Politechnika Wrocławska

# Struktury danych i złożoność obliczeniowa
# Wykład 4

Prof. dr hab. inż. Jan Magott

# Racjonalny” model komputera

**„Racjonalny” model komputera** to taki, dla którego istnieje wielomian ograniczający od góry zakres pracy wykonywanej w jednostce czasu.

Model mogący wykonywać dowolnie dużą liczbę operacji równolegle nie jest „racjonalnym”.

# Modele algorytmów

Program w języku wysokiego poziomu

Kierunek wzrostu poziomu abstrakcji

Program w języku asemblera

Zbliżony poziom abstrakcji

Program dla maszyny RAM

Program dla wielotaśmowej maszyny Turinga

Program dla jednotaśmowej maszyny Turinga

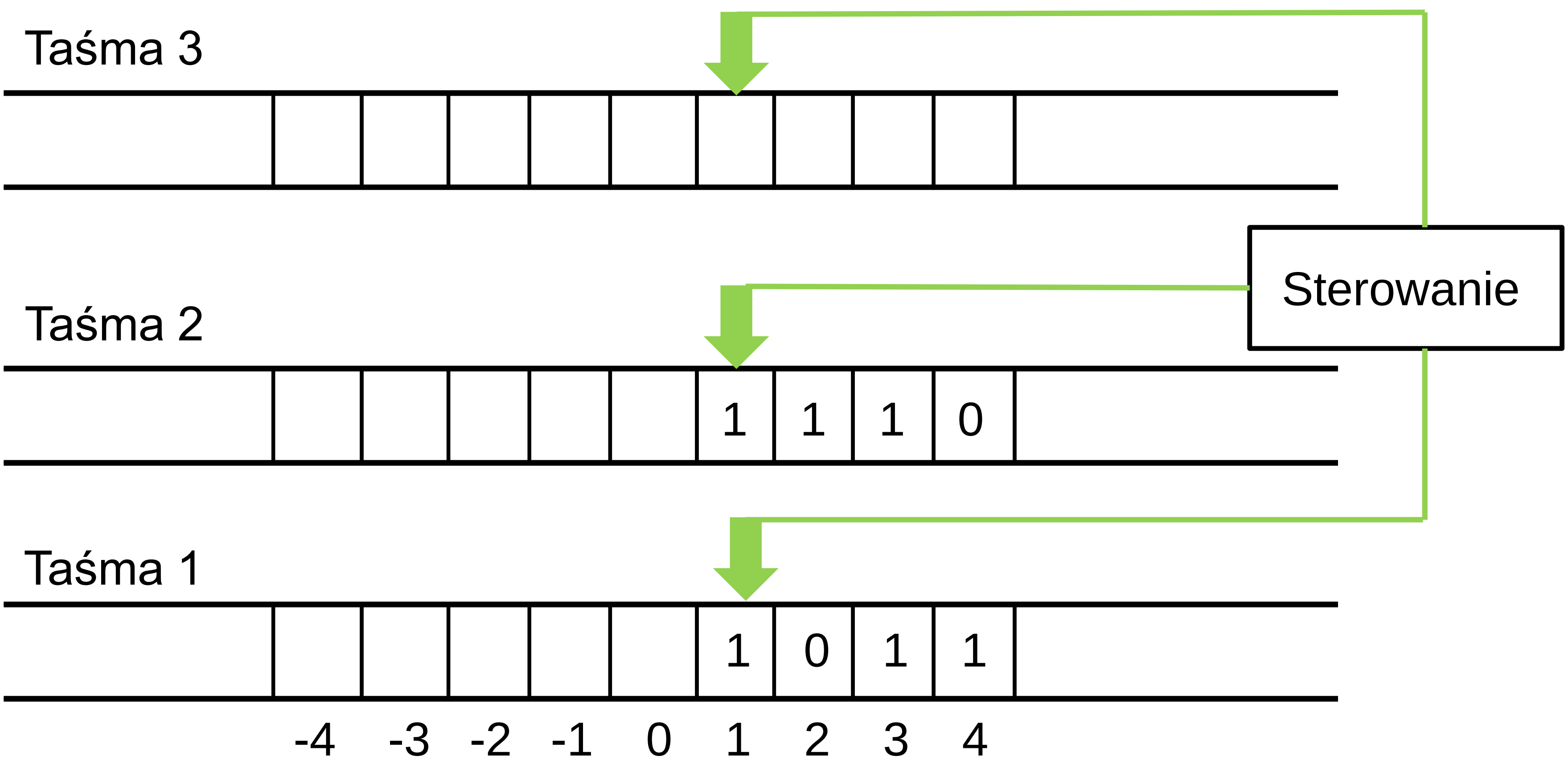
# Model *RAM* (ang. random access machine)

- Jeden procesor,
- Operacje elementarne (zapisania, dodawania, odejmowania, porównania dwu liczb, itp.) wymagają jednego kroku czasowego,
- Taśma wejściowa z głowicą odczytującą,
- Taśma wyjściowa z głowicą zapisującą,
- Pamięć danych jest zbiorem rejestrów z wyróżnionym rejestrem – akumulatorem, w którym wykonywane są obliczenia,
- Dostęp do pamięci wymaga jednego kroku,
- Możliwość adresowania pośredniego,
- Program, który nie jest przechowywany w pamięci jest sekwencją rozkazów (nie podlega automodyfikacji),
- Licznik rozkazów.

# Model *RAM* c.d.

- Każda komórka taśm: wejściowej i wyjściowej oraz każdy rejestr może zawierać dowolną liczbę całkowitą.
- Po odczycie z komórki taśmy wejściowej (zapisie w komórce taśmy wyjściowej), głowica taśmy jest przesuwana o jedną pozycję w prawo. Treści wpisanej na taśmie wyjściowej nie można zmienić.
- Rozkazy (zbiór rozkazów nie jest precyzyjnie zdefiniowany, ale nie może zawierać instrukcji niespotykanych w rzeczywistych komputerach):
  - Arytmetyczne (+, -, x, /),
  - Wejścia-wyjścia,
  - Rozgałęzienia przepływu sterowania.

**Start:** Maszyna znajduje się w stanie początkowym $q_0$, natomiast głowice odczytują symbole z trzech taśm z komórek o numerze 1.



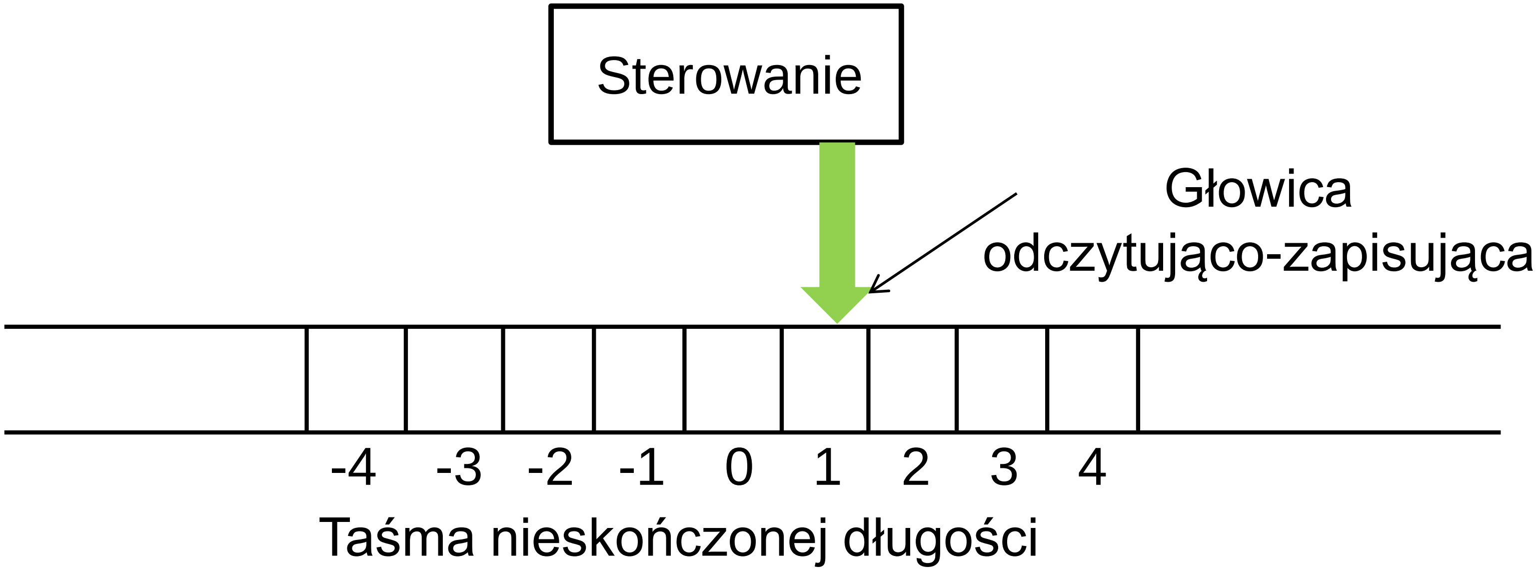
***Program dla k- taśmowej DMT*** składa się ze:

Skończonego zbioru symboli $\sum$ zawierającego separator (symbol pusty) $\sqcup$ zapisywanych/odczytywanych na/z $k$ taśm,

Skończonego zbioru stanów $Q = \{q_0, \dots, q_m\}$ zawierającego stan początkowy $q_0$, dwa wyróżnione stany końcowe $q_Y$ (odpowiedź „tak") i $q_N$ (odpowiedź „nie"),

Funkcji przejść $\delta\colon (Q\backslash\{q_Y, q_N\})\times \Sigma^k \rightarrow Q \times \Sigma^k \times \{-1{,}0, +1\}^k$.

# *Deterministyczna jednotaśmowa maszyna Turinga (DMT)*



***Program dla DMT*** składa się z:

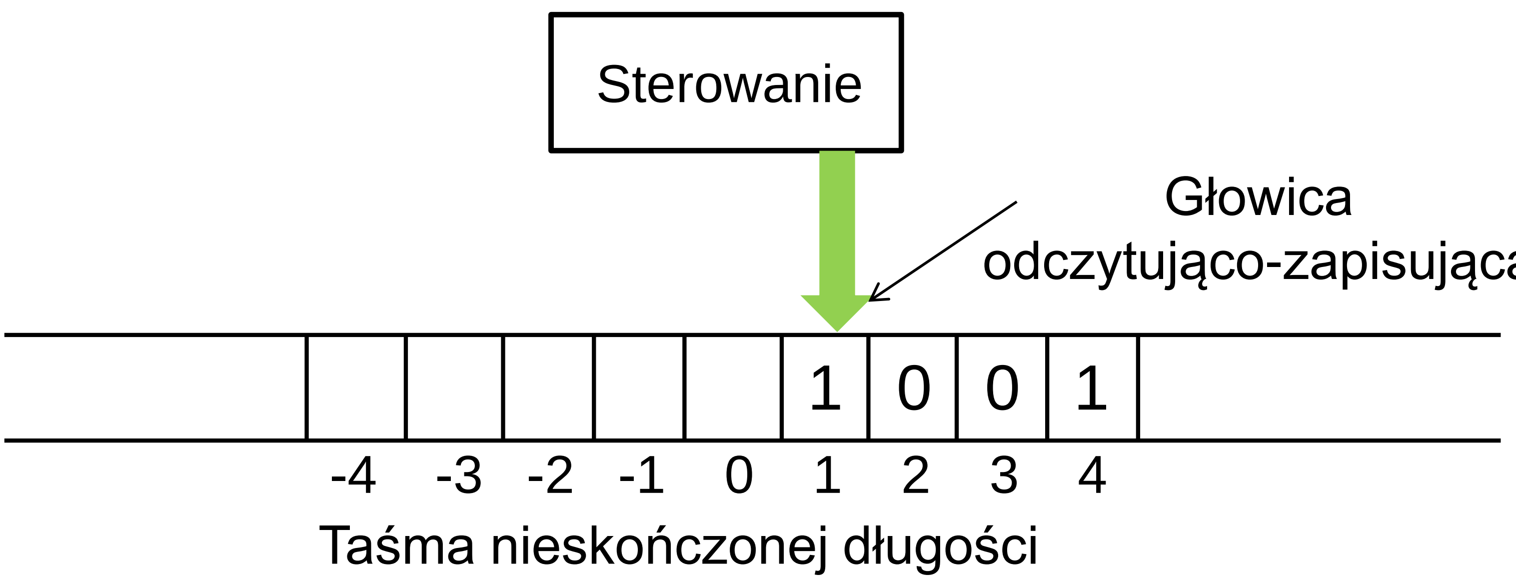
Skończonego zbioru symboli taśmy $\sum$ zawierającego separator (symbol pusty) $\sqcup$ ,

Skończonego zbioru stanów $Q = \{q_0, \dots, q_m\}$ zawierającego stan początkowy $q_0$, dwa wyróżnione stany końcowe $q_Y$ (odpowiedź „tak”) i $q_N$ (odpowiedź „nie”),

Funkcji przejść $\delta: (Q \backslash \{q_Y, q_N\}) \times \sum \quad \rightarrow Q \times \sum \times \{-1{,}0,+1\}$ .

## *Dane programu na DMT*

Pierwsze słowo wejściowe $x$ (skończony ciąg symboli) ze zbioru $\sum \setminus \{\sqcup\}$ jest zapisane po jednym symbolu w kolejnych komórkach taśmy o numerach od 1 do $|x|$. Kolejne słowa oddzielone są separatorami.

