Theoretische Informatik

Vorlesung 02
Grundbegriffe formaler Sprachen, Endliche Automaten

Matthias Heizmann

Professur Softwaretechnik
Albert-Ludwigs-University Freiburg

19. Oktober, 2018

Vorlesungsinhalt

1. Formale Sprachen und Automatentheorie
   - Endliche Automaten
   - Reguläre Ausdrücke
   - Grammatiken
   - Automaten mit Stapelspeicher
   - Turingmaschinen
2. Berechenbarkeitstheorie
   - Nicht berechenbare Funktionen
   - Unentscheidbare Probleme
   - Reduktion
3. Komplexitätstheorie
   - Berechnungskomplexität auf Turingmaschinen
   - Komplexitätsklassen P und NP

Formale Sprachen und Automatentheorie

"Rechnen mit Zeichenketten und Mengen von Zeichenketten"

	typische Objekte	typische Rechenoperationen
lineare Algebra	$\mathbb{Q}$,$\mathbb{R}$ Vektorräume,	$+, -, \cdot, \backslash\, ^2$ berechne Determinante
Geometrie	$\mathbb{Q}$,$\mathbb{R}$ Flächen, Winkel	$+, -, \cdot$, *sin*, *cos* berechne Flächeninhalt, berechne Radius
formale Sprachen	Zeichenketten, Mengen von Zeichenketten	$\cap, \cup, \backslash, \in$ Konkatenation, Potenzierung, Stern,


() xayn — Konsole
matthias@xaYn ~/ultimate $ git puhs
git: 'puhs' is not a git command. See 'git --help'.
The most similar command is
push
matthias@xaYn ~/ultimate $

Vorspann: Formale Sprachen

Grundbegriffe, Notationen

Alphabet Σ

Ein *Alphabet* ist eine endliche Menge von *Zeichen*.

Zeichen sind hier beliebige abstrakte Symbole.

Alphabet Σ

Ein *Alphabet* ist eine endliche Menge von *Zeichen*.

Zeichen sind hier beliebige abstrakte Symbole.

Bsp

für Alphabete, die in dieser Vorlesung, im täglichen Umgang mit Computern oder in der Forschung an unserem Lehrstuhl eine Rolle spielen

- $\{a, \ldots, z\}$
- $\{0, 1\}$
- $\{rot, gelb, grün\}$ (Ampelfarben)
- Die Menge aller ASCII-Symbole
- Die Menge aller Statements eines Computerprogramms

Wort w über Σ

Wir nennen eine endliche Folge von Elementen aus Σ ein *Wort*.

Wir schreiben solche Folge ohne Trennsymbole.

z.B. `einhorn` statt `e,i,n,h,o,r,n`

Wort w über Σ

Wir nennen eine endliche Folge von Elementen aus Σ ein *Wort*.

Wir schreiben solche Folge ohne Trennsymbole.

z.B. `einhorn` statt `e,i,n,h,o,r,n`

Leeres Wort ε

Die leere Folge nennen wir das *leere Wort*.

Wort w über Σ

Wir nennen eine endliche Folge von Elementen aus Σ ein *Wort*.

Wir schreiben solche Folge ohne Trennsymbole.

z.B. `einhorn` statt `e,i,n,h,o,r,n`

Leeres Wort ε

Die leere Folge nennen wir das *leere Wort*.

Notation:

- Menge aller Wörter: Σ^*
- Menge aller nicht leeren Wörter: Σ^+

Wort w über Σ

Wir nennen eine endliche Folge von Elementen aus Σ ein *Wort*.

Wir schreiben solche Folge ohne Trennsymbole.

z.B. `einhorn` statt `e,i,n,h,o,r,n`

Leeres Wort ε

Die leere Folge nennen wir das *leere Wort*.

Notation:

- Menge aller Wörter: Σ^*
- Menge aller nicht leeren Wörter: Σ^+

Länge

Die *Länge* eines Wortes, $|\cdot| : \Sigma^* \to \mathbb{N}$, ist die Anzahl der Elemente der Folge.

Konkatenation von Wörtern

Die *Konkatenation*, $\cdot : \Sigma^* \times \Sigma^* \to \Sigma^*$, ist für $u = u_1 \dots u_n \in \Sigma^*$ und $v = v_1 \dots v_m \in \Sigma^*$ definiert durch: $u \cdot v = u_1 \dots u_n v_1 \dots v_m$

Konkatenation von Wörtern

Die *Konkatenation*, $\cdot : \Sigma^* \times \Sigma^* \to \Sigma^*$, ist für $u = u_1 \dots u_n \in \Sigma^*$ und $v = v_1 \dots v_m \in \Sigma^*$ definiert durch: $u \cdot v = u_1 \dots u_n v_1 \dots v_m$

- assoziativ
- nicht kommutativ!
- wir dürfen Klammern weglassen
- wir dürfen Operationssymbol weglassen

Konkatenation von Wörtern

Die *Konkatenation*, $\cdot : \Sigma^* \times \Sigma^* \to \Sigma^*$, ist für $u = u_1 \dots u_n \in \Sigma^*$ und $v = v_1 \dots v_m \in \Sigma^*$ definiert durch: $u \cdot v = u_1 \dots u_n v_1 \dots v_m$

- assoziativ
- nicht kommutativ!
- wir dürfen Klammern weglassen
- wir dürfen Operationssymbol weglassen

Definition

Die *Potenzierung* von Wörtern, $\cdot^{\cdot} : \Sigma^* \times \mathbb{N} \to \Sigma^*$, ist induktiv definiert durch

1. $w^0 = \varepsilon$
2. $w^{n+1} = w \cdot w^n$

Sprache über Σ

Eine *Sprache* über Σ ist eine Menge $L \subseteq \Sigma^*$.

Sprache über Σ

Eine *Sprache* über Σ ist eine Menge $L \subseteq \Sigma^*$.

Konkatenation und Potenzierung von Sprachen

Seien $U, V \subseteq \Sigma^*$. Dann ist die *Konkatenation* von U und V definiert durch

$$U \cdot V = \{u \cdot v \mid u \in U, v \in V\}$$

und die *Potenzierung* von U induktiv definiert durch

1. $U^0 = \{\varepsilon\}$
2. $U^{n+1} = U \cdot U^n$

Sprache über Σ

Eine *Sprache* über Σ ist eine Menge $L \subseteq \Sigma^*$.

Konkatenation und Potenzierung von Sprachen

Seien $U, V \subseteq \Sigma^*$. Dann ist die *Konkatenation* von U und V definiert durch

$$U \cdot V = \{u \cdot v \mid u \in U, v \in V\}$$

und die *Potenzierung* von U induktiv definiert durch

1. $U^0 = \{\varepsilon\}$
2. $U^{n+1} = U \cdot U^n$

Kleene-Abschluss, Kleene-Stern

Sei $U \subseteq \Sigma^*$. Der *Kleene-Abschluss* ist definiert durch

1. $U^* = \bigcup_{n \in \mathbb{N}} U^n \quad [\ni \varepsilon]$
2. $U^+ = \bigcup_{n \geq 1} U^n$

Kapitel 2:
Reguläre Sprachen und endliche Automaten

Endliche Automaten

Herausforderung 1

Wie können wir eine unendlich große Sprache mit Hilfe von endlich viel Speicherplatz darstellen?

Beispiel

Sprache L_{even}: Die Menge aller Wörter über dem Alphabet $\Sigma = \{0, 1\}$ die eine gerade Anzahl Einsen enthalten.

- Natürliche Sprache. (Beschreibung der Sprache ist bereits eine Darstellung die nur endlich viel Speicherplatz benötigt.)
- Mix aus mathematischen Symbolen und natürlicher Sprache ("Mengennotation").

$$L_{\text{even}} = \{w \in \{0,1\}^* \mid \text{ die Anzahl der Einsen in } w \text{ ist gerade. } \}$$

Das Wortproblem

Sei $L \subset \Sigma^*$ eine beliebige Sprache und $w \in \Sigma^*$ ein beliebiges Wort. Wir nennen das Entscheidungsproblem

$$w \in L?$$

das *Wortproblem*.

Ein Entscheidungsverfahren für das Wortproblem würde also

$$01001 \in L_{\text{even}}$$

mit "ja" und

$$010101 \in L_{\text{even}}$$

mit "nein" beantworten.

