q \mid q \in Q\}$, reprezentuje $\{(L_M(q))_k \mid q \in Q\}$.
- Dokładniej $X_q = \{(1w10^*)_k \mid w \in L_M(q)\}$

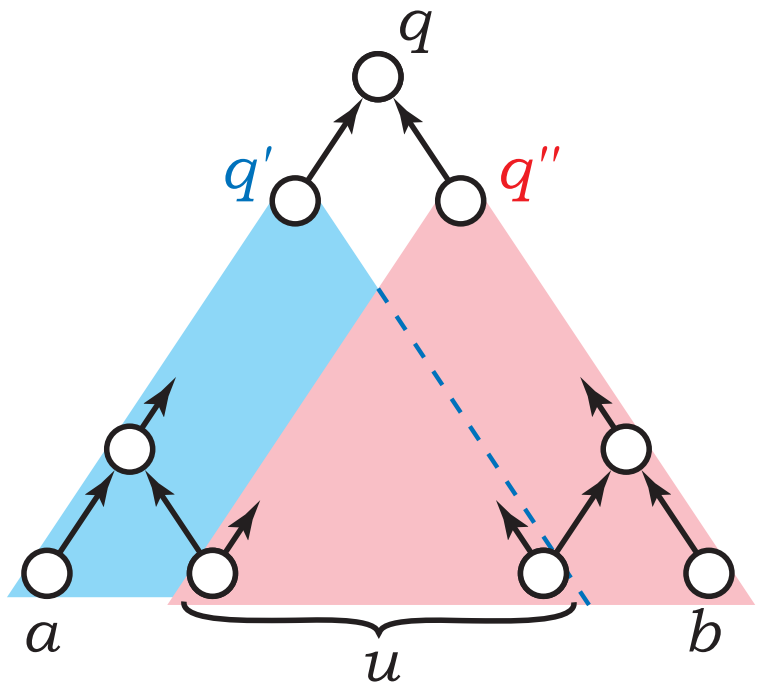
- $aub \in L_M(q)$ wtw
  $\exists q', q'' : \delta(q', q'') = q$,
  $au \in L_M(q')$,



Idea konstrukcji

- Zbiór zmiennych $\{X_q \mid q \in Q\}$, reprezentuje $\{(L_M(q))_k \mid q \in Q\}$.
- Dokładniej $X_q = \{(1w10^*)_k \mid w \in L_M(q)\}$

- $aub \in L_M(q)$ wtw
  $\exists q', q'' : \delta(q', q'') = q$,
  $au \in L_M(q')$,
  $ub \in L_M(q'')$.

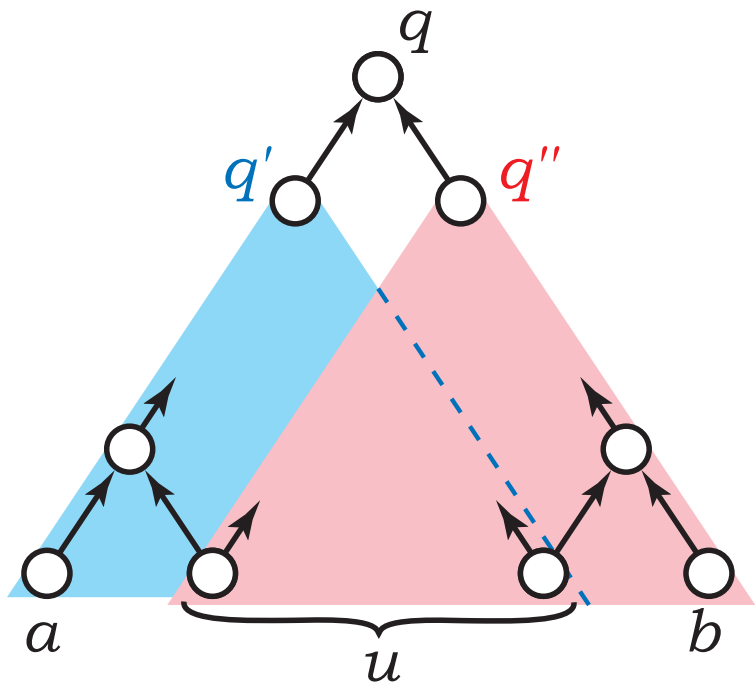


Idea konstrukcji

- Zbiór zmiennych $\{X_q \mid q \in Q\}$, reprezentuje $\{(L_M(q))_k \mid q \in Q\}$.
- Dokładniej $X_q = \{(1w10^*)_k \mid w \in L_M(q)\}$