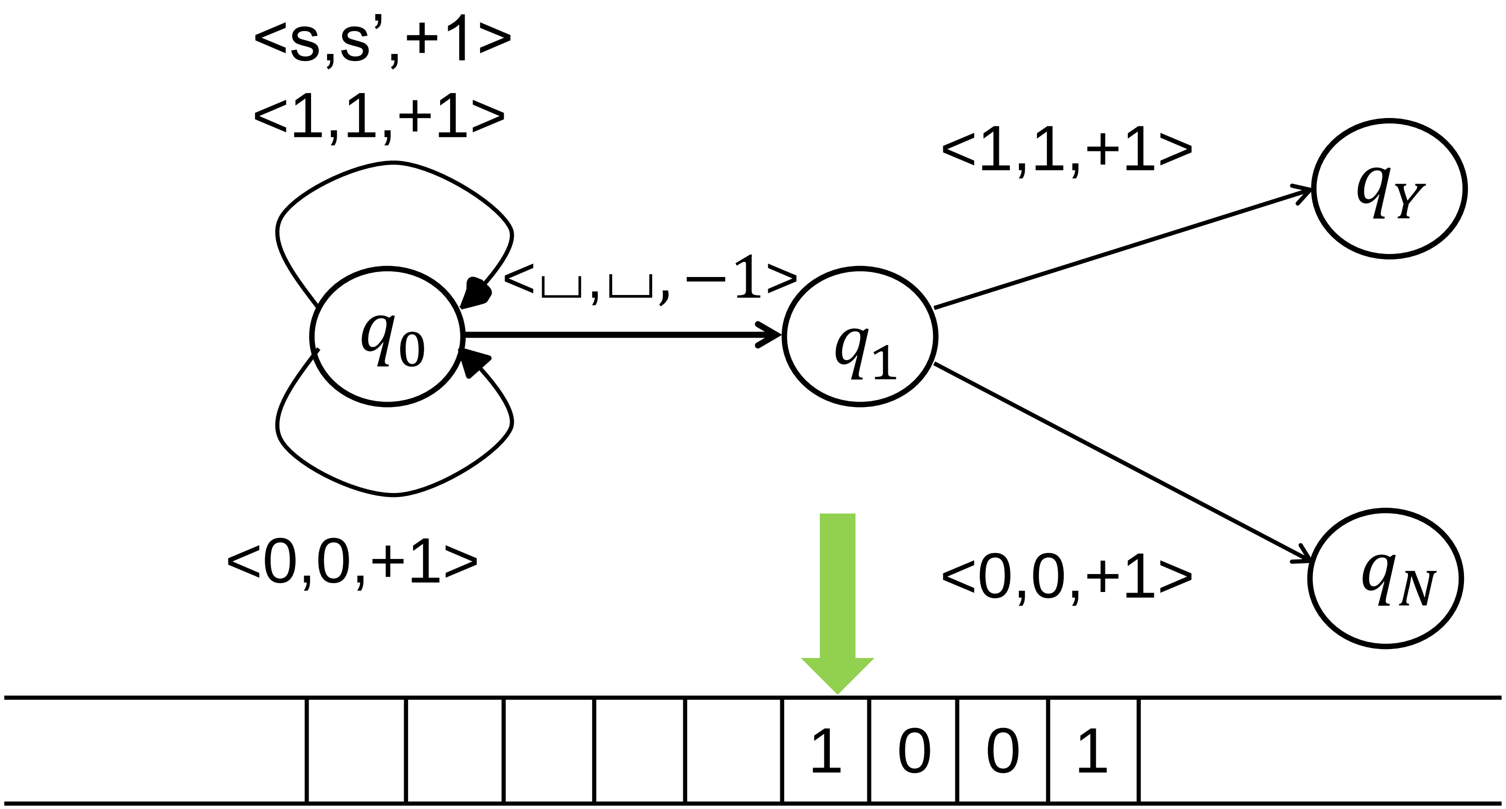


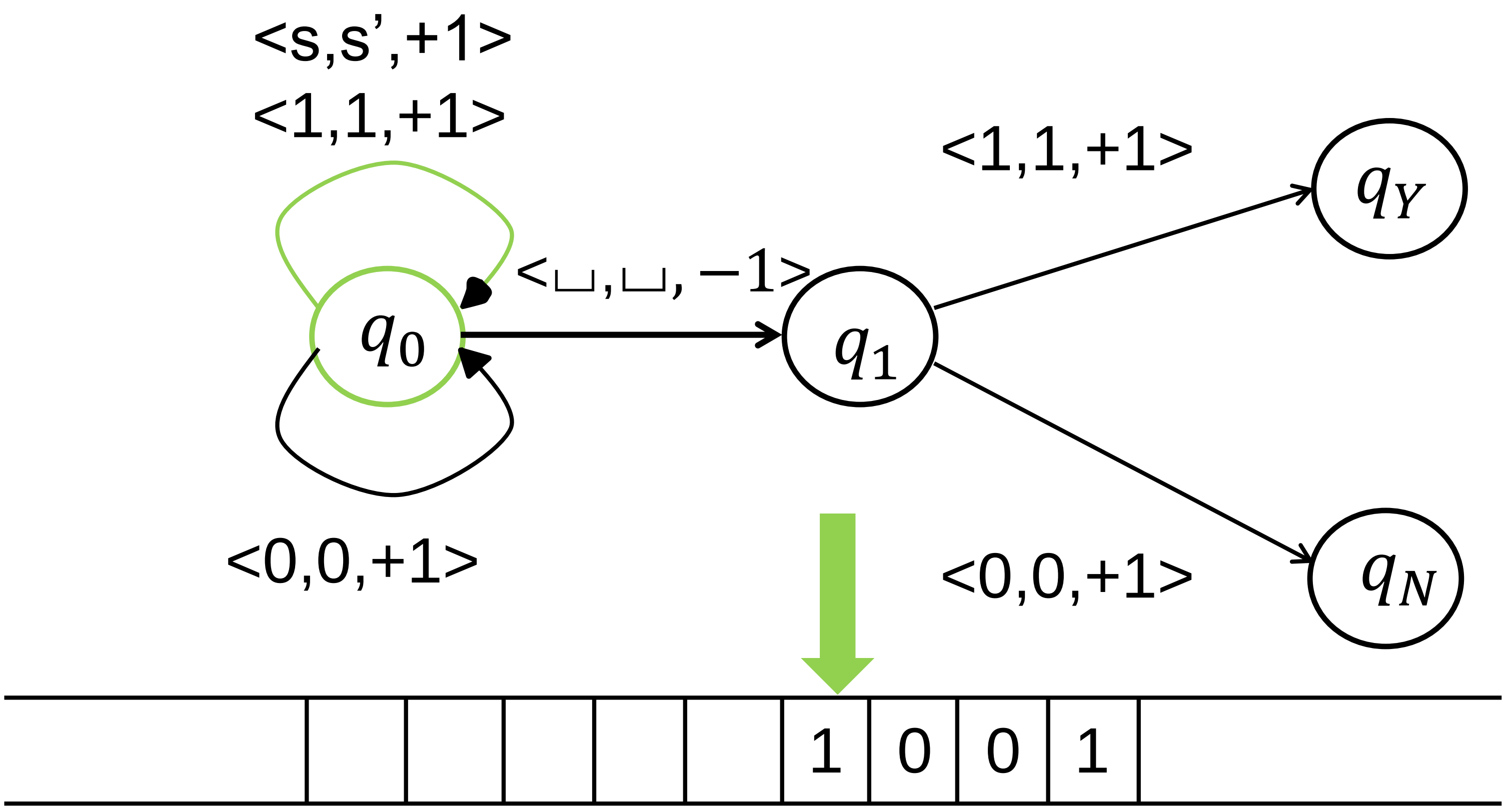
# *Wykonanie programu DMT*

1. Maszyna znajduje się w stanie początkowym $q_0$, natomiast głowica odczytuje symbol $s$ z komórki o numerze 1.
2. Jeśli DMT znajduje się w stanie $q \epsilon Q \backslash \{q_Y, q_N\}$, a w komórce, nad którą jest głowica jest symbol $s \epsilon \sum$, to maszyna wykonuje czynności określone funkcją przejścia $\delta(q, s) = (q', s', \Delta)$:
    1. Głowica w miejsce symbolu $s$ wpisuje $s'$,
    2. Głowica przesuwa się o jedną komórkę: w lewo jeśli $\Delta = -1$, w prawo jeśli $\Delta = +1$, lub nie zmienia pozycji jeśli $\Delta = 0$,
    3. Stan DTM się na $q'$.
3. Wykonywanie trwa do czasu gdy DTM znajdzie się w stanie końcowym.

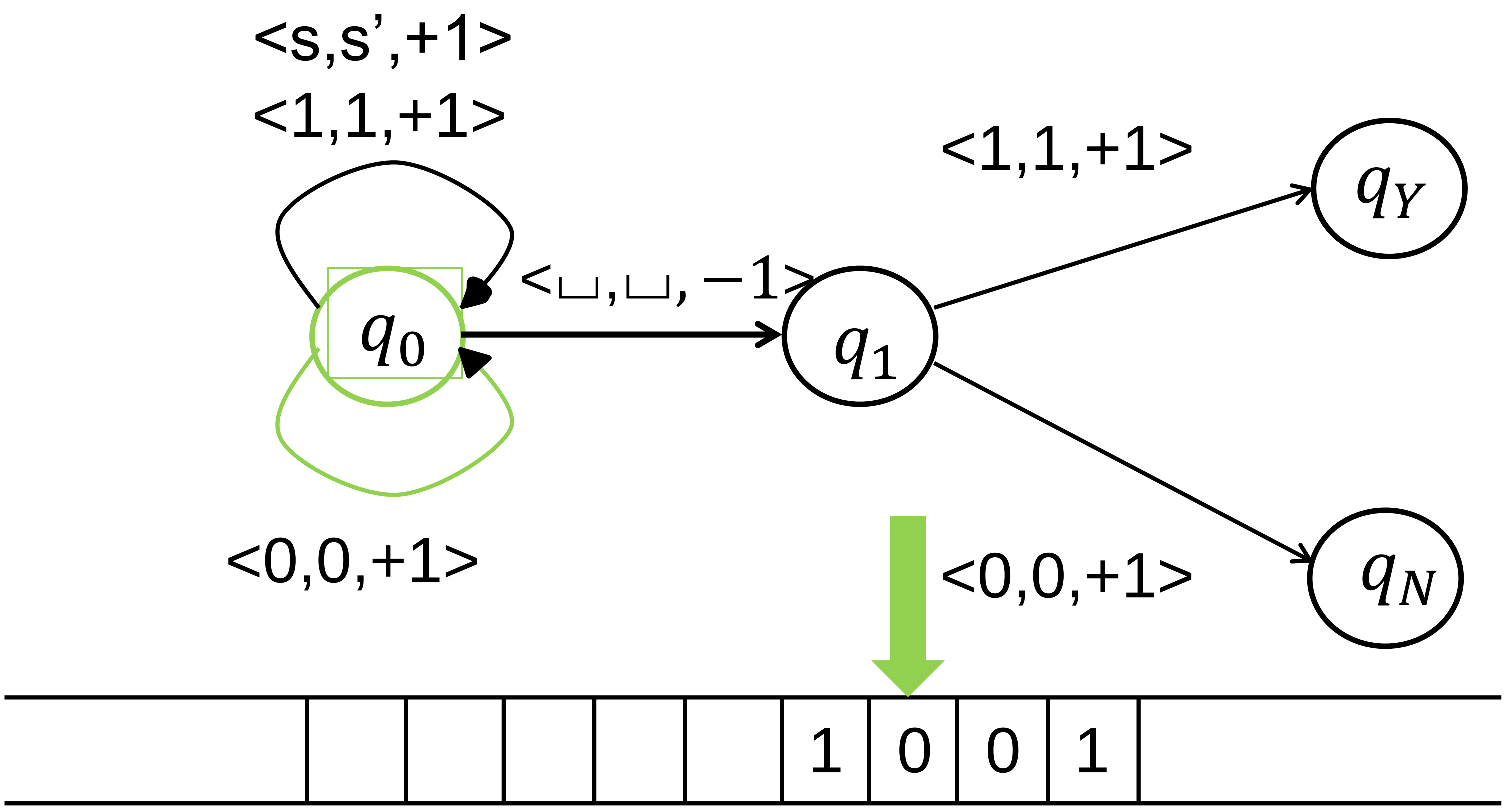
***DTM rozwiązuje problem decyzyjny $\pi$ przy kodowaniu*** $\boldsymbol{e}$, jeśli zatrzymuje się dla wszystkich słów wejściowych (reprezentujących konkretne problemy) i kończy obliczenia w stanie $q_Y$ dla wszystkich słów wejściowych $x(I)$ instancji $I$ takich, że $I \epsilon Y_\pi$ i tylko dla nich.

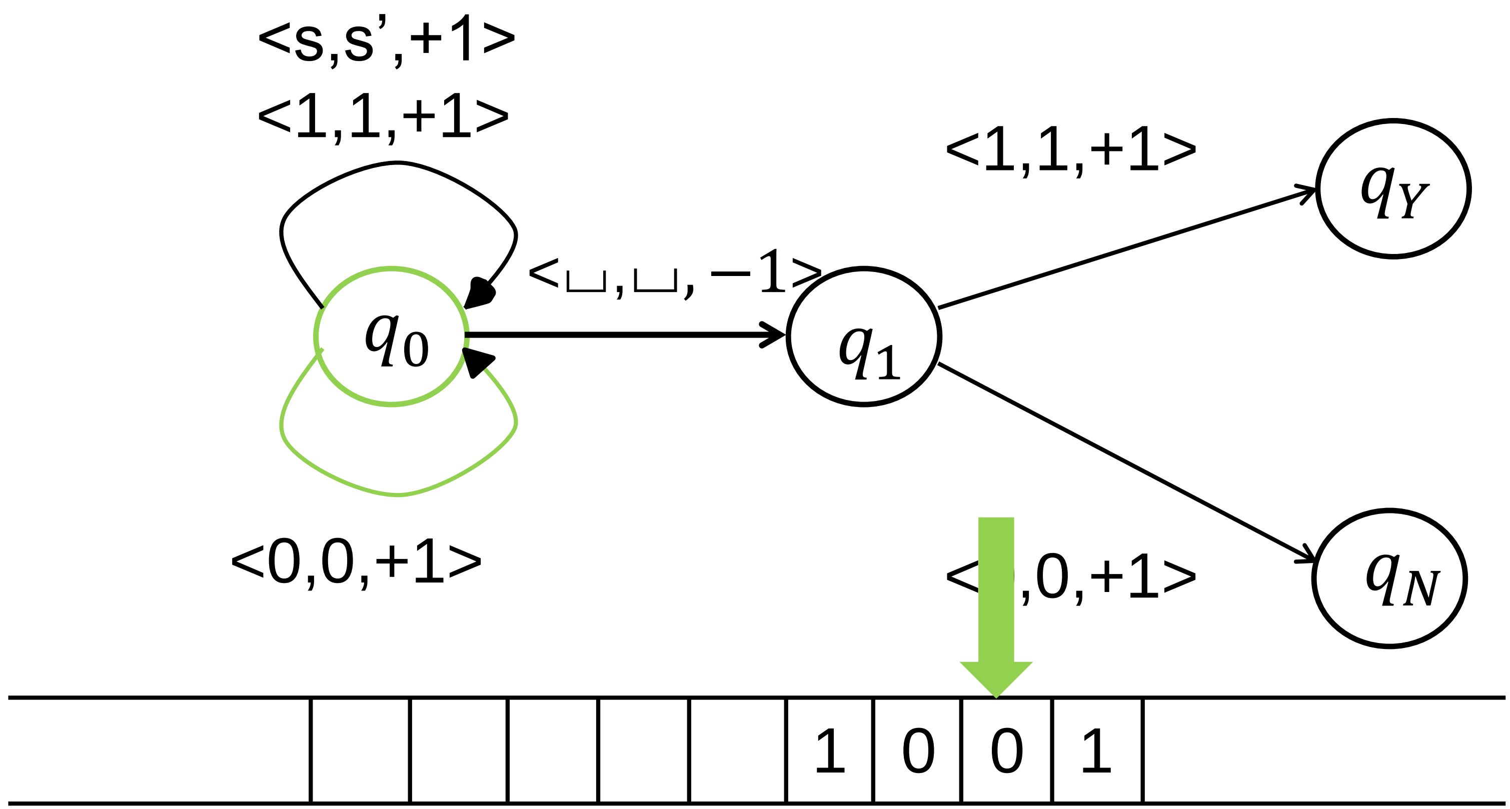
$Y_\pi$ - zbiór wszystkich instancji problemu decyzyjnego $\boldsymbol{\pi}$, dla których odpowiedzią jest „tak”.