Endliche Automaten

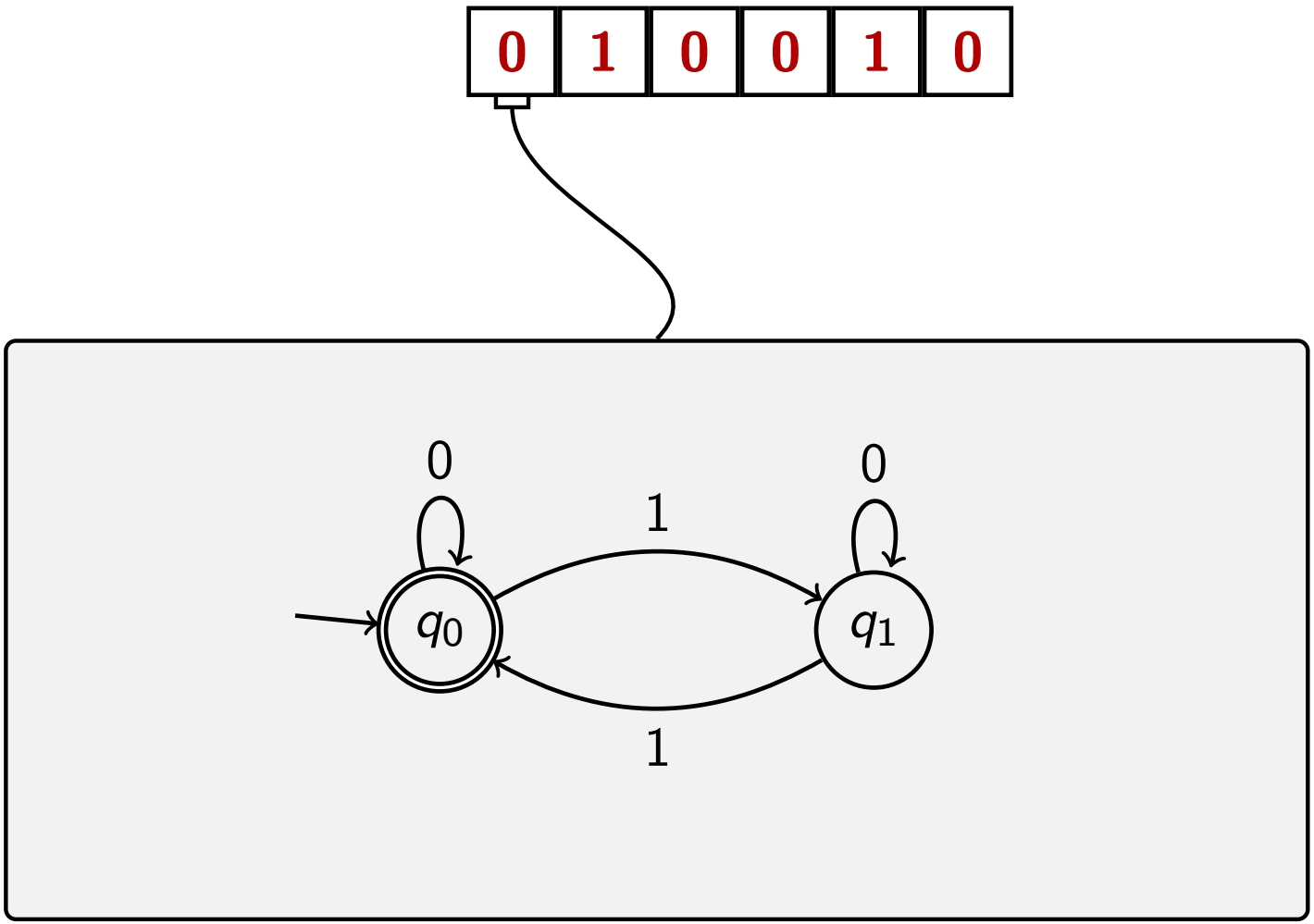
Herausforderung 2

Wie können wir eine unendlich große Sprache mit Hilfe von endlich viel Speicherplatz, so darstellen dass wir das Wortproblem mit einem "einfachen" Algorithmus lösen können?

- Idee 1: Natürliche Sprache
  Problem: Algorithmus müsste natürliche Sprache verstehen.
- Idee 2: Mengennotation
  Problem: Algorithmus müsste natürliche Sprache verstehen oder wir müssten zulässige Texte stark einschränken.
- Idee 3: Endlicher Automat

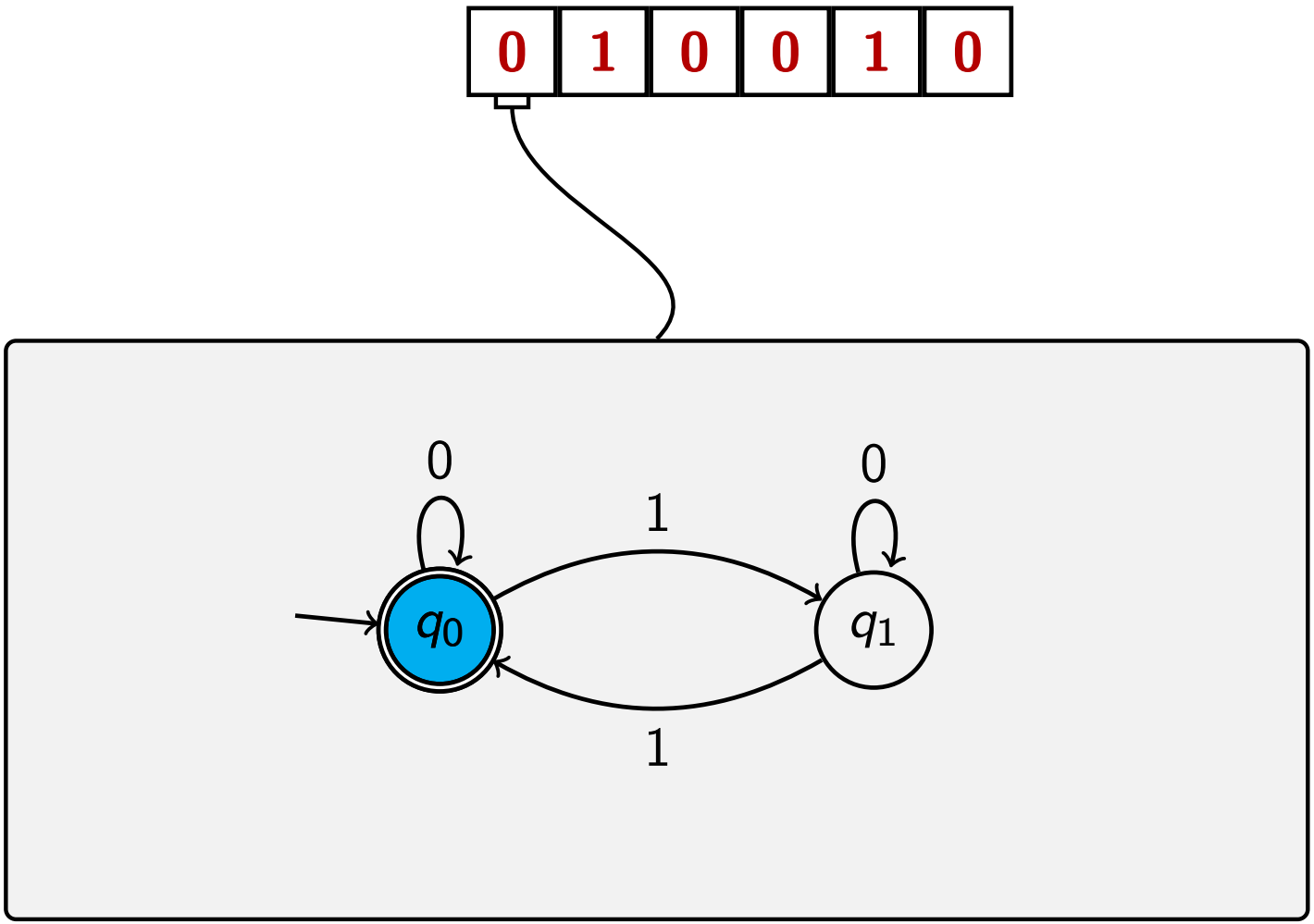
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