- $aub \in L_M(q)$ wtw
  $\exists q', q'' : \delta(q', q'') = q$,
  $au \in L_M(q')$,
  $ub \in L_M(q'')$.
- Niech $(1au10^*)_k \subseteq X_{q'}$,
  $(1ub10^*)_k \subseteq X_{q''}$.

$$X_q = \bigcup_{\substack{q',q'':\delta(q',q'')=q \\ a,b\in\Sigma_k}} \rho_b(X_{q'}) \cap \lambda_a(X_{q''})$$

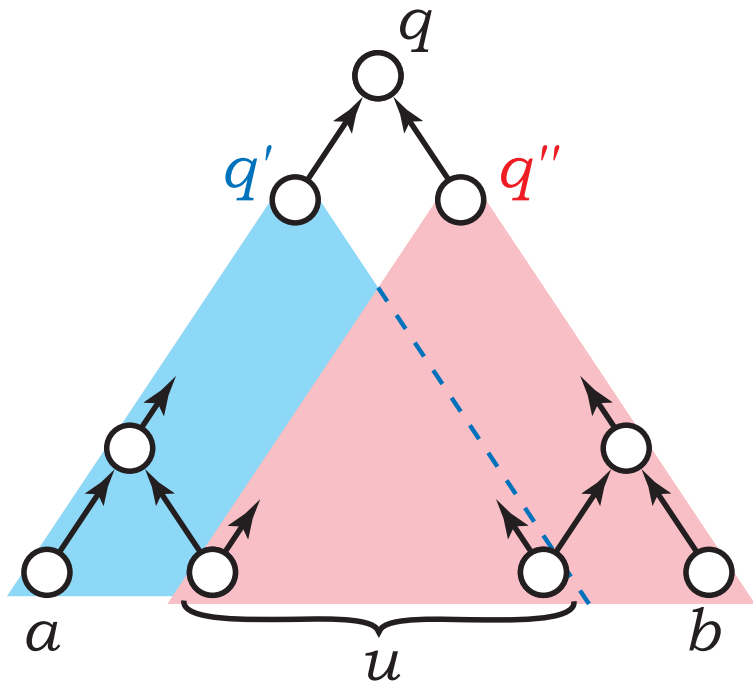


Idea konstrukcji

- Zbiór zmiennych $\{X_q \mid q \in Q\}$, reprezentuje $\{(L_M(q))_k \mid q \in Q\}$.
- Dokładniej $X_q = \{(1w10^*)_k \mid w \in L_M(q)\}$

- $aub \in L_M(q)$ wtw
  $\exists q', q'' : \delta(q', q'') = q$,
  $au \in L_M(q')$,
  $ub \in L_M(q'')$.
- Niech $(1au10^*)_k \subseteq X_{q'}$,
  $(1ub10^*)_k \subseteq X_{q''}$.

$$X_q = \bigcup_{\substack{q',q'':\delta(q',q'')=q \\ a,b\in\Sigma_k}} \rho_b(X_{q'}) \cap \lambda_a(X_{q''})$$



$$\lambda_a((1w10^\ell)_k) = (1aw10^\ell)_k$$
$$\rho_b((1w10^\ell)_k) = (1wb10^{\ell-1})_k$$

Złożoność obliczeniowa

Twierdzenie

EQ zawiera zbiory EXPTIME-trudne.

Złożoność obliczeniowa

Twierdzenie

EQ zawiera zbiory EXPTIME-trudne.

Uwaga

- Bez $\cap$ — NP trudne
- zawarte w EXPTIME

Złożoność obliczeniowa

Twierdzenie

EQ zawiera zbiory EXPTIME-trudne.

Uwaga

- Bez $\cap$ — NP trudne
- zawarte w EXPTIME

Idea

- *kodowanie ATM*

Złożoność obliczeniowa

Twierdzenie

EQ zawiera zbiory EXPTIME-trudne.

Uwaga

- Bez $\cap$ — NP trudne
- zawarte w EXPTIME

Idea

- *kodowanie ATM*
- *liczba — konfiguracja*

Złożoność obliczeniowa

Twierdzenie

EQ zawiera zbiory EXPTIME-trudne.

Uwaga

- Bez $\cap$ — NP trudne
- zawarte w EXPTIME

Idea

- *kodowanie ATM*
- *liczba — konfiguracja*
- $\cap$, $\cup$ — *alternacja*

Złożoność obliczeniowa

Twierdzenie

EQ zawiera zbiory EXPTIME-trudne.

Uwaga

- Bez $\cap$ — NP trudne
- zawarte w EXPTIME

Idea

- *kodowanie ATM*
- *liczba — konfiguracja*
- $\cap$, $\cup$ — *alternacja*
- *problem — liczby rosną — licznik*

Równania $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$

Fakt (Z ogólnej teorii równań języków)

Rozwiązania najmniejsze(największe, jedyne) są RE (co-RE, rekurencyjne).