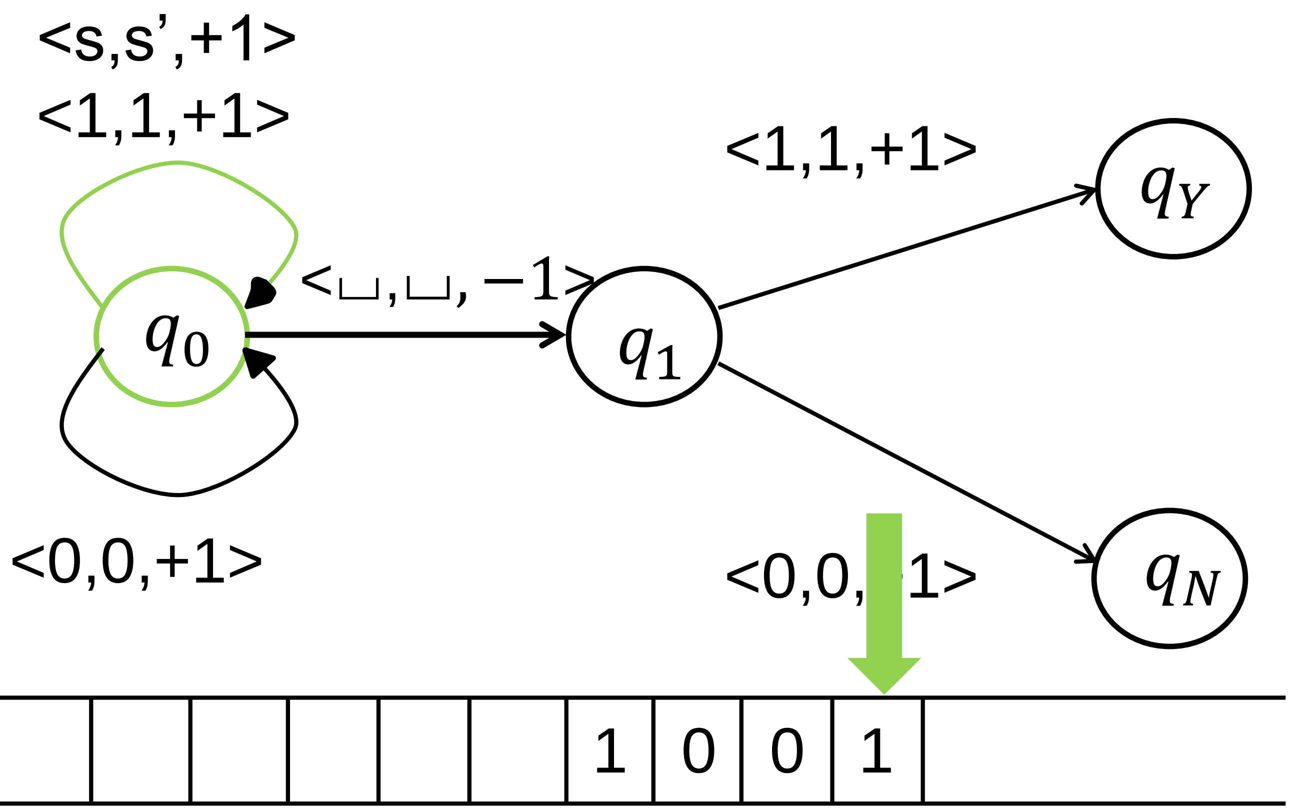

<s,s',+1>
<1,1,+1>
<␣,␣, −1>
$q_0$
$q_1$
<1,1,+1>
$q_Y$
<0,0,+1>
<0,0,+1>
$q_N$
1
0
0
1

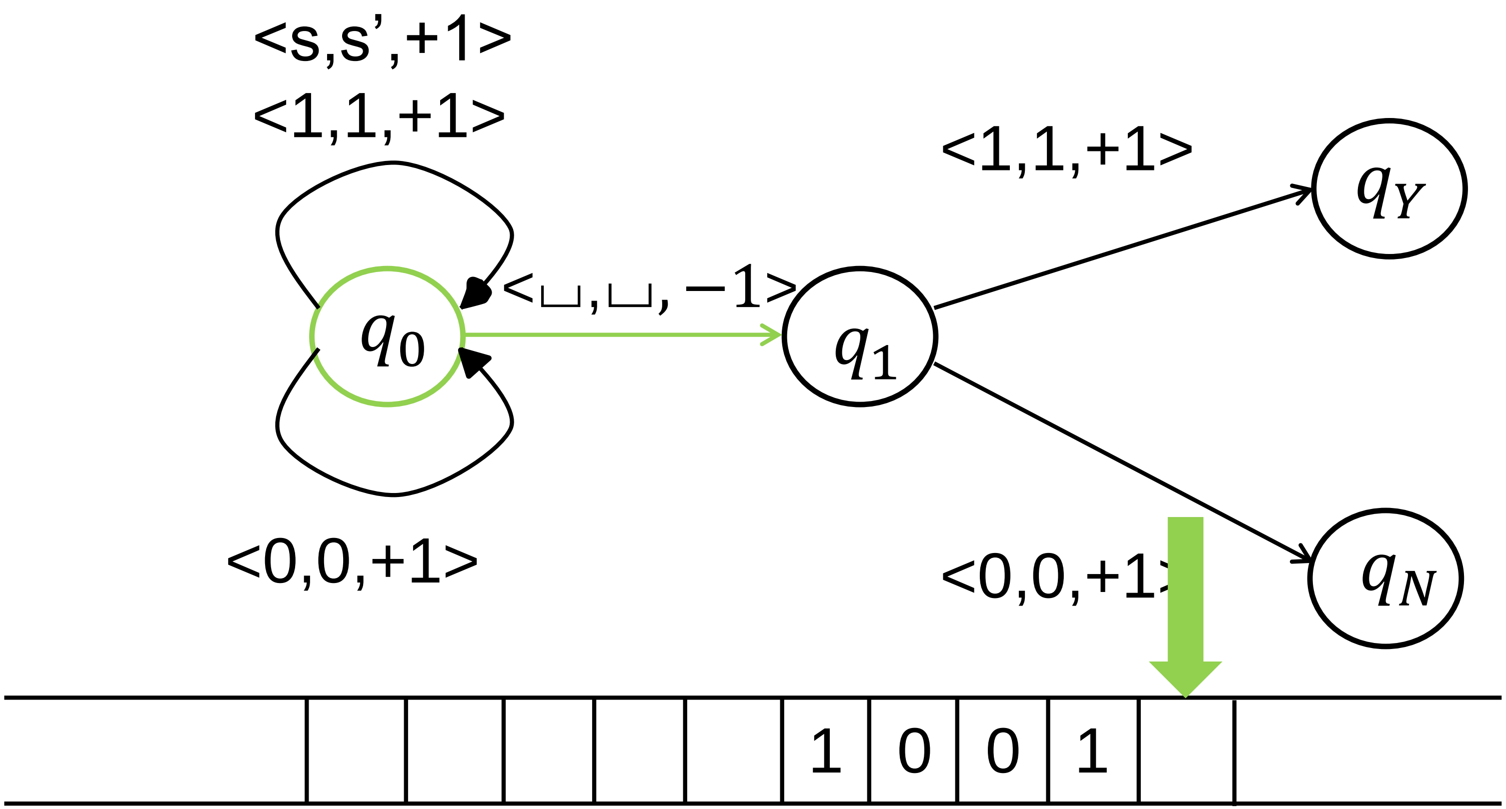

<s,s’,+1>
<1,1,+1>
<0,0,+1>
$q_0$
<⊔,⊔, −1>
$q_1$
<1,1,+1>
$q_Y$
<0,0,+1>
$q_N$
1
0
0
1