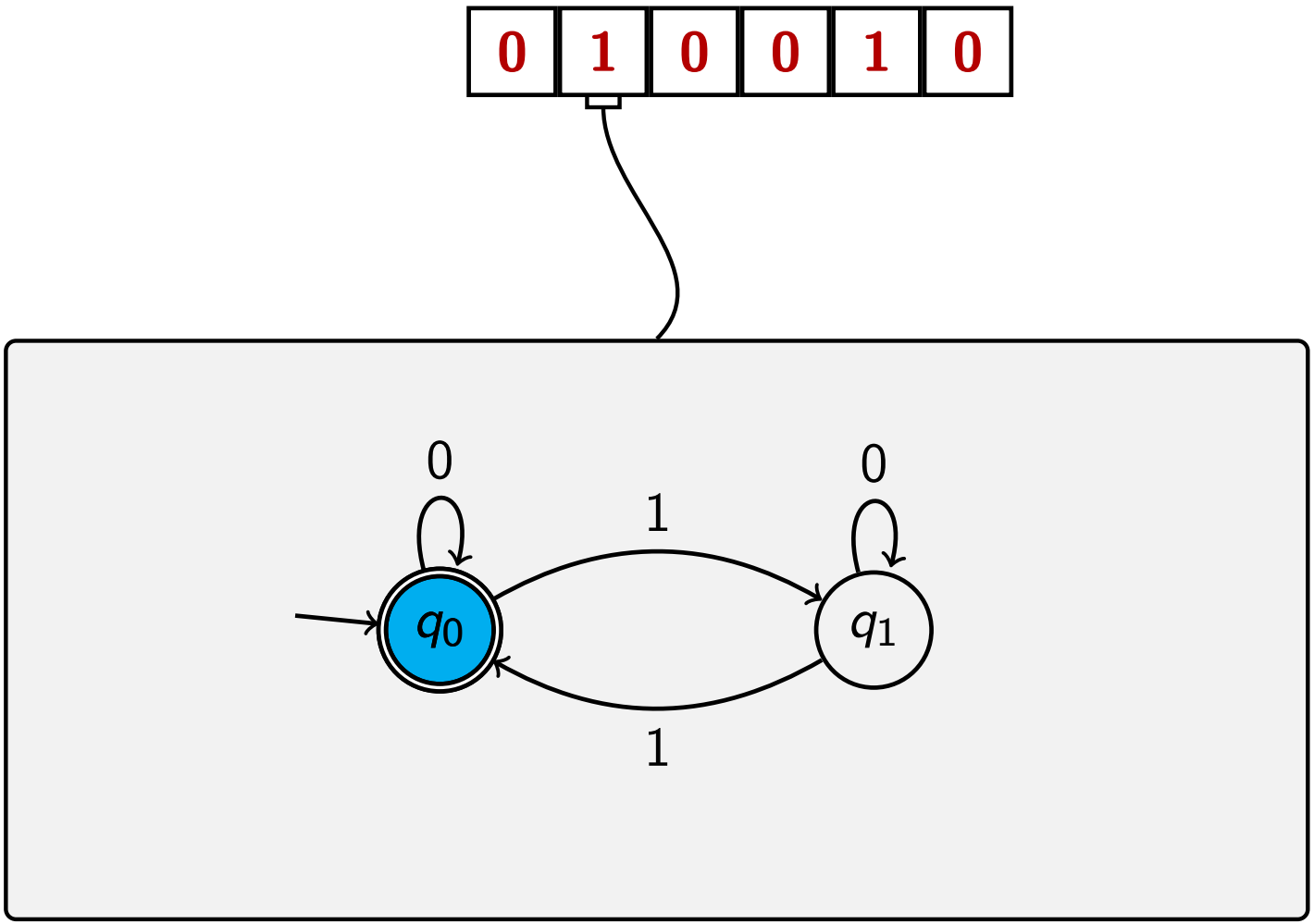
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



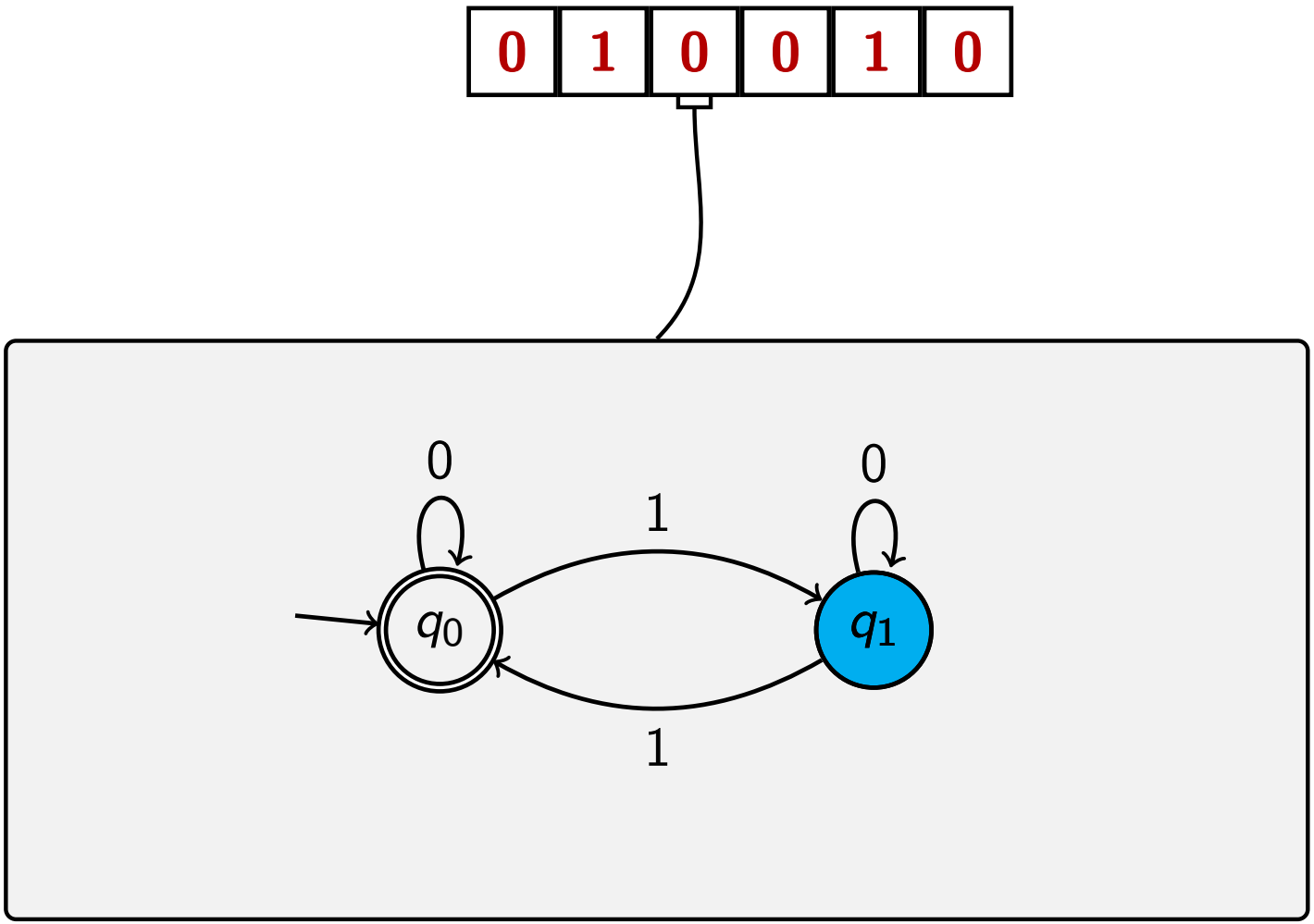
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