Równania $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$

Fakt (Z ogólnej teorii równań języków)

Rozwiązania najmniejsze(największe, jedyne) są RE (co-RE, rekurencyjne).

Twierdzenie

Każdy zbiór RE (co-RE, rekurencyjny) jest rozwiązaniem najmniejszym (największym, jedynym) układu równań postaci $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$*, gdzie* φ, ψ *używają* $\cap, +$ *(*$\cup, +$*).*

Równania $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$

Fakt (Z ogólnej teorii równań języków)

Rozwiązania najmniejsze(największe, jedyne) są RE (co-RE, rekurencyjne).

Twierdzenie

Każdy zbiór RE (co-RE, rekurencyjny) jest rozwiązaniem najmniejszym (największym, jedynym) układu równań postaci $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$*, gdzie* φ, ψ *używają* $\cap, +$ *(*$\cup, +$*).*

Idea

- *M — maszyna Turinga*

Równania $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$

Fakt (Z ogólnej teorii równań języków)

Rozwiązania najmniejsze(największe, jedyne) są RE (co-RE, rekurencyjne).

Twierdzenie

Każdy zbiór RE (co-RE, rekurencyjny) jest rozwiązaniem najmniejszym (największym, jedynym) układu równań postaci $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$*, gdzie* φ, ψ *używają* $\cap, +$ *(*$\cup, +$*).*

Idea

- *M — maszyna Turinga*
- *tworzymy* $VALC(M) \in EQ$

Równania $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$

Fakt (Z ogólnej teorii równań języków)

Rozwiązania najmniejsze(największe, jedyne) są RE (co-RE, rekurencyjne).

Twierdzenie

Każdy zbiór RE (co-RE, rekurencyjny) jest rozwiązaniem najmniejszym (największym, jedynym) układu równań postaci $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$*, gdzie* φ, ψ *używają* $\cap, +$ *(*$\cup, +$*).*

Idea

- *M — maszyna Turinga*
- *tworzymy* $VALC(M) \in EQ$
- *ogólne równania z jedynym rozwiązaniem* $VALC(M)$

Równania $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$

Fakt (Z ogólnej teorii równań języków)

Rozwiązania najmniejsze(największe, jedyne) są RE (co-RE, rekurencyjne).

Twierdzenie

Każdy zbiór RE (co-RE, rekurencyjny) jest rozwiązaniem najmniejszym (największym, jedynym) układu równań postaci $\psi_i(X_1, \ldots, X_n) = \varphi_i(X_1, \ldots, X_n)$*, gdzie* φ, ψ *używają* $\cap, +$ *(*$\cup, +$*).*

Idea

- *M — maszyna Turinga*
- *tworzymy* $VALC(M) \in EQ$
- *ogólne równania z jedynym rozwiązaniem* $VALC(M)$
- *odzyskujemy słowa z* $L(M)$

Jedna zmienna

Problem

Ile zmiennych potrzeba do uzyskania tych wyników.

Jedna zmienna

Problem

Ile zmiennych potrzeba do uzyskania tych wyników.

Idea

Kodowanie

$$(S_1, \ldots, S_k) \to \bigcup_{i=1}^{k} p \cdot S_i - d_i \ .$$

Jedna zmienna

Problem

Ile zmiennych potrzeba do uzyskania tych wyników.

Idea

Kodowanie

$$(S_1, \ldots, S_k) \to \bigcup_{i=1}^{k} p \cdot S_i - d_i \ .$$

- EXPTIME trudność należenia do minimalnego rozwiązania $X = \varphi(X)$

Jedna zmienna

Problem

Ile zmiennych potrzeba do uzyskania tych wyników.

Idea

Kodowanie

$$(S_1, \ldots, S_k) \to \bigcup_{i=1}^{k} p \cdot S_i - d_i \ .$$

- EXPTIME trudność należenia do minimalnego rozwiązania $X = \varphi(X)$
- najmniejsze rozwiązanie $\varphi(X) = \psi(X)$ — RE-trudne

Jedna zmienna

Problem

Ile zmiennych potrzeba do uzyskania tych wyników.

Idea

Kodowanie

$$(S_1, \ldots, S_k) \to \bigcup_{i=1}^{k} p \cdot S_i - d_i \ .$$

- EXPTIME trudność należenia do minimalnego rozwiązania $X = \varphi(X)$
- najmniejsze rozwiązanie $\varphi(X) = \psi(X)$ — RE-trudne
- problemy decyzyjne, itd.