Politechnika Wrocławska

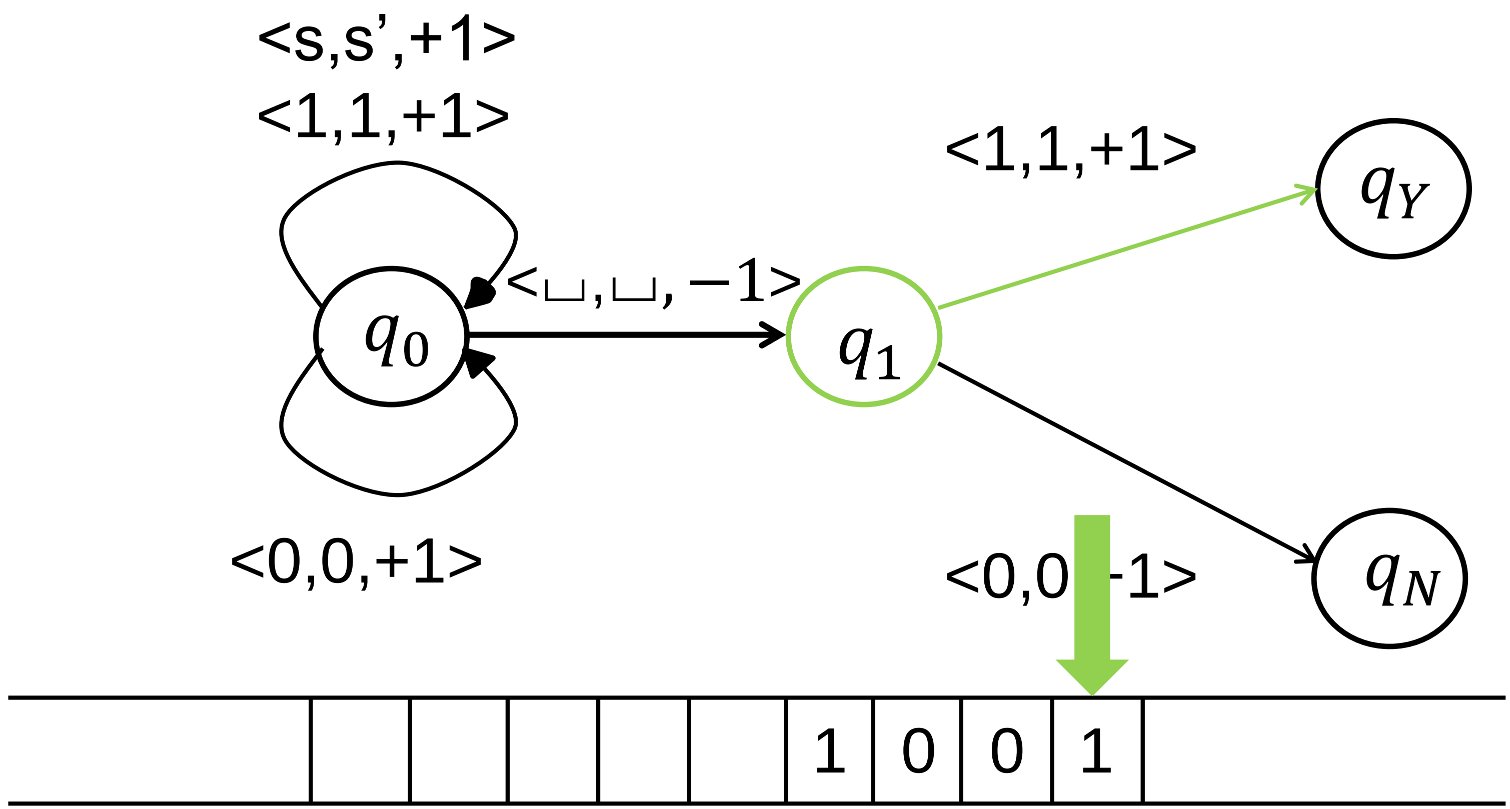


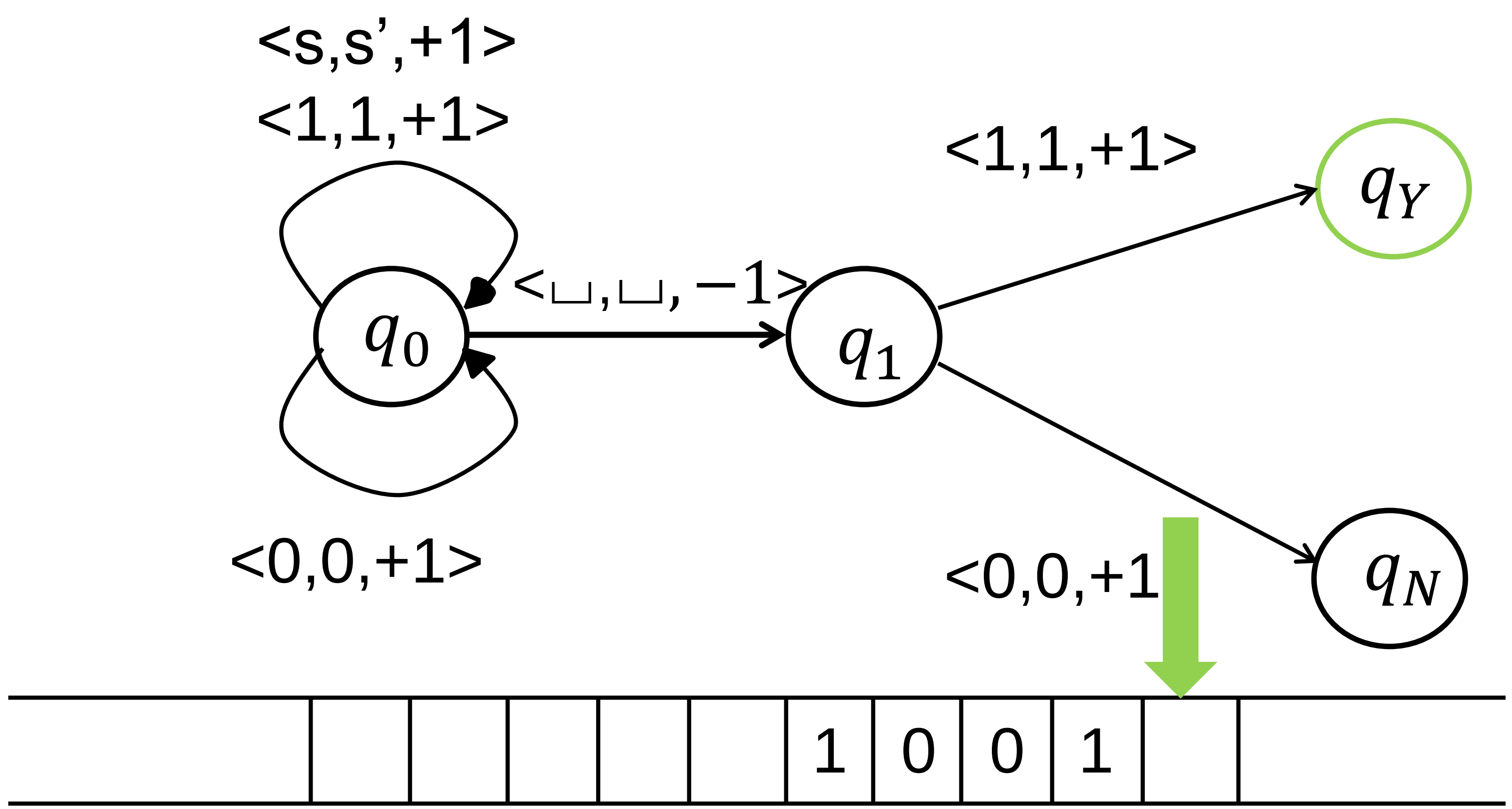

<s,s’,+1>
<1,1,+1>
<0,0,+1>
<␣,␣, −1>
$q_0$
$q_1$
<1,1,+1>
$q_Y$
,0,+1>
$q_N$
1 0 0 1


<s,s',+1>
<1,1,+1>
<⊔,⊔, −1>
$q_0$
$q_1$
$q_Y$
$q_N$
<1,1,+1>
<0,0,+1>
<0,0, 1>
1 0 0 1



<s,s’,+1>
<1,1,+1>
<⊔,⊔, −1>
$q_0$
$q_1$
<1,1,+1>
$q_Y$
<0,0,+1>
<0,0,+1>
$q_N$
1 0 0 1


<s,s',+1>
<1,1,+1>
<␣,␣, −1>
$q_0$
$q_1$
<1,1,+1>
$q_Y$
$q_N$
<0,0,+1>
<0,0,-1>
1 0 0 1

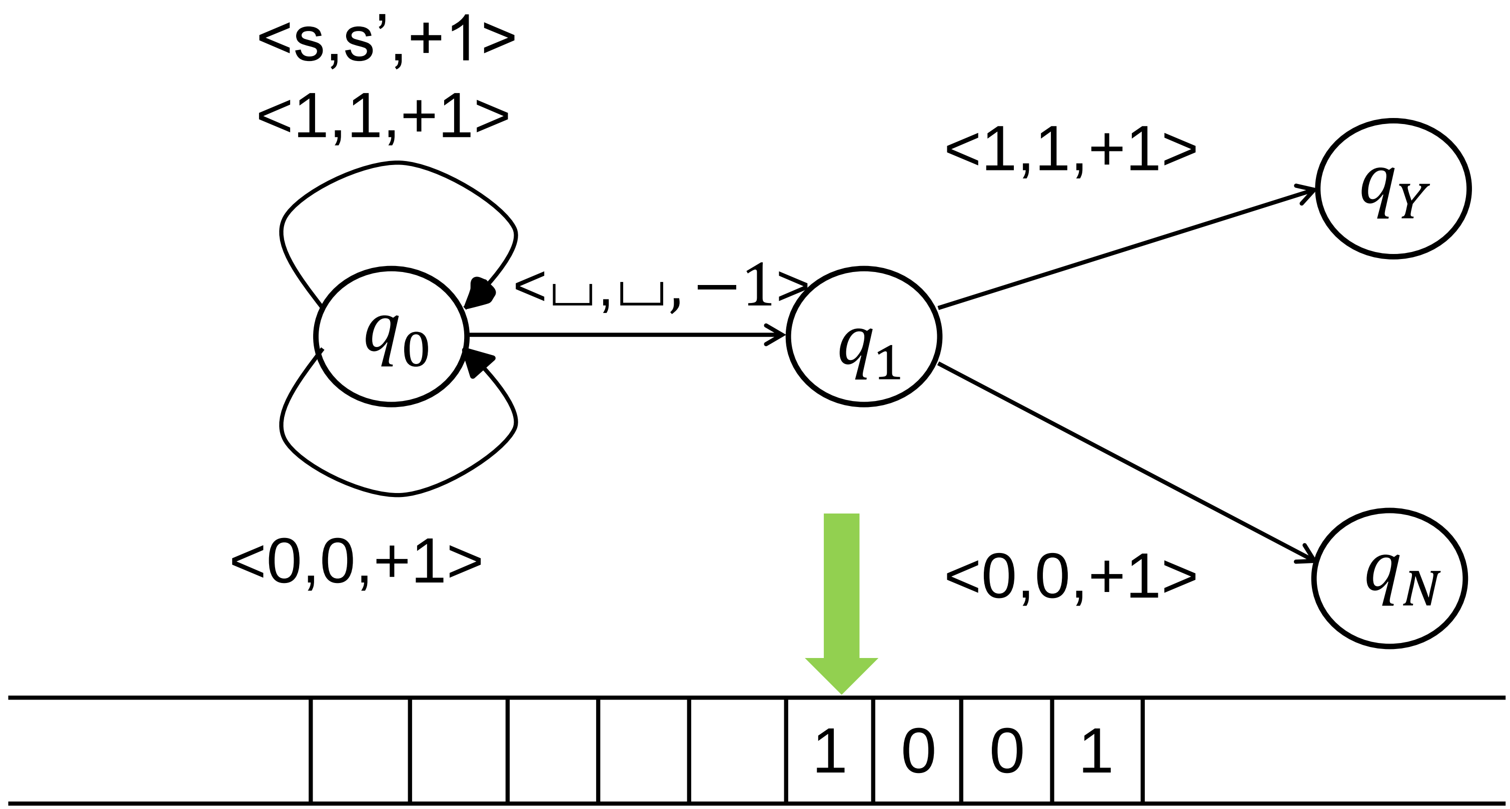


<s,s',+1>
<1,1,+1>
<0,0,+1>
<⊔,⊔, −1>
<1,1,+1>
<0,0,+1
$q_0$
$q_1$
$q_Y$
$q_N$
1 0 0 1

Ruch w prawo bez zmiany zawartości komórki

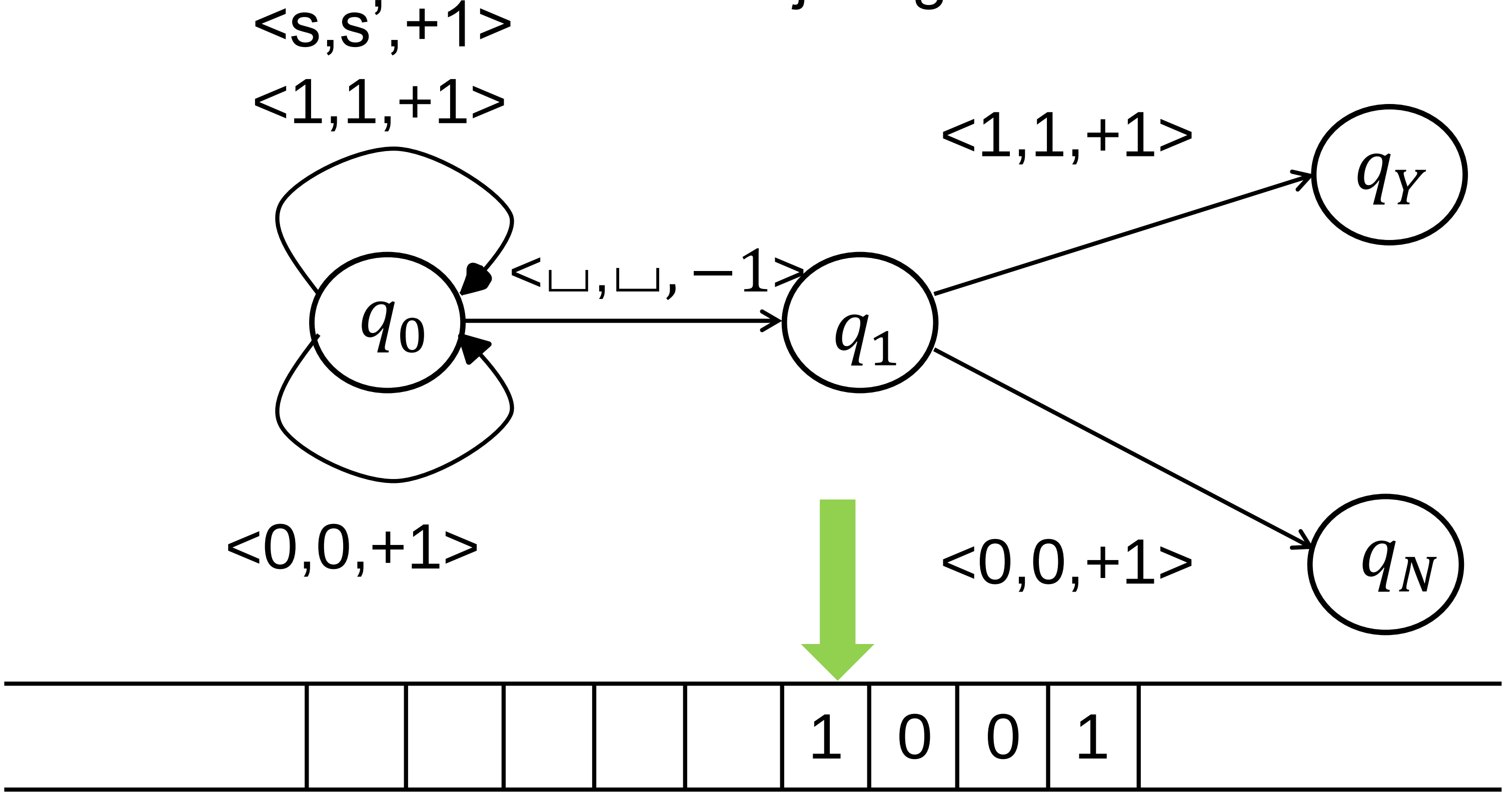
Wykrycie końca słowa

Rozróżnienie ostatniej litery: 0 czy 1

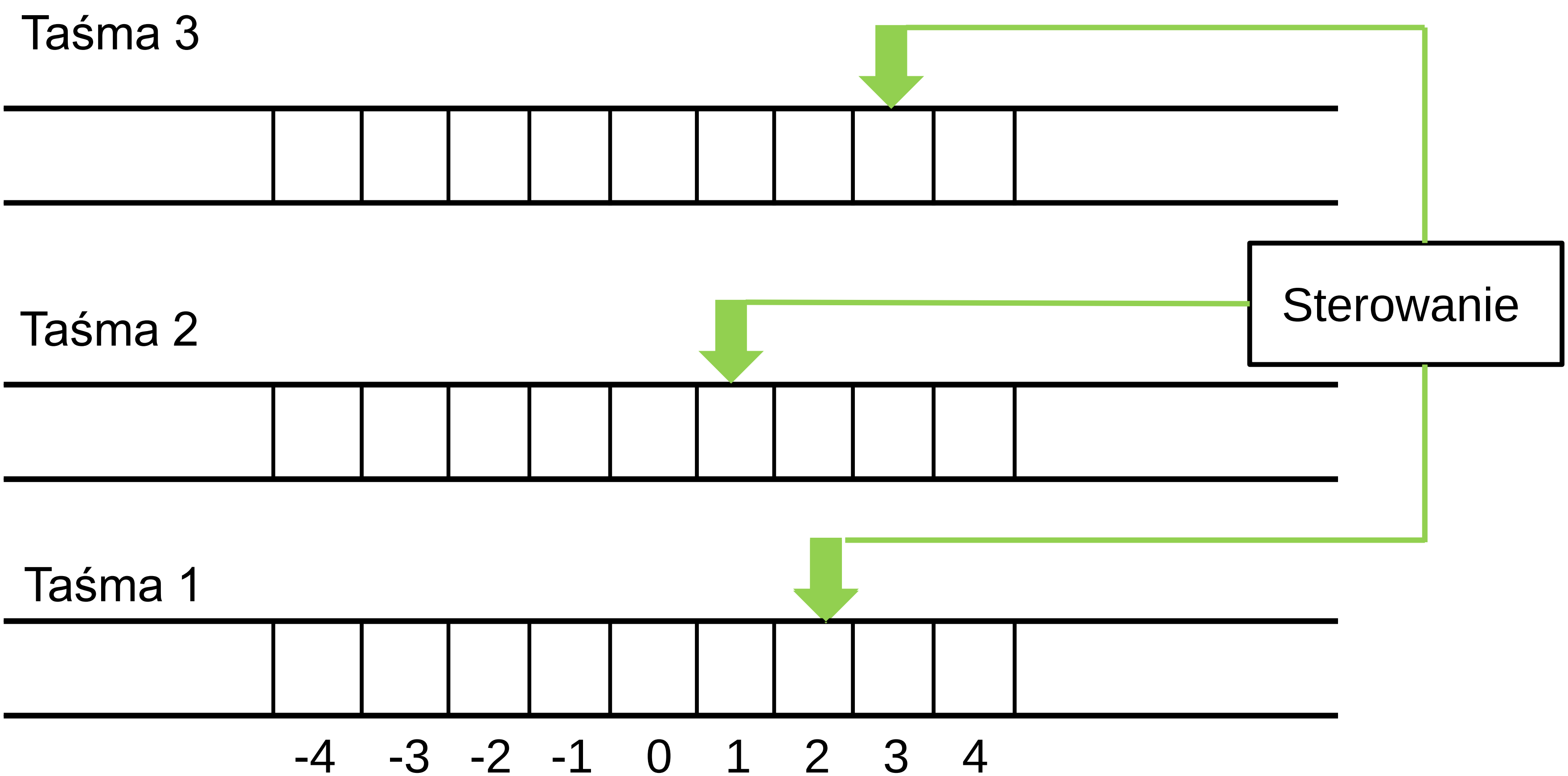


DMT rozpoznająca liczby nieparzyste

Jakiej długości?