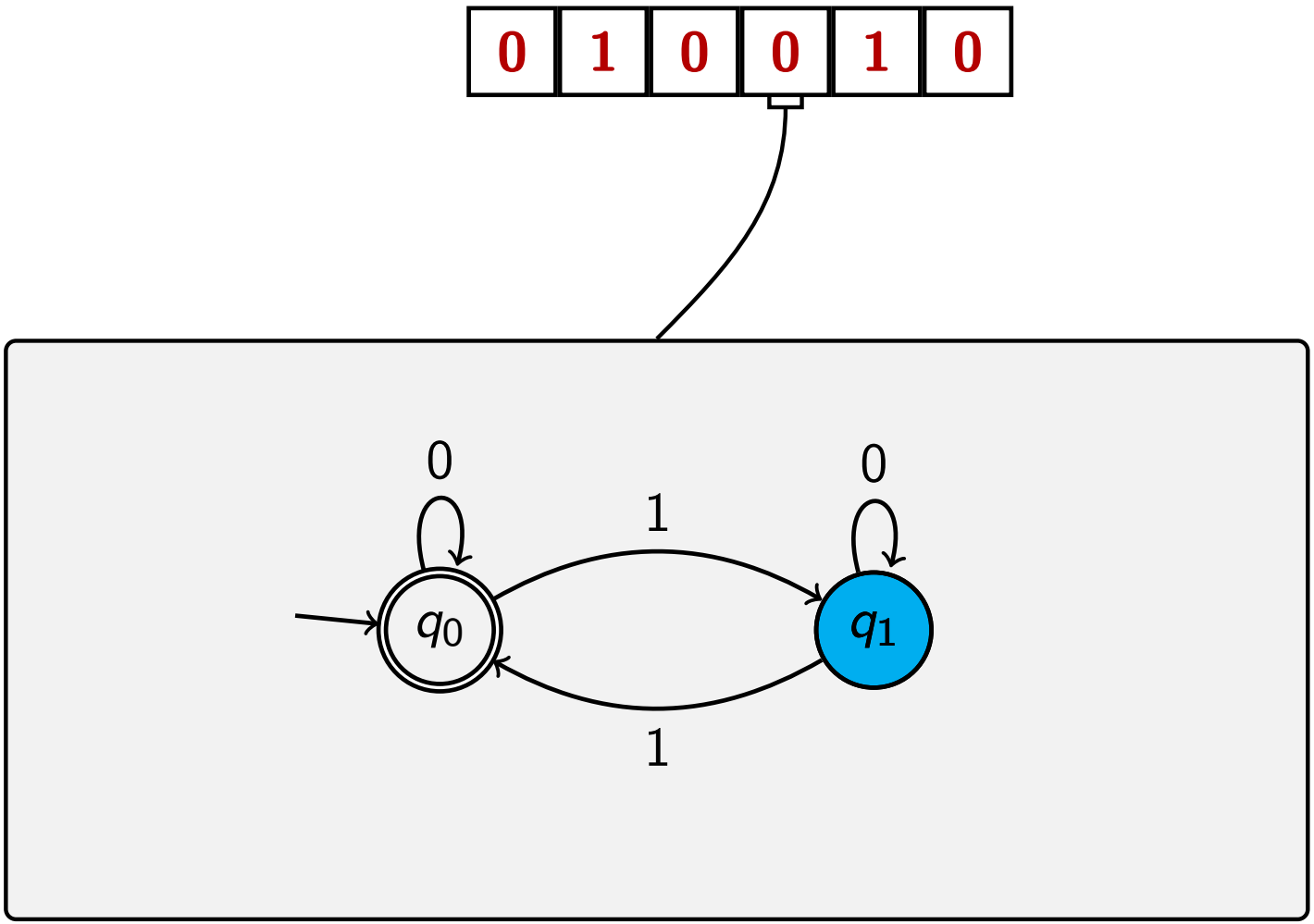
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid \text{ die Anzahl der Einsen in } w \text{ ist gerade. } \}$



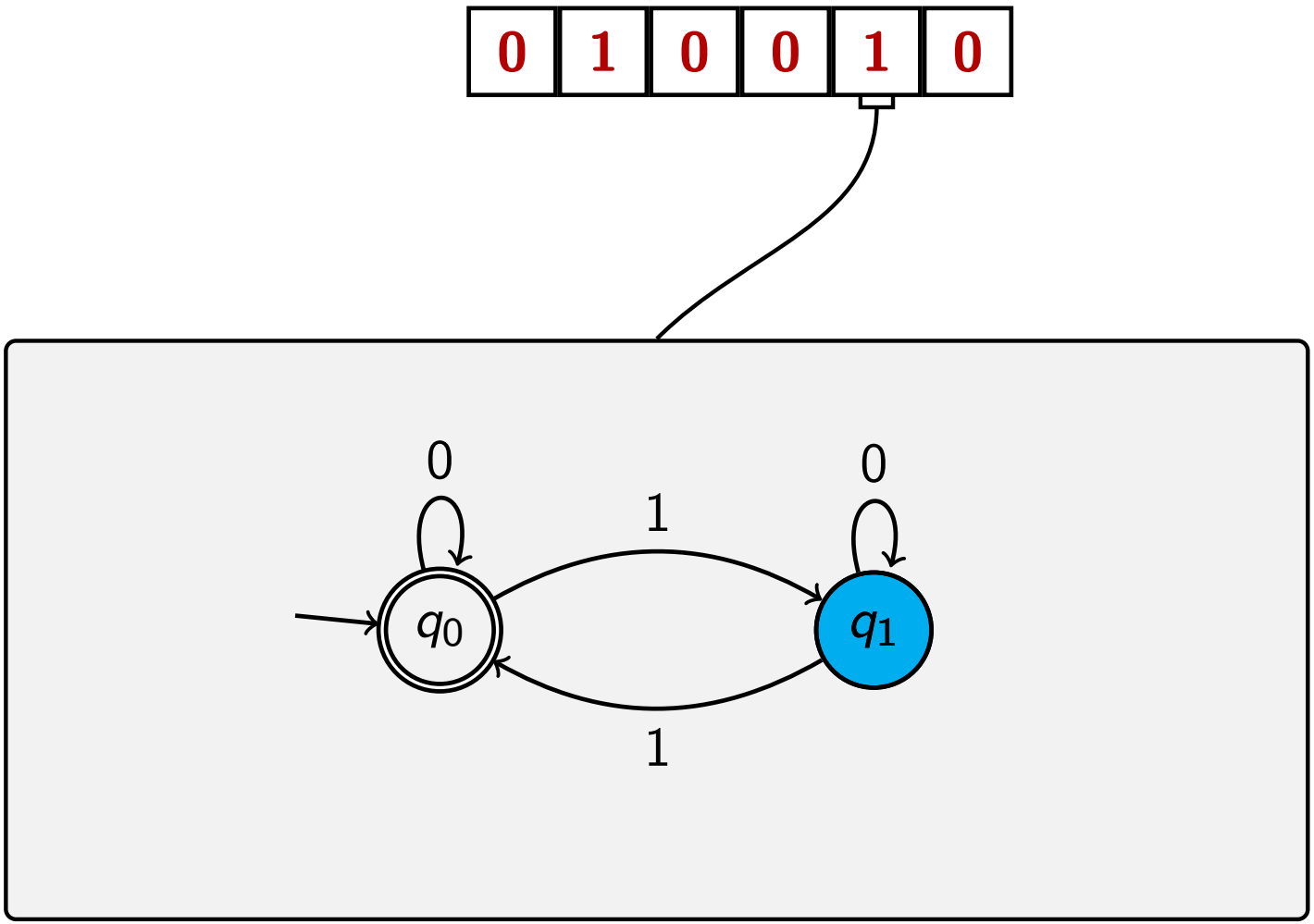
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid \text{ die Anzahl der Einsen in } w \text{ ist gerade. } \}$



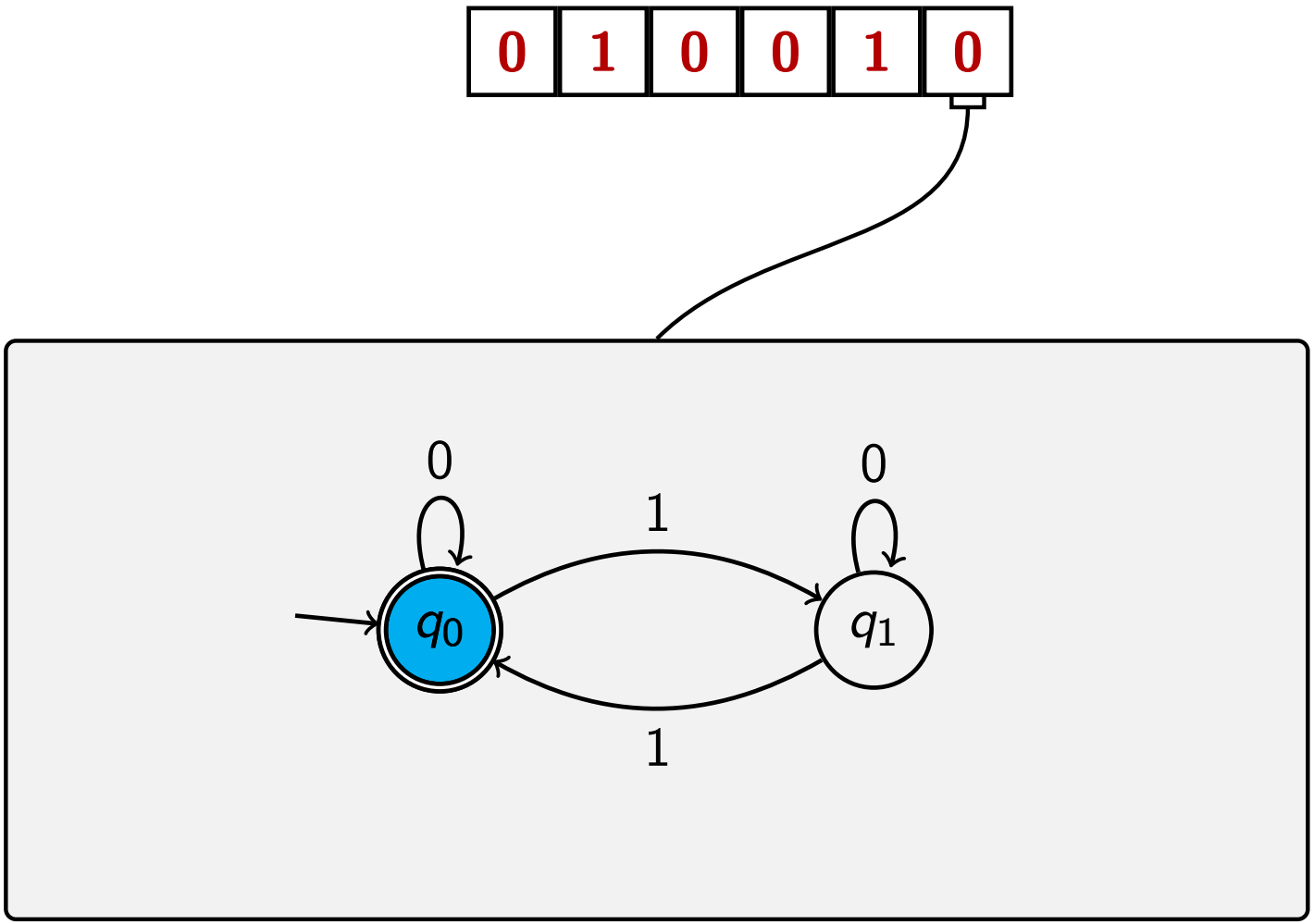
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



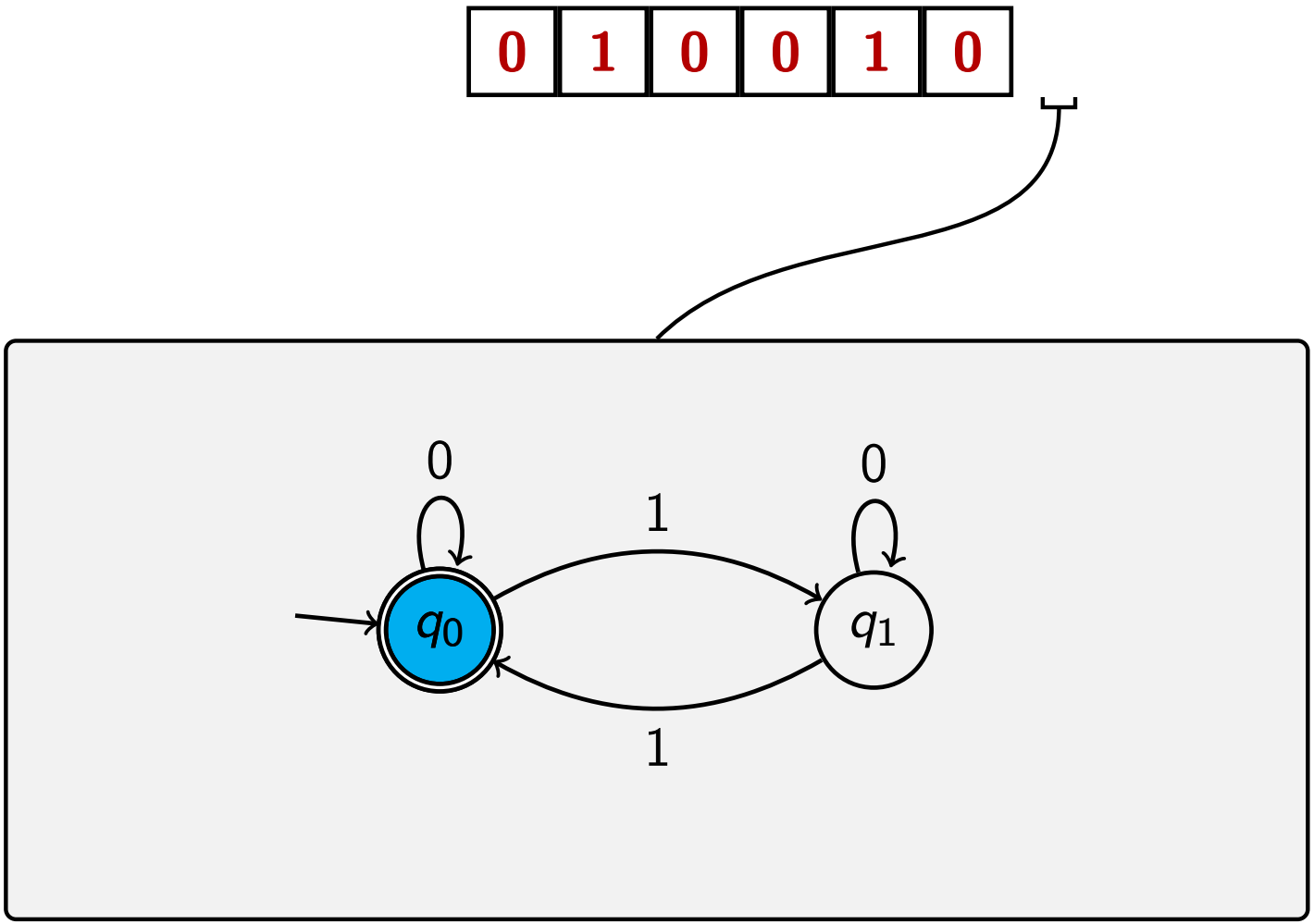
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



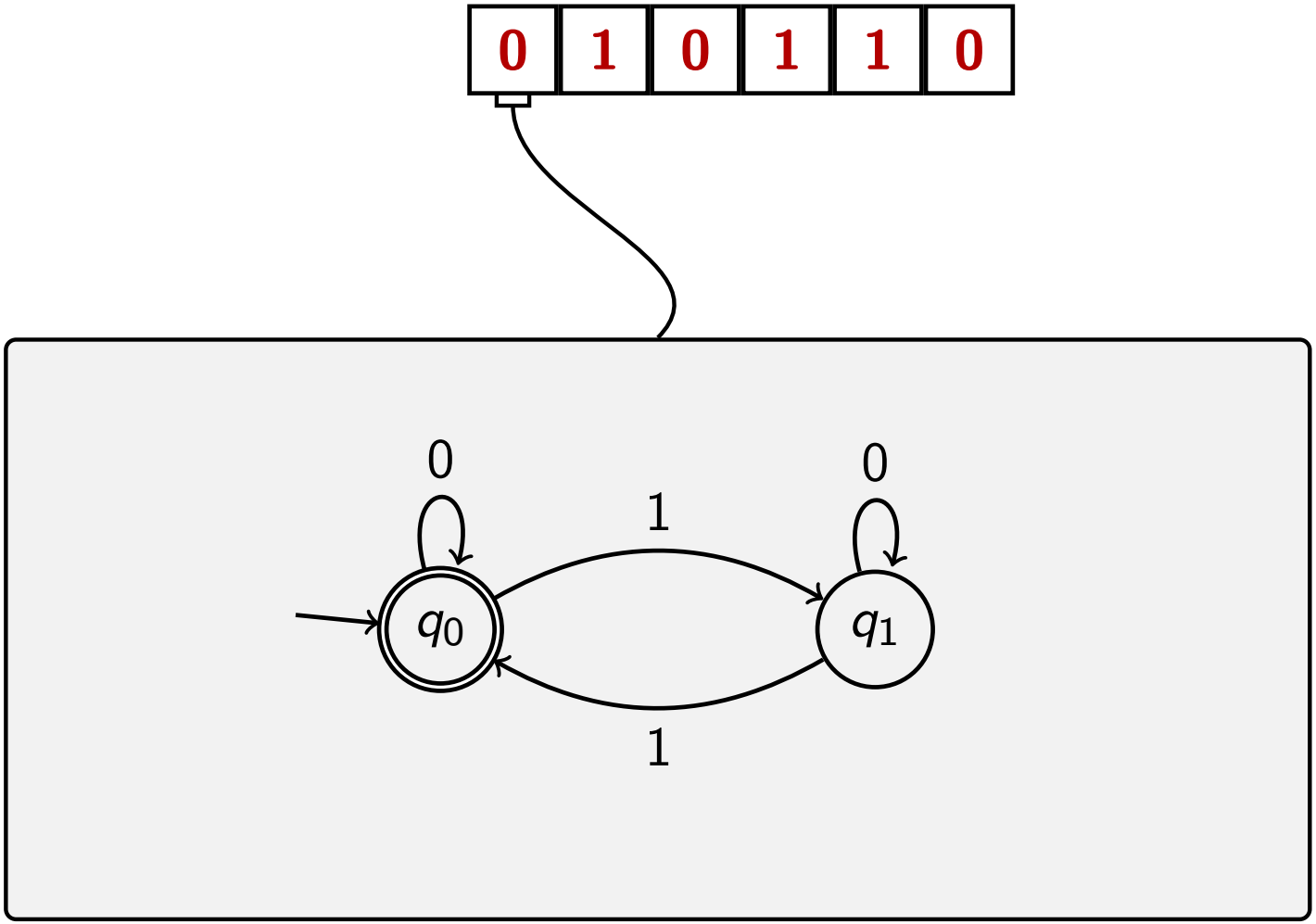
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