# *Deterministyczna trójtaśmowa DMT*



**Cel:**

Dodawanie dwu liczb dwójkowych tej samej długości, znajdujących się na Taśmach 1 i 2, których najstarsze pozycje znajdują się w komórkach o numerze 1.

Wynik tworzony jest na Taśmie 3. Cyfra najmłodszej pozycji powinna być umieszczona tak jak cyfry najmłodszych pozycji danych wejściowych. Głowica tej taśmy powinna wskazywać najstarszą cyfrę wyniku.

**Plan:**

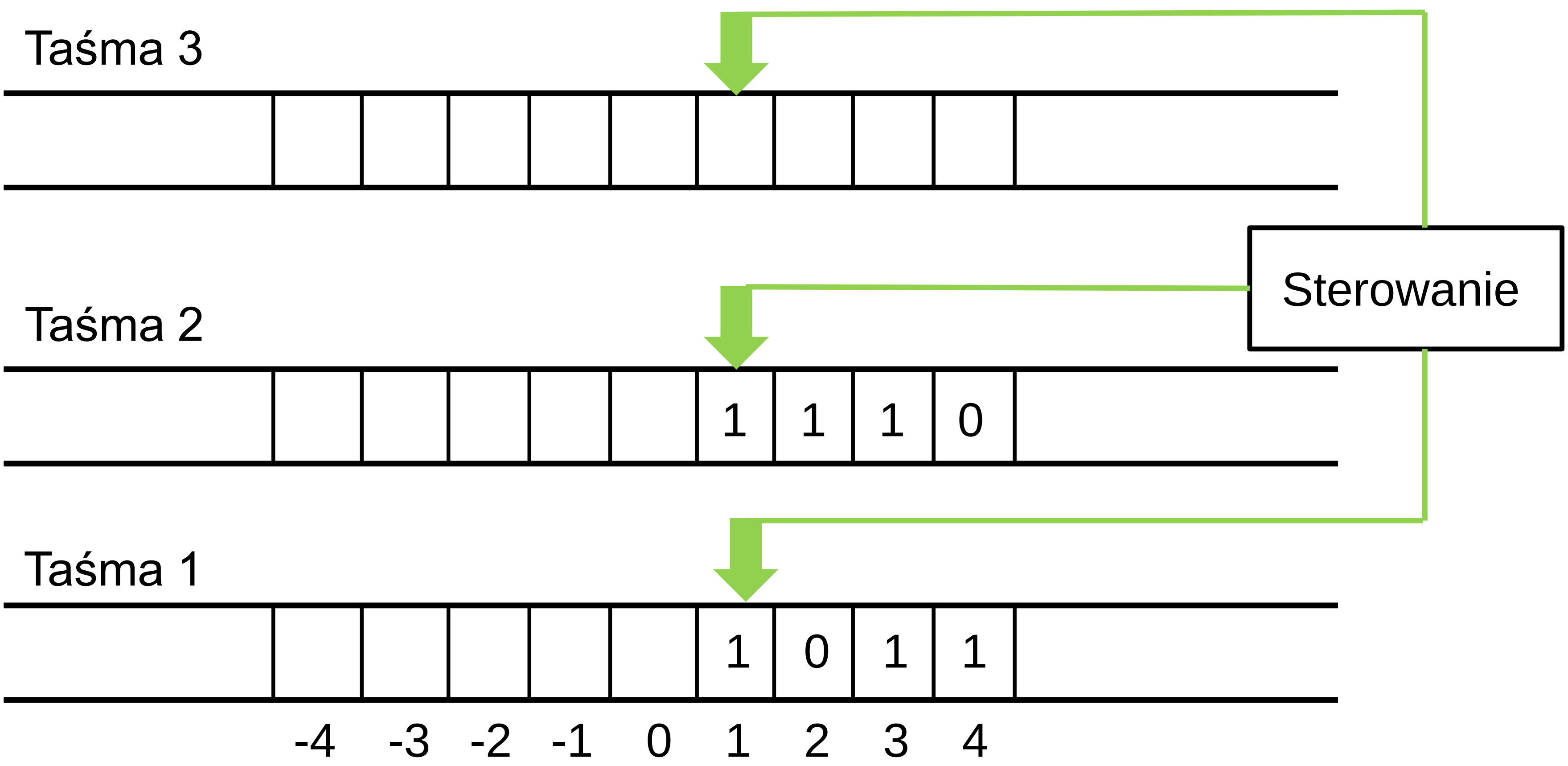
Wymagania:

Zewnętrzne objawy zachowania trójtaśmowej DMT.

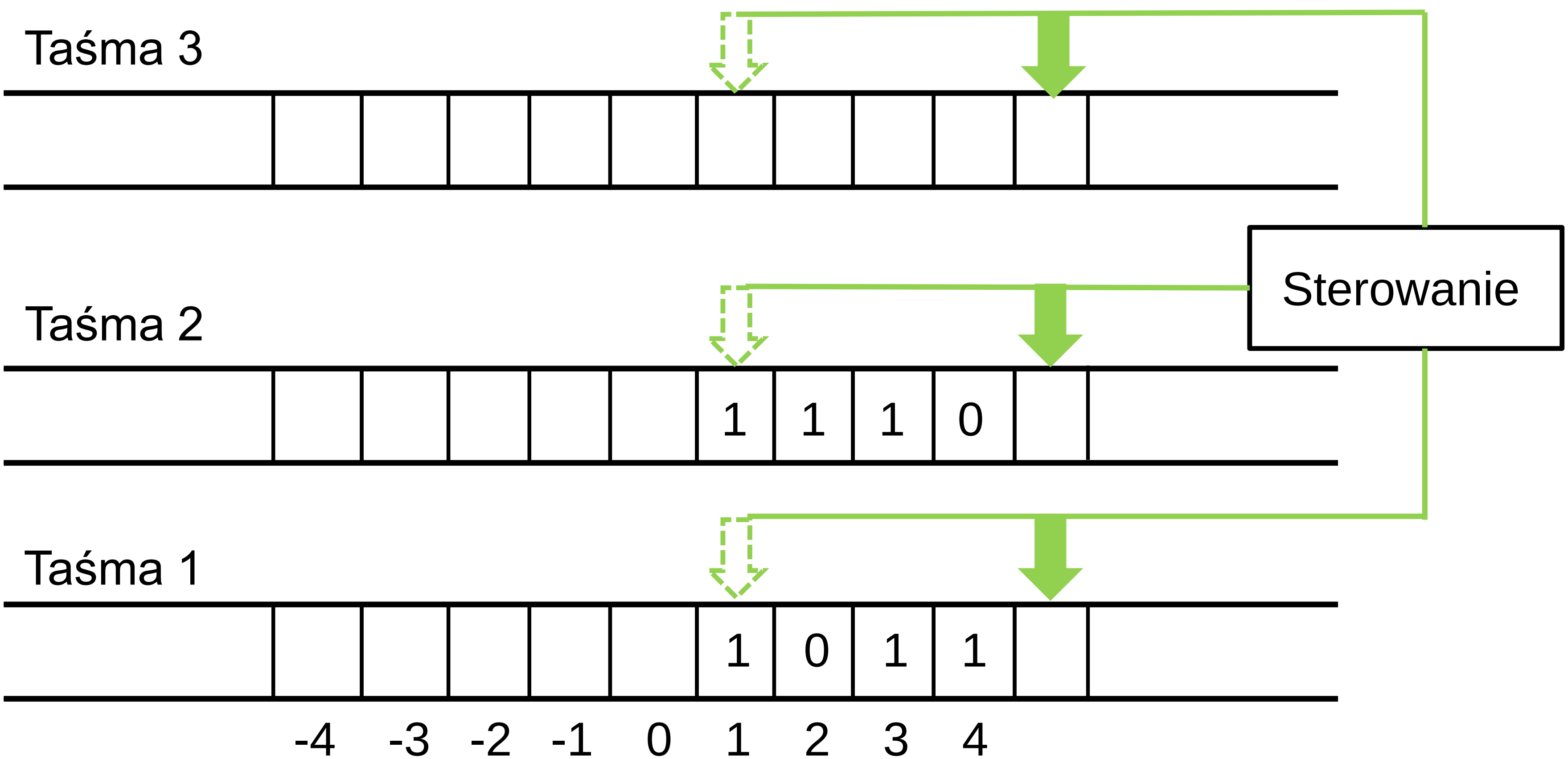
Specyfikacja:

Diagram przejść maszyny.

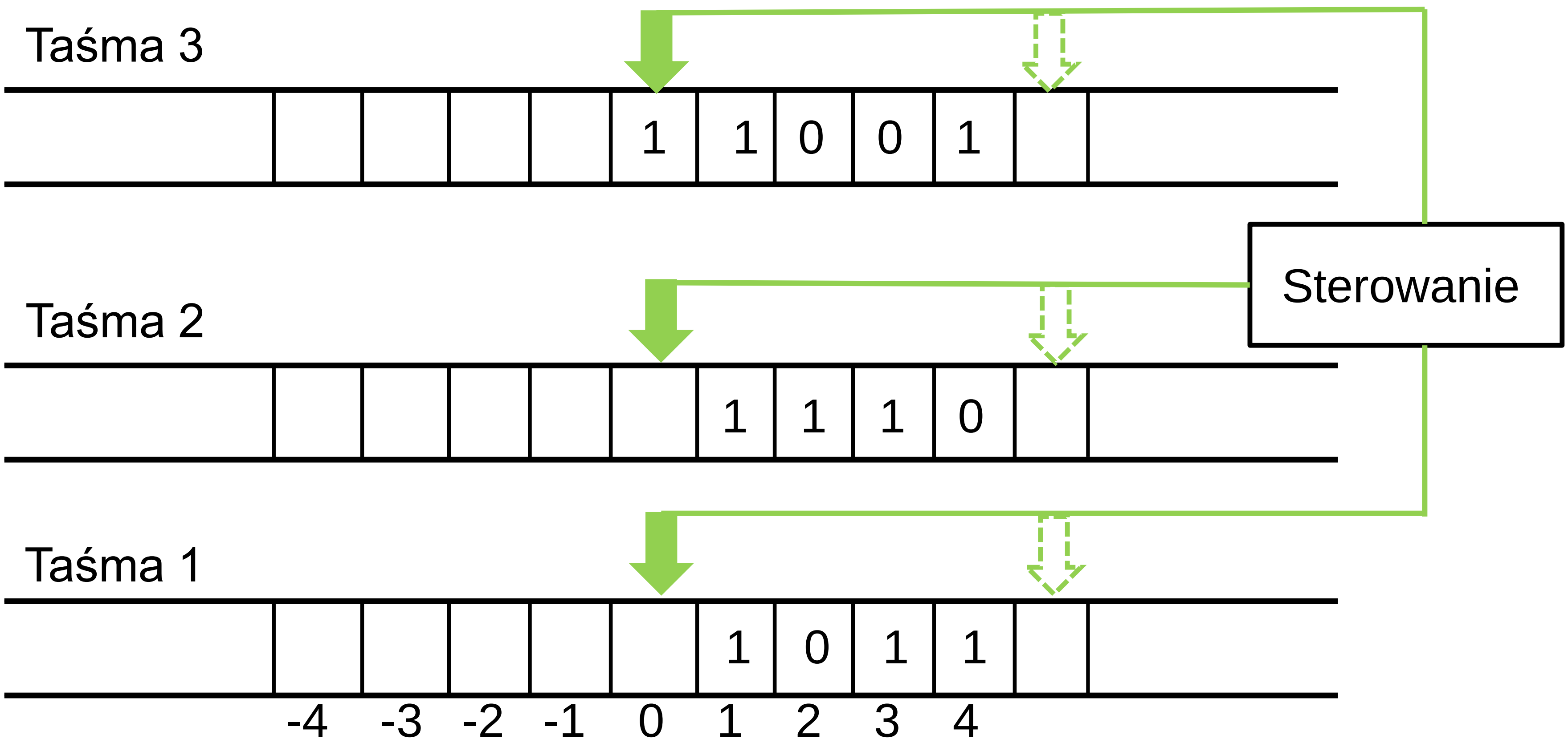
**Start:** Maszyna znajduje się w stanie początkowym $q_0$, natomiast głowice odczytują symbole z trzech taśm z komórek o numerze 1.