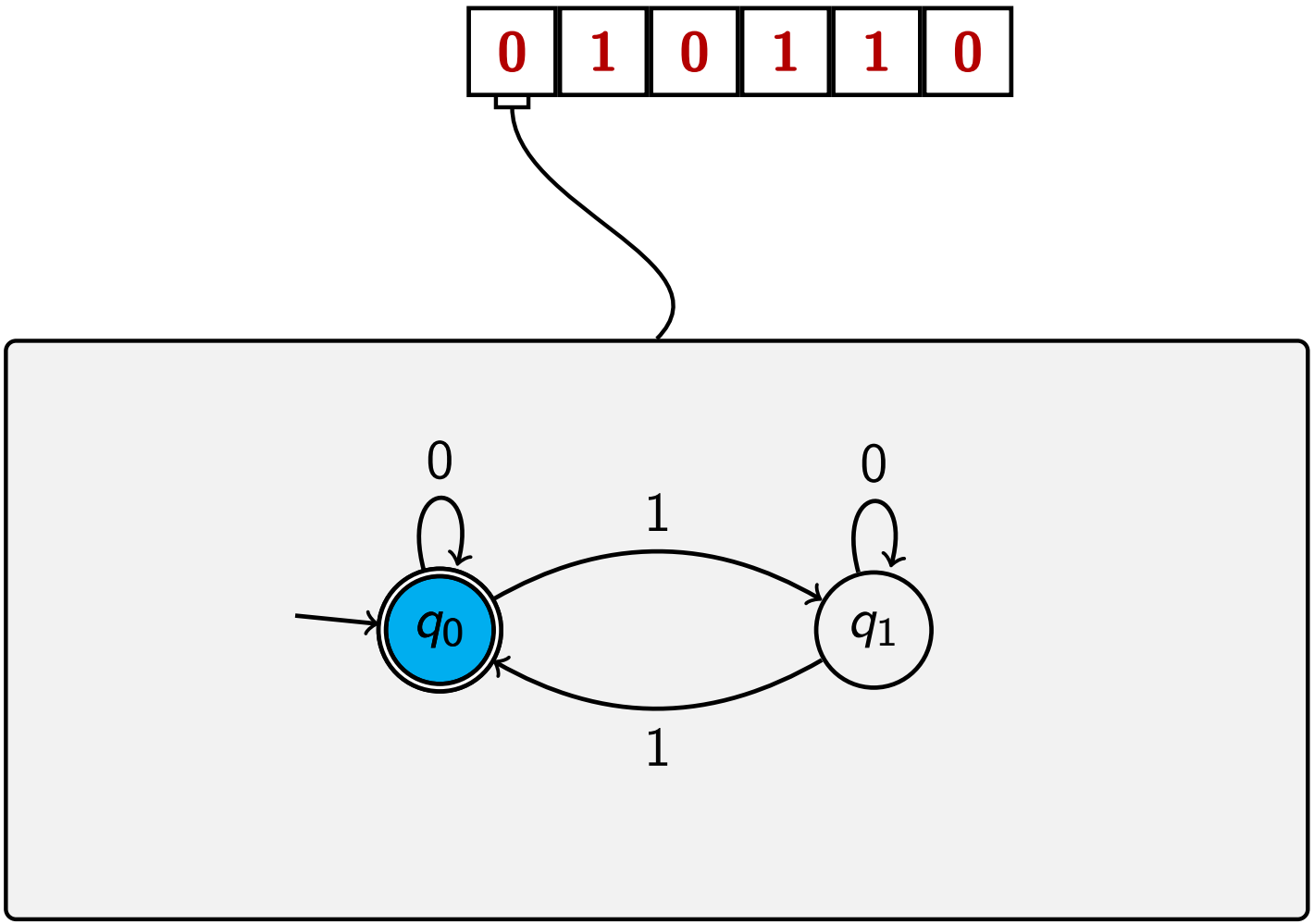
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



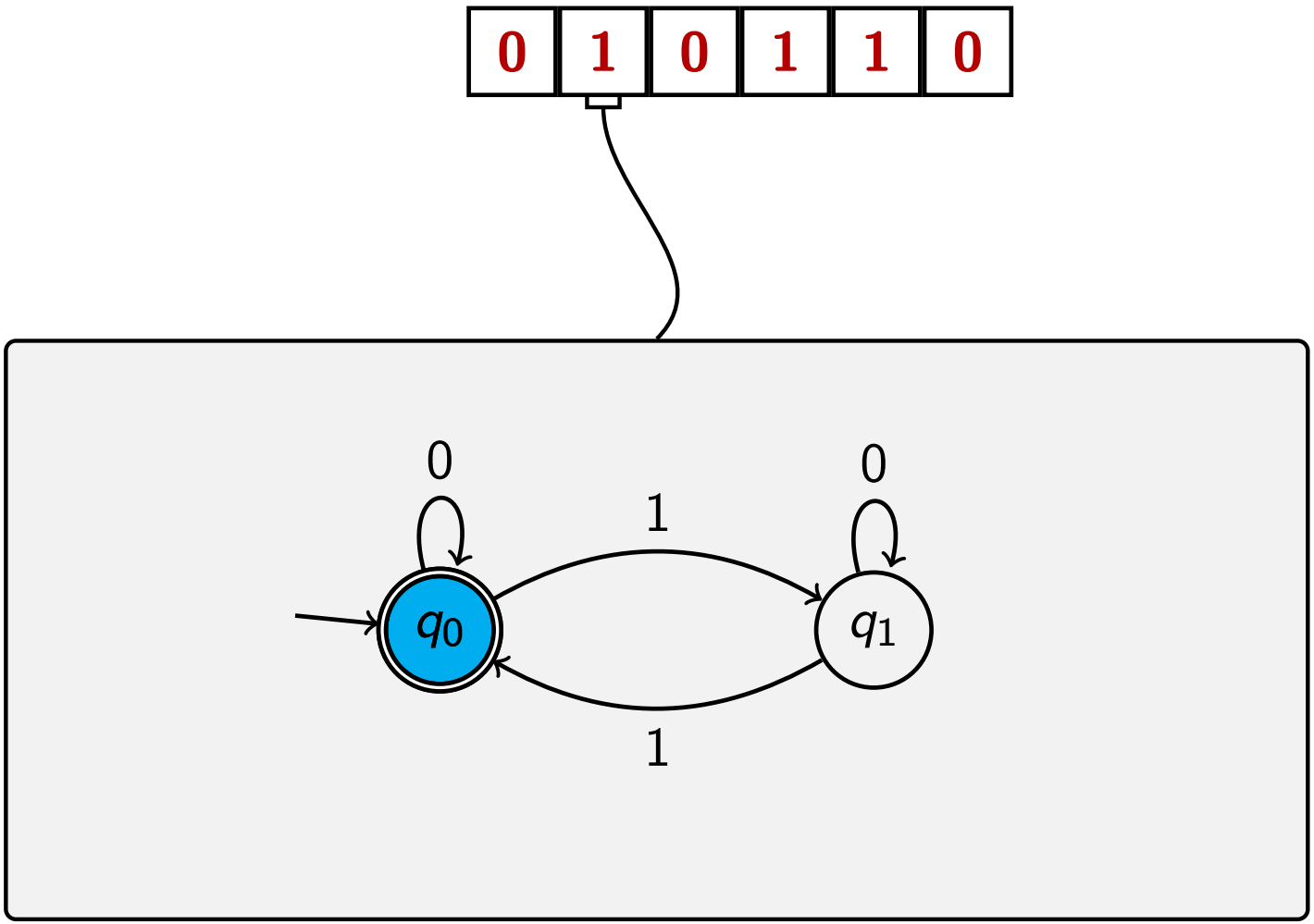
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



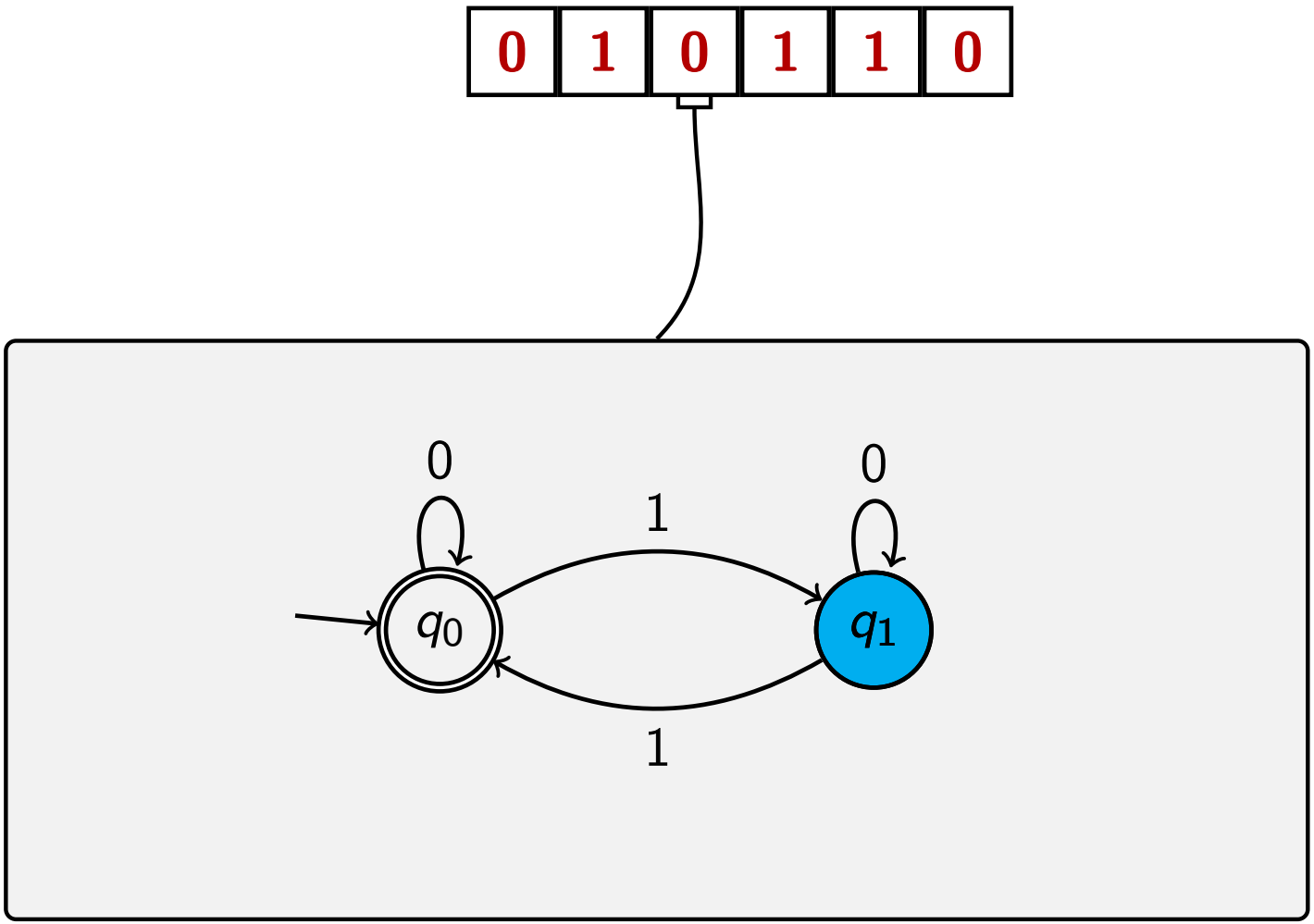
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



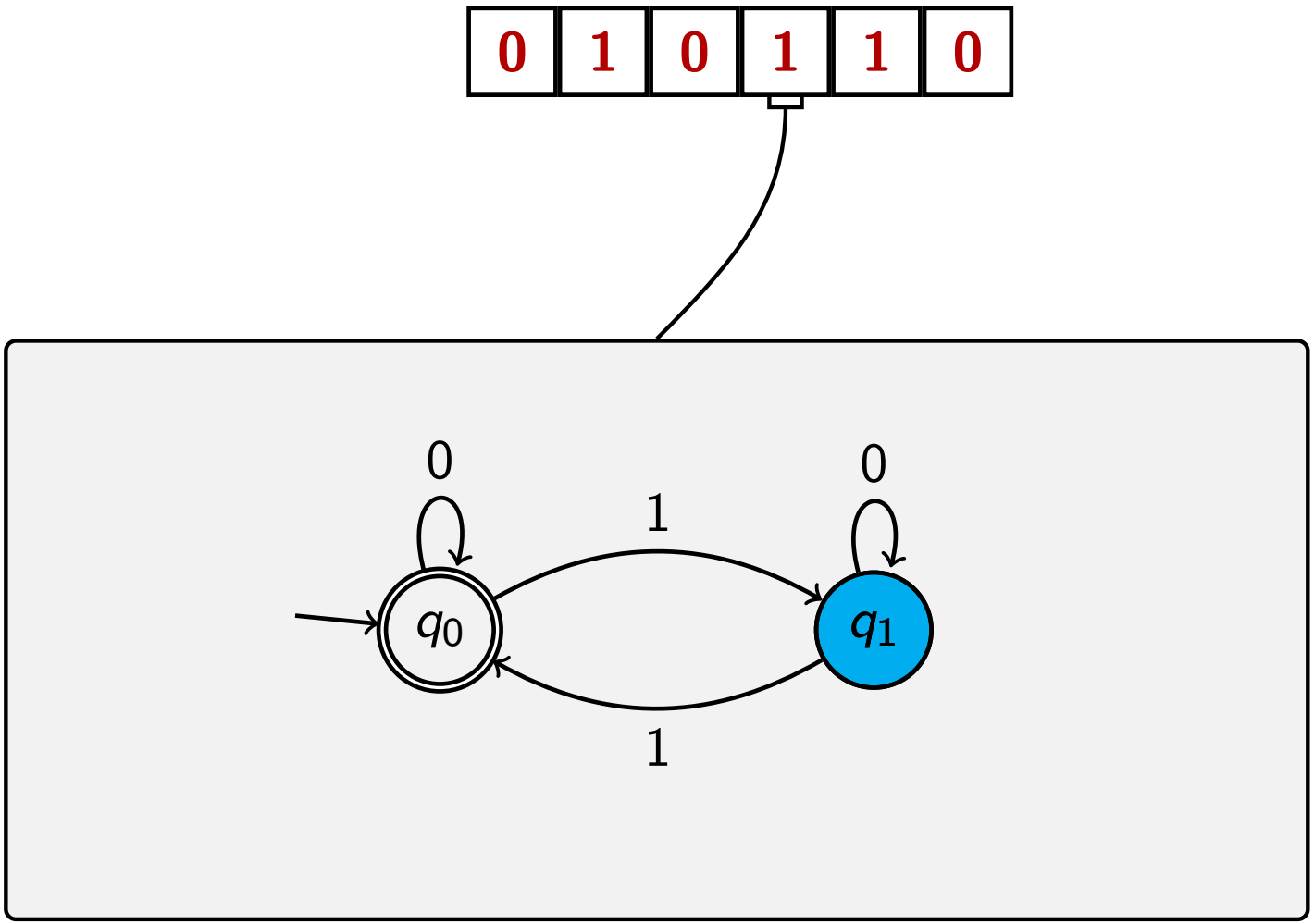
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid$ die Anzahl der Einsen in w ist gerade. $\}$