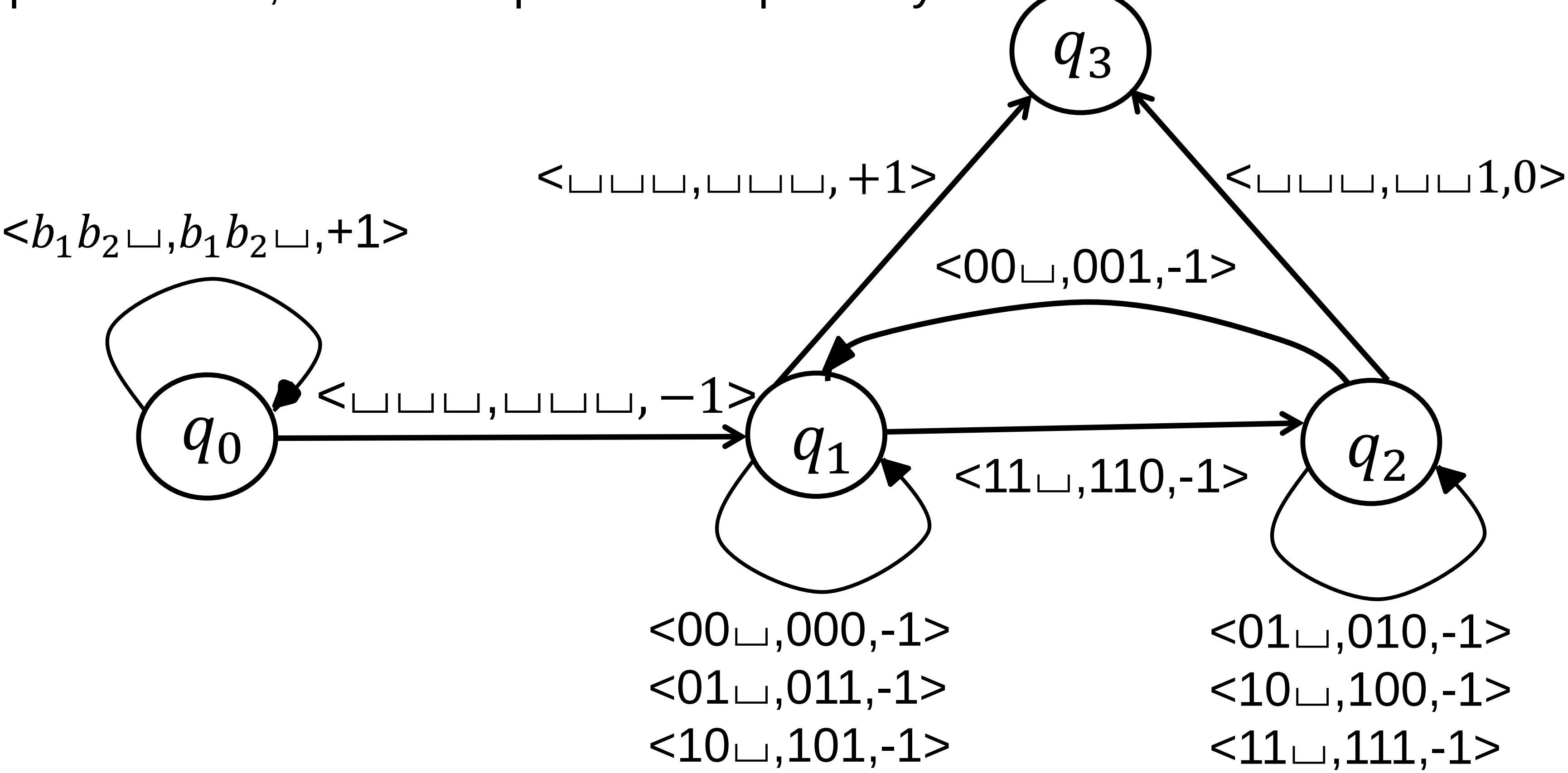
Głowice trzech taśm przesuwają się w prawo do osiągnięcia komórek ze znakami pustymi znajdujących się na prawo od liczb.



Głowice trzech taśm przesuwają się w lewo do osiągnięcia komórek ze znakami pustymi znajdujących się na lewo od liczb na Taśmach 1 i 2. Kolejne cyfry sumy zapisywane są na Taśmie 3.



$b_i$ - cyfra dwójkowa $i$–tej taśmy.
W tym przykładzie, trzy głowice poruszają się synchronicznie. Stąd, dla uproszczenia, zamiast np. +1+1+1 piszemy +1.

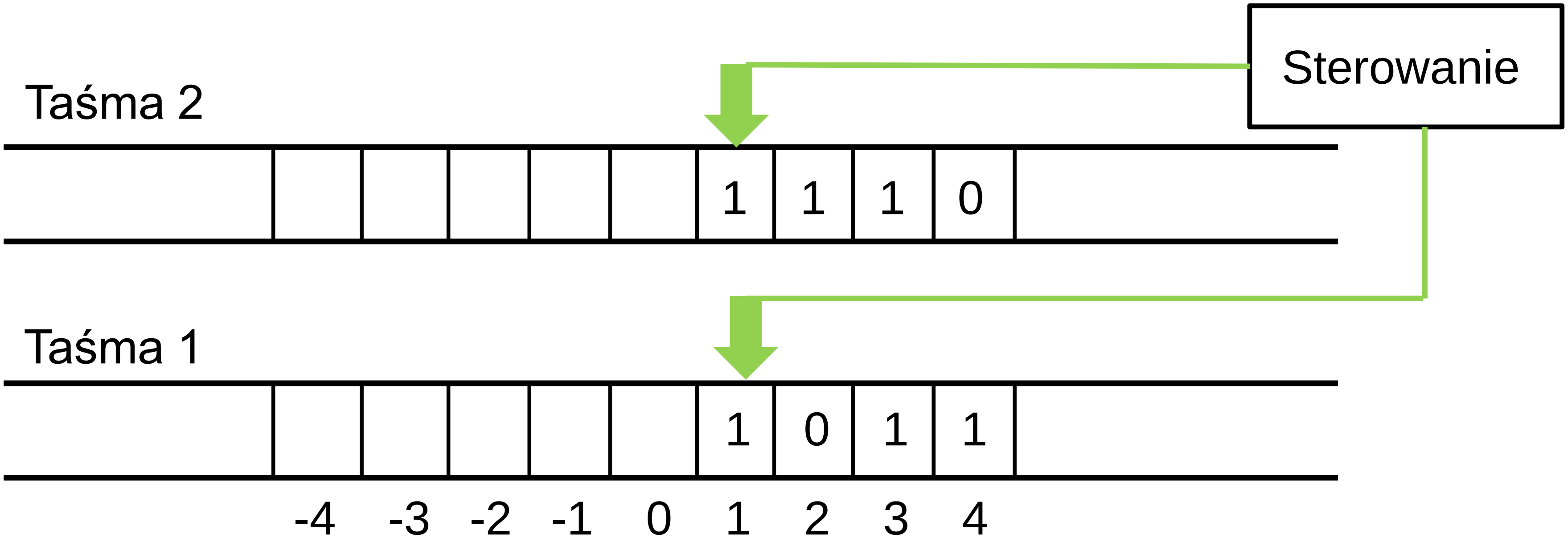


**Cel:**

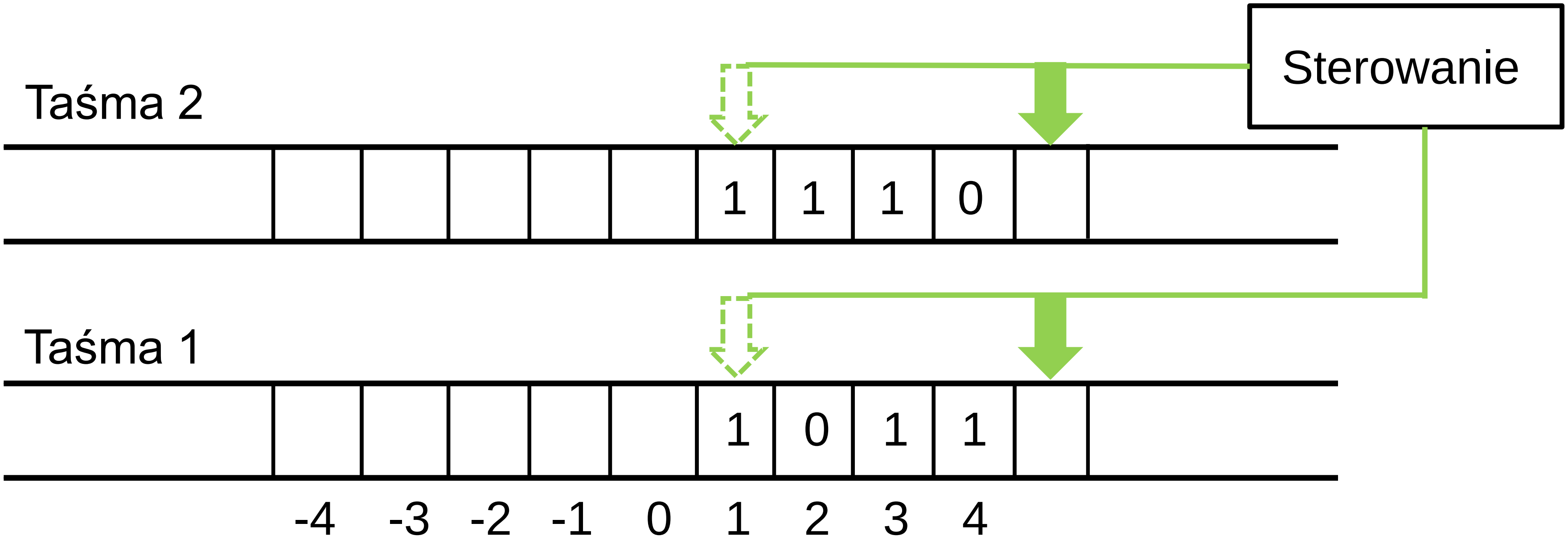
Dodawanie dwu liczb dwójkowych tej samej długości, znajdujących się na Taśmach 1 i 2, których najstarsze pozycje znajdują się w komórkach o numerze 1.

Wynik tworzony jest na Taśmie 2. Głowica tej taśmy powinna wskazywać najstarszą cyfrę wyniku.

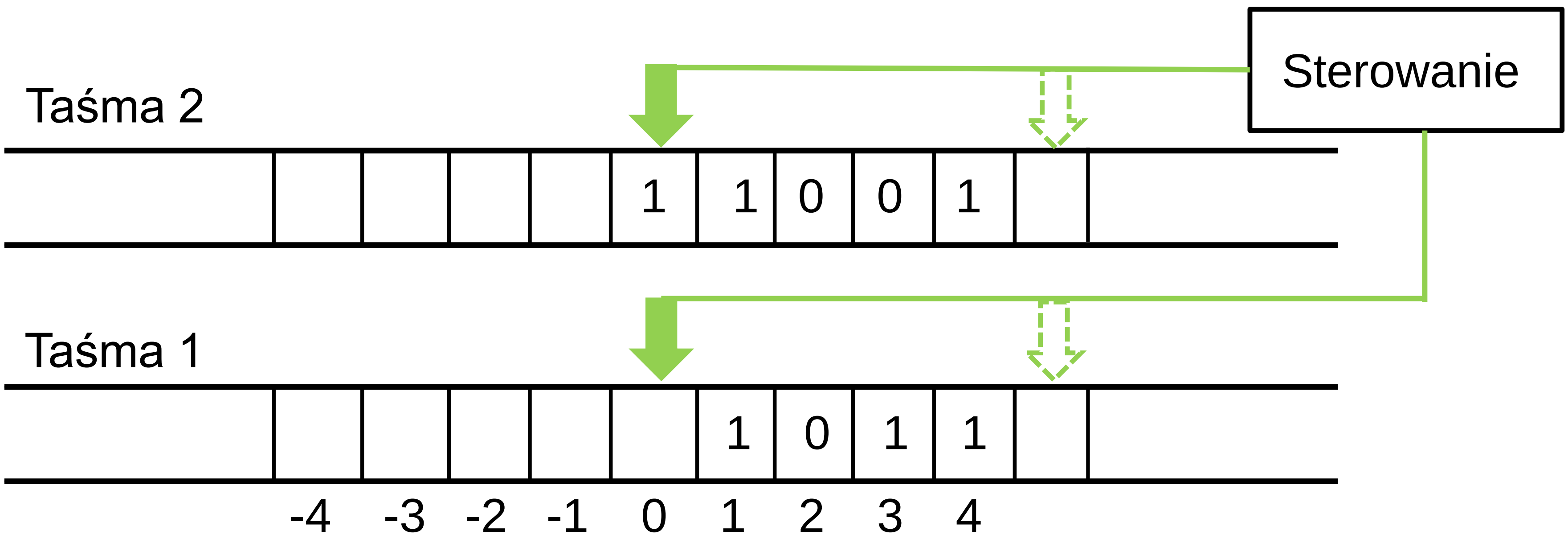
**Start:** Maszyna znajduje się w stanie początkowym $q_0$, natomiast głowice odczytują symbole z dwu taśm z komórek o numerze 1.



Głowice dwu taśm przesuwają się w prawo do osiągnięcia komórek ze znakami pustymi znajdujących się na prawo od liczb.

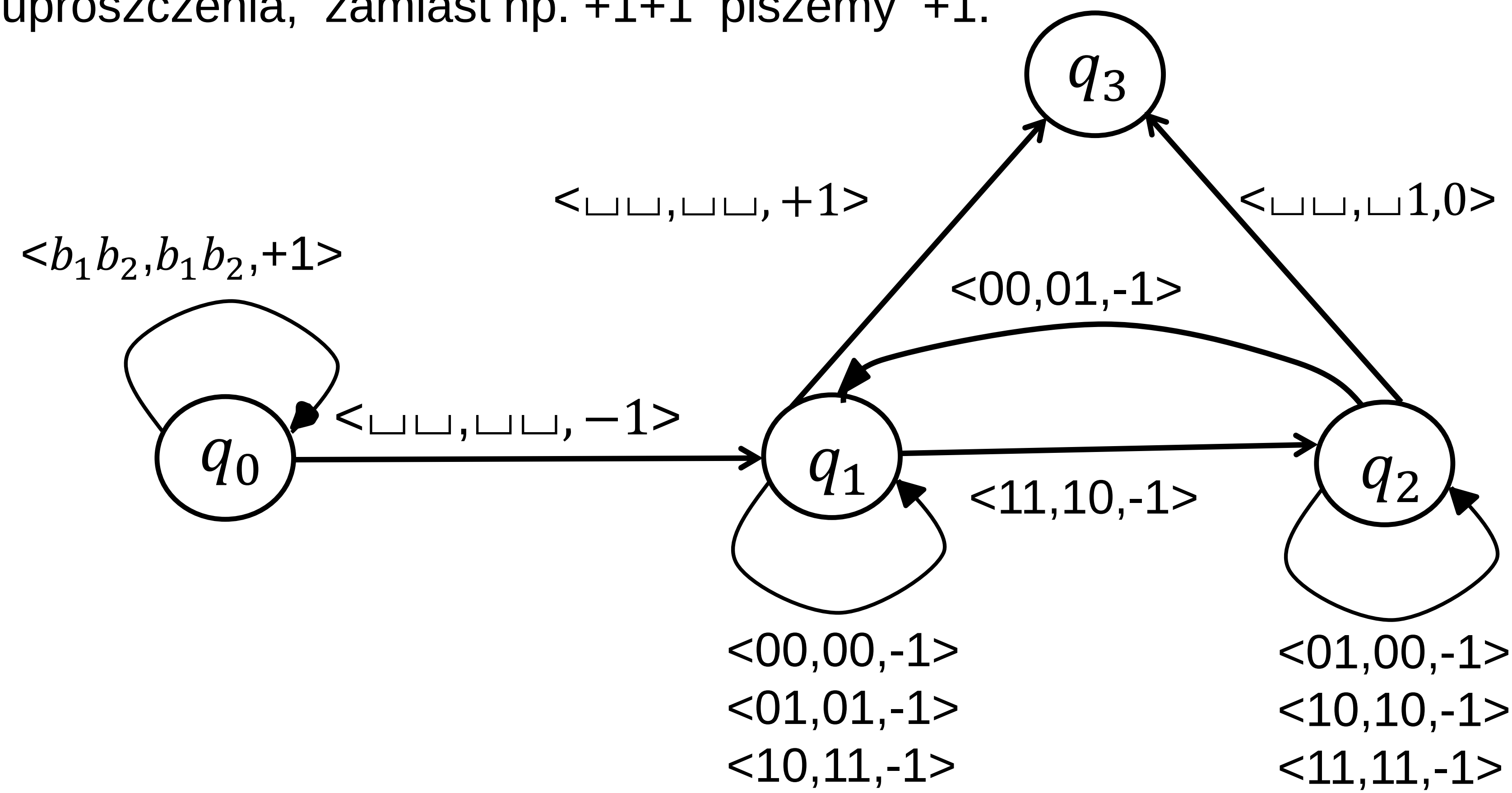


Głowice dwu taśm przesuwają się w lewo do osiągnięcia komórek ze znakami pustymi znajdujących się na lewo od liczb na Taśmach 1 i 2. Kolejne cyfry sumy zapisywane są na Taśmie 2.