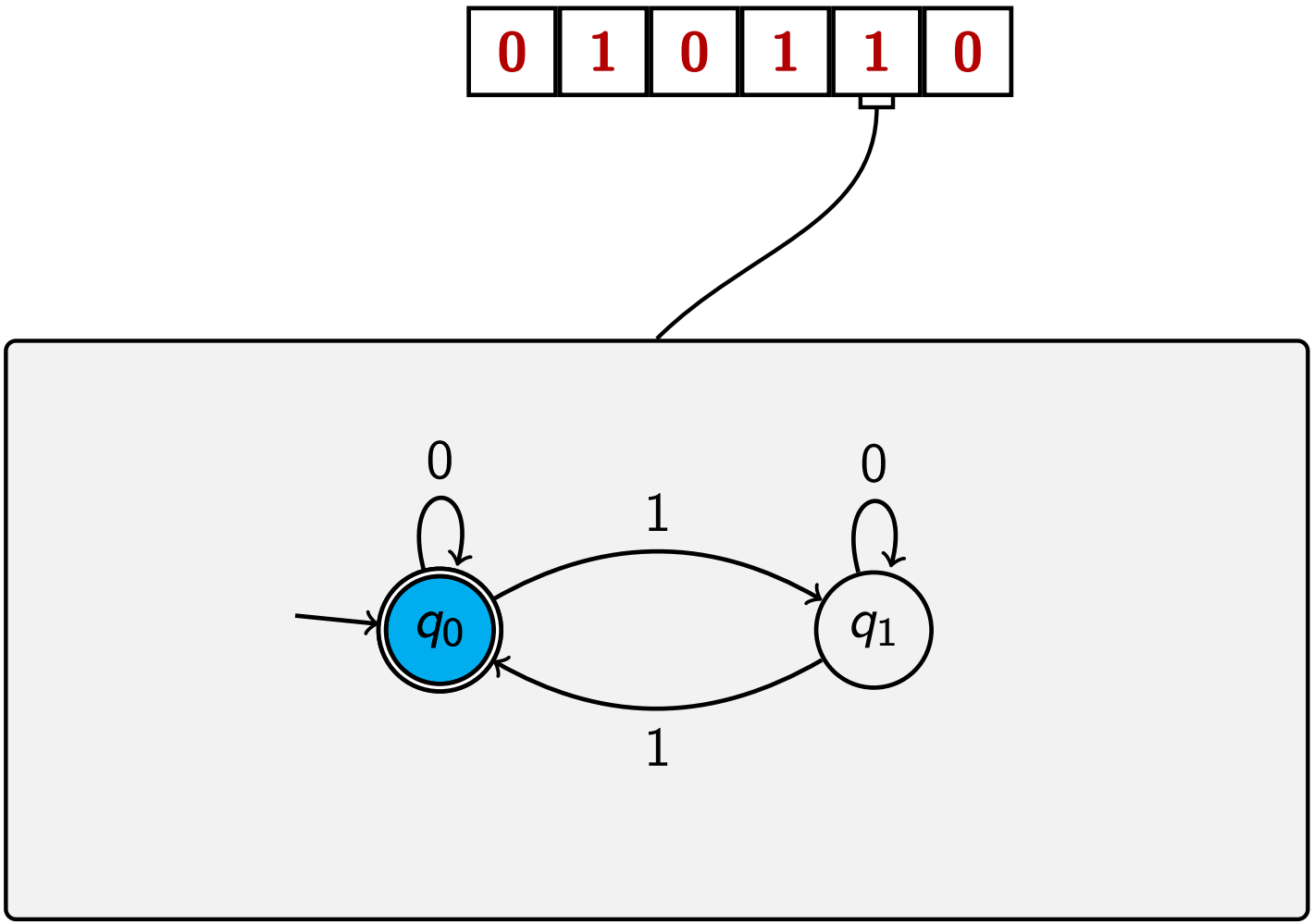
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid \text{ die Anzahl der Einsen in } w \text{ ist gerade. } \}$



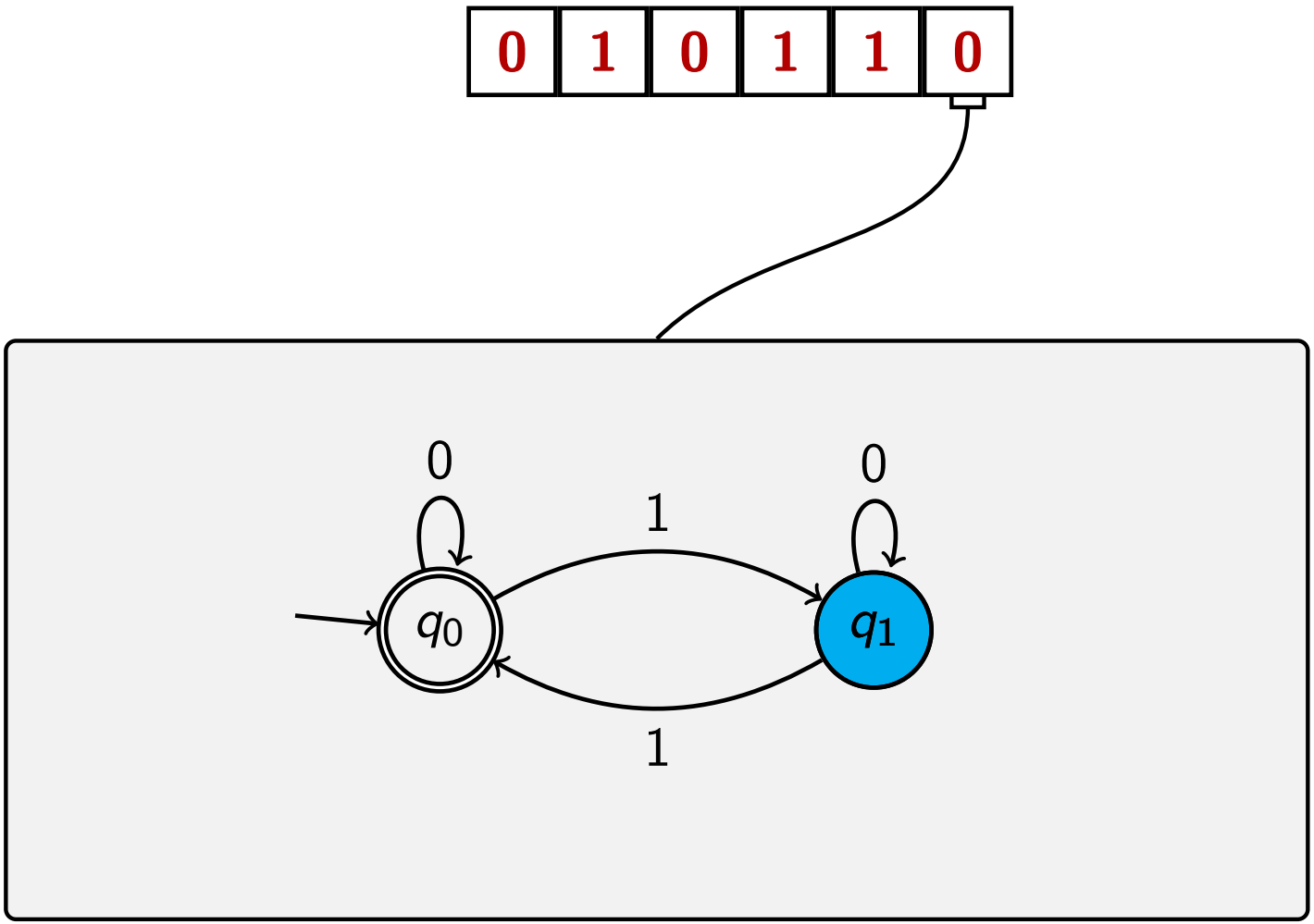
Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid \text{ die Anzahl der Einsen in } w \text{ ist gerade. } \}$



Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid \text{ die Anzahl der Einsen in } w \text{ ist gerade. } \}$



Endlicher Automat für die Sprache

$L_{\text{even}} = \{w \in \{0,1\}^* \mid \text{ die Anzahl der Einsen in } w \text{ ist gerade. } \}$