# <1,1,+1>

$b_i$ - cyfra dwójkowa $i$–tej maszyny.
W tym przykładzie, dwie głowice poruszają się synchronicznie.  Stąd, dla uproszczenia,  zamiast np. +1+1  piszemy  +1.



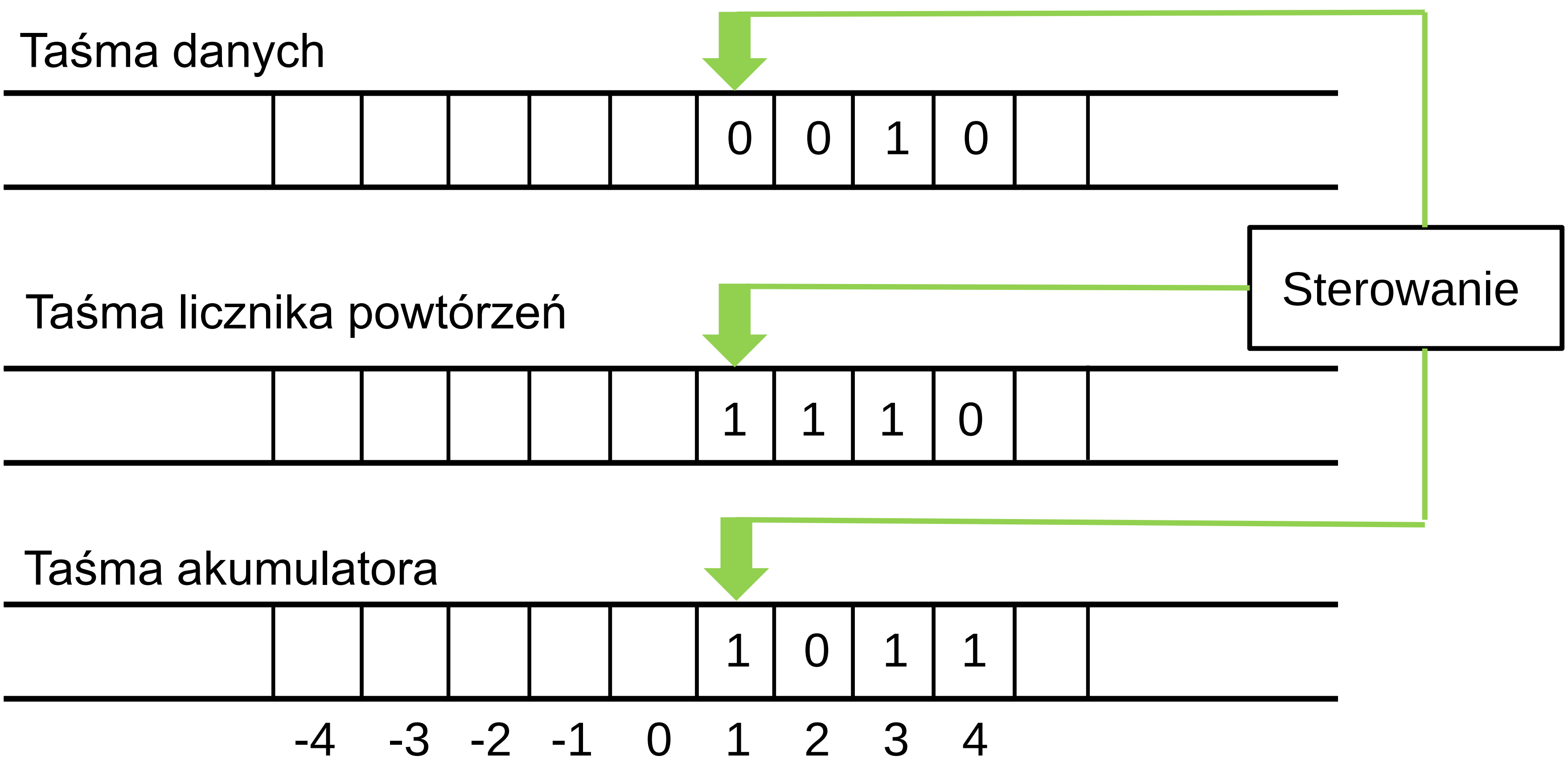
**Cel:**

Dodawanie dwu liczb dwójkowych tej samej długości, znajdujących się na Taśmie 1. Pierwsza jest zapisana od komórki o numerze 1, natomiast druga po spacji oddzielającej obie liczby.

Wynik tworzony jest na Taśmie 1 przed spacją poprzedzającą pierwszą liczbę. Głowica tej taśmy powinna wskazywać najstarszą cyfrę wyniku.

# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

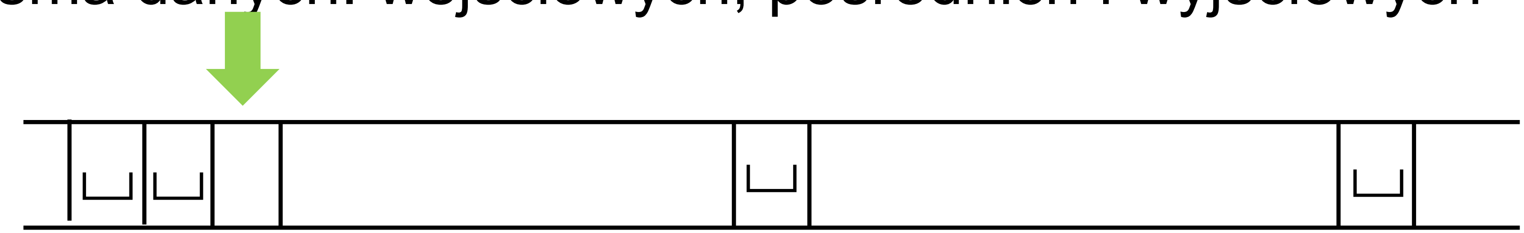
## Założenia:



# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

## Założenia:

Taśma danych: wejściowych, pośrednich i wyjściowych



1. Dana
12 kom.

2. Dana
12 kom.

W stanie początkowym:

Głowica nad pierwszą komórką pierwszej danej.

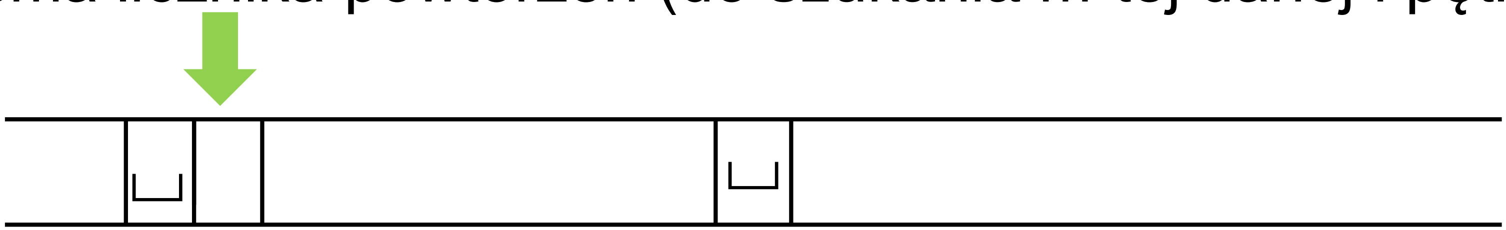
Po wykonaniu instrukcji:

Głowica nad pierwszą komórką pierwszej danej.

# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

## Założenia:

Taśma licznika powtórzeń (do szukania m-tej danej i pętli



Dana
12 kom.

W stanie początkowym:

Głowica nad pierwszą komórką zawartości licznika powt.

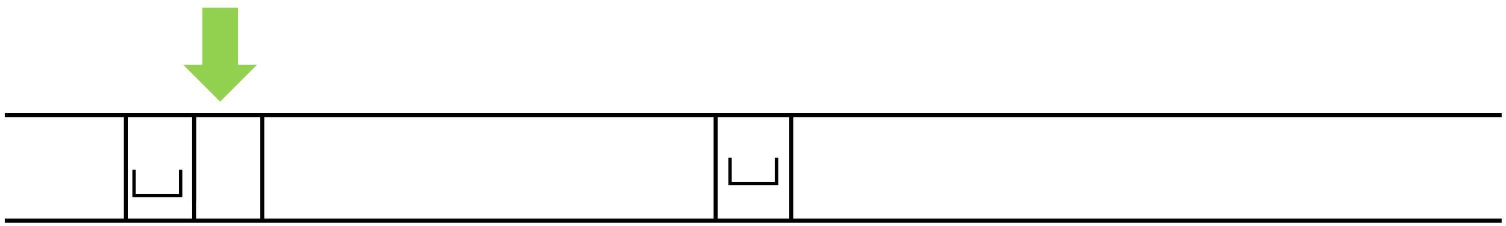
Po wykonaniu instrukcji:

Głowica nad pierwszą komórką zawartości licznika powt.

# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

**Założenia:**

Taśma akumulatora




Dana
12 kom.

W stanie początkowym:

Głowica nad pierwszą komórką zawartości akumulatora.

Po wykonaniu instrukcji:

Głowica nad pierwszą komórką zawartości akumulatora.

# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

Etykiety łuków grafu maszyny Turinga opisane następująco:

$$< x_d x_l x_a, x_d{}' x_l{}' x_a, r_d r_l r_a >$$

$x_d x_l x_a$ - odczytywane symbole na taśmach: danych, licznika powtórzeń i akumulatora, $x_d, x_l, x_a \in \{\sqcup, 0,1\}$,

$x_d{}' x_l{}' x_a{}'$ - zapisywane symbole na taśmach: danych, licznika powtórzeń i akumulatora, $x_d{}', x_l{}', x_a{}' \in \{\sqcup, 0,1\}$,

$r_d r_l r_a$ - ruchy taśm: danych, licznika powtórzeń i akumulatora,

# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

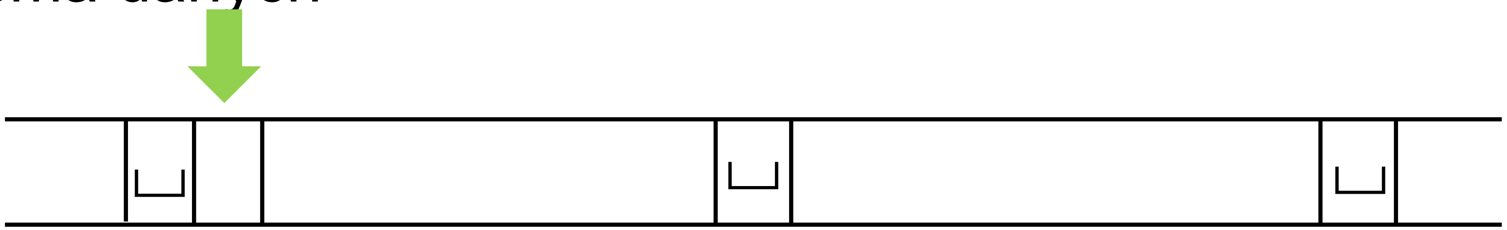
Wykonanie instrukcji dodawania wartości $i$-tej danej do akumulatora:

1. Szukanie $i$-tej danej na taśmie danych i ustawianie głowicy nad jej najmłodszą pozycją,
2. Ustawianie głowicy taśmy akumulatora nad najmłodszą pozycją,
3. Dodawanie wartości $i$-tej danej do wartości akumulatora i zapamiętanie w tymże.
4. Ustawienie taśmy danych nad pierwszą komórką pierwszej danej.

# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

## Założenia:

Taśma danych




1. Dana
12 kom.

2. Dana
12 kom.

W stanie początkowym:

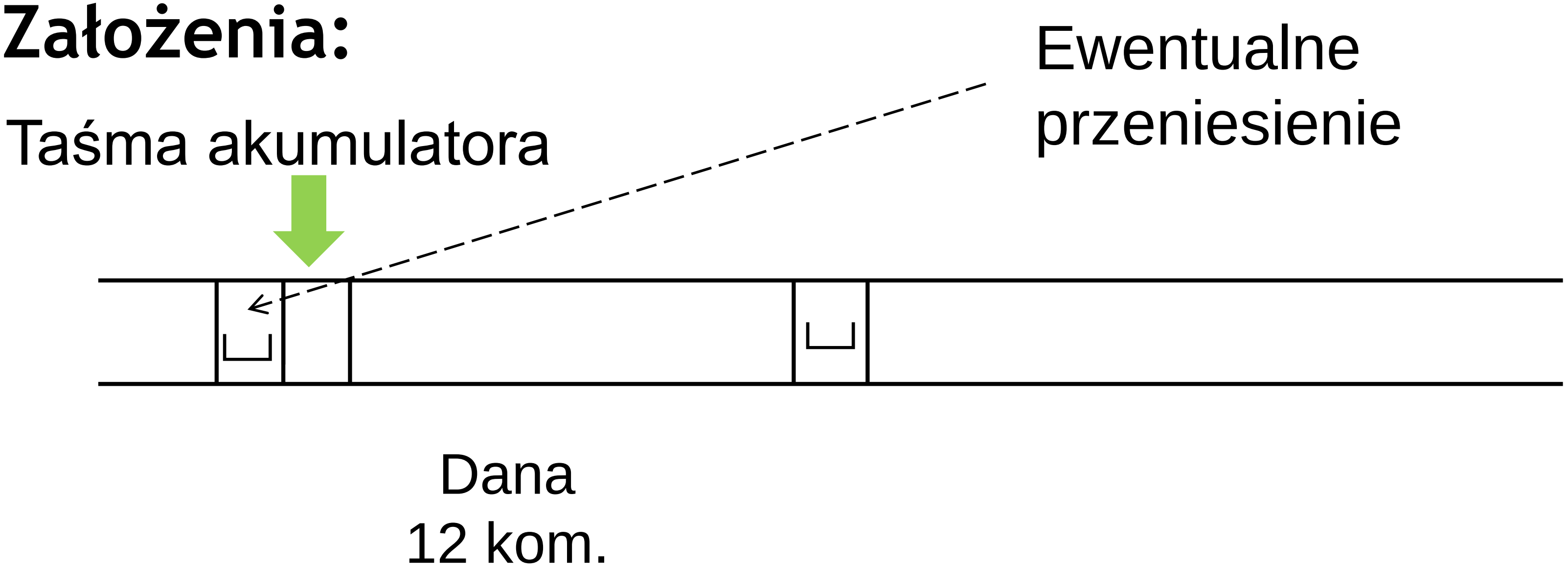
Głowica nad pierwszą komórką pierwszej danej.

Po wykonaniu instrukcji:

Głowica nad pierwszą komórką pierwszej danej.

# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

## Założenia:




W stanie początkowym:

Głowica nad pierwszą komórką zawartości akumulatora.

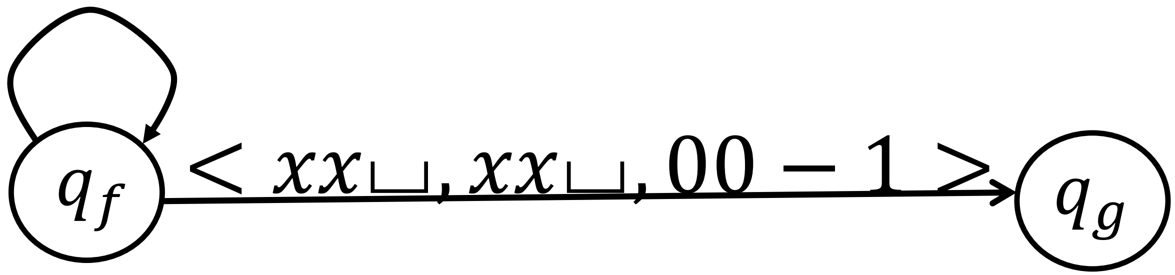
Po wykonaniu instrukcji:

Głowica nad pierwszą komórką zawartości akumulatora.

# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

- $b \in \{0,1\}$

$< xxb, xxb, 00 + 1 >$

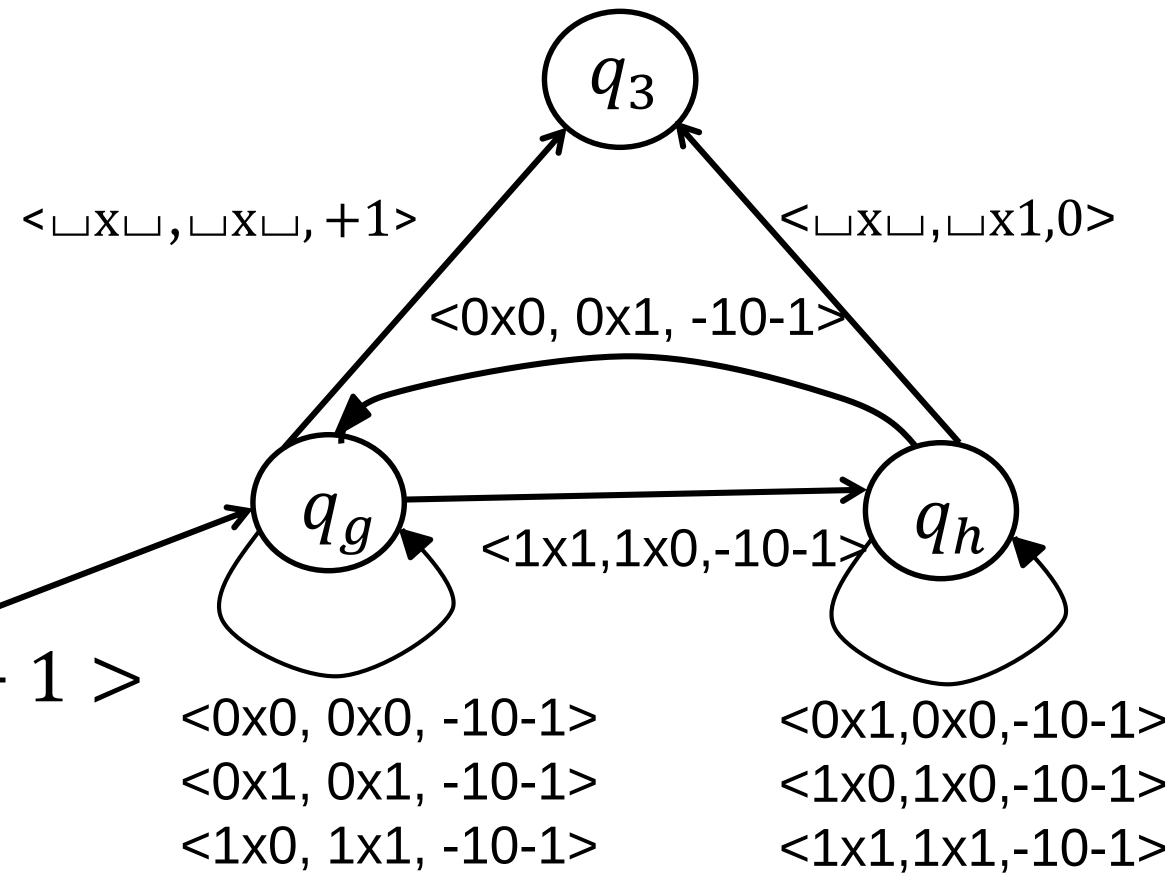



2. Ustawianie głowicy taśmy akumulatora nad najmłodszą pozycją

# Maszyna Turinga z taśmami: danych, licznika powtórzeń i akumulatora

3. Dodawanie danej do wartości akumulatora i zapamiętanie w tymże.

$$< xx\sqcup, xx\sqcup, 00-1 >$$



Model ***RAM*** (ang. random access machine) zawiera:

- Taśmę wejściową z głowicą odczytującą,
- Taśmę wyjściową z głowicą zapisującą,
- Pamięć danych jest zbiorem rejestrów z wyróżnionym rejestrem – akumulatorem, w którym wykonywane są obliczenia,
- Program, który nie jest przechowywany w pamięci jest sekwencją rozkazów (nie podlega automodyfikacji),
- Licznik rozkazów.

**Kryteria kosztów operacji elementarnych**

(zapisania, dodawania, odejmowania, porównania dwu liczb, itp.)

***Logarytmiczne kryterium kosztów***

Czas wykonania elementarnej operacji zależy liniowo od długości łańcucha danych kodujących liczby, a zatem od logarytmów liczb.

Analiza teoretyczna z użyciem DMT prowadzona jest przy tym kryterium.

***Jednorodne kryterium kosztów***

Czas wykonania elementarnej operacji jest jednostkowy.

Analiza praktyczna często oparta jest na tym kryterium.

## Twierdzenie

***Modele procesu obliczeń:***

- ***Jednotaśmowa maszyna Turinga,***
- ***Wielotaśmowa maszyna Turinga,***
- ***Maszyna RAM***

***są równoważne*** w tym sensie, że jeśli dany problem jest rozwiązywany przez jeden model w czasie ograniczonym od góry przez wielomian zależny od rozmiarów problemu, to przy założeniu logarytmicznego kryterium kosztów jest on również rozwiązywany przez każdy inny model w czasie ograniczonym od góry przez wielomian zależny od jego rozmiarów.

Interpretacja na gruncie teorii języków formalnych faktu:

Program (algorytm) $P$ rozwiązuje problem decyzyjny $\boldsymbol{\pi}$

będzie tematem sześciu najbliższych slajdów.

# Program (algorytm) $P$ rozwiązuje problem decyzyjny $\pi$

***Językiem*** $L$ ***alfabetu*** $\Sigma$ jest skończony ciąg słów tego alfabetu oddzielonych separatorami ␣ .

Przykłady

Język dziesiętny: 402␣31␣58

Język dwójkowy: 1011 ␣101111 ␣11 ␣1000

Język jedynkowy: 11␣1111␣1␣111␣111

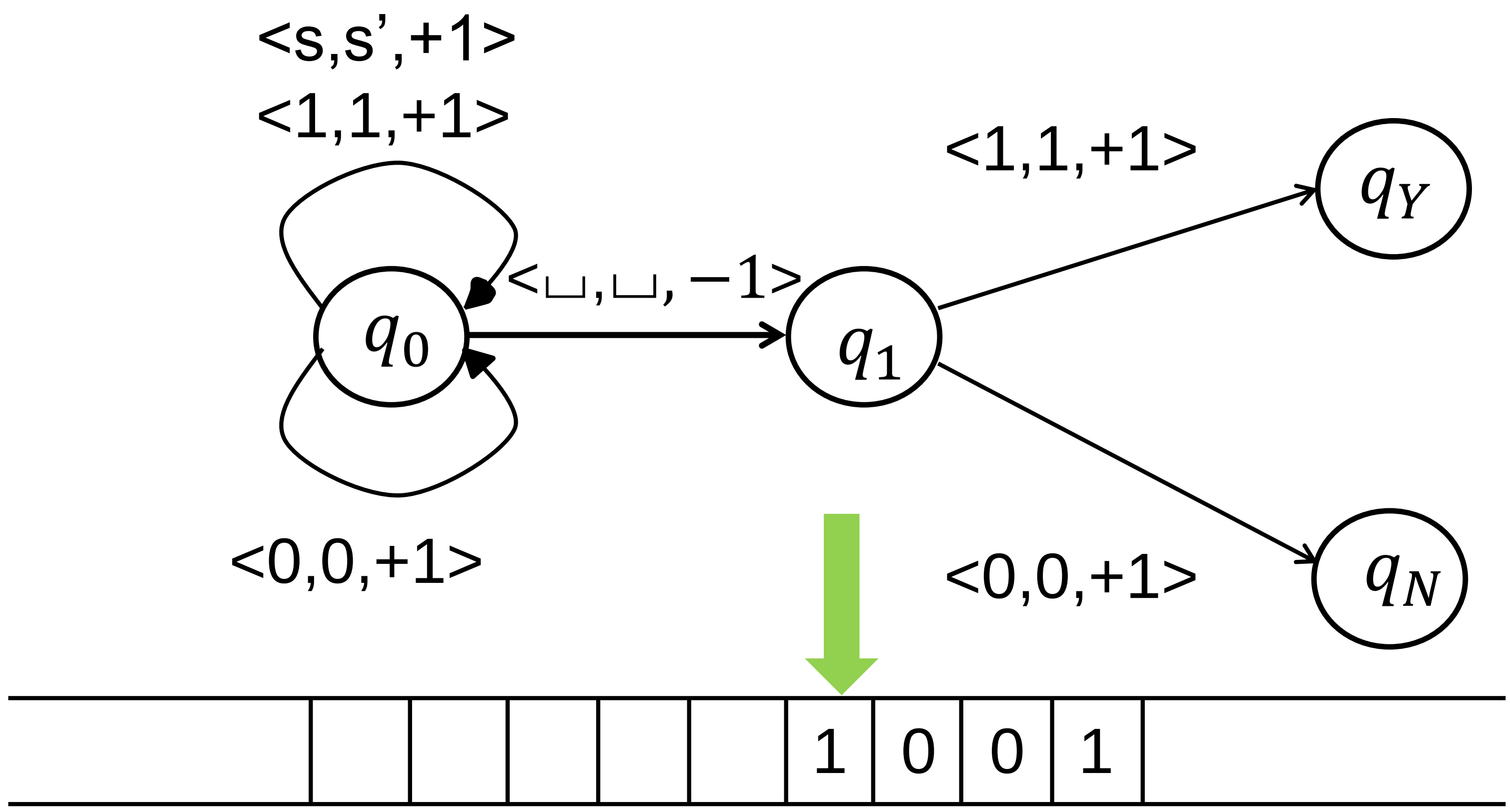
# Program (algorytm) $P$ rozwiązuje problem decyzyjny $\pi$

***Program*** (algorytm) ***$P$ na DMT rozpoznaje język $L$ alfabetu*** $\sum$ jeżeli dla tego języka program $P$ zatrzymuje się w stanie $q_{Y}$.

$L_Y(P) = \{L: P\ \ rozpoznaje\ \ L\}$ - zbiór języków rozpoznawanych przez program $P$ .

# Program (algorytm) $P$ rozwiązuje problem decyzyjny $\pi$

DMT o programie rozpoznającym wszystkie języki zakończone „1” czyli liczby nieparzyste



# Program (algorytm) $P$ rozwiązuje problem decyzyjny $\pi$

***DTM rozwiązuje problem decyzyjny $\pi$ przy kodowaniu*** $e$, jeśli zatrzymuje się dla wszystkich słów wejściowych (reprezentujących konkretne problemy) i kończy obliczenia w stanie $q_Y$ dla wszystkich słów wejściowych $x(I)$ instancji $I$ takich, że $I \epsilon Y_\pi$ i tylko dla nich.

$Y_\pi$ - zbiór wszystkich instancji problemu decyzyjnego $\boldsymbol{\pi}$, dla których odpowiedzią jest „tak".

# Program (algorytm) $P$ rozwiązuje problem decyzyjny $\pi$

Dla problemu decyzyjnego $\pi$, programu $P$, kodowania $e$, języki $L$ alfabetu $\sum$ można podzielić na zbiory:

$L_0$ - języki, które nie kodują instancji problemu $\pi$,

$L_N(P)$ - języki, które kodują instancje problemu $\pi$ i nie są rozpoznawane przez $P$ (osiągany jest stan inny niż $q_Y$)

$L_Y(P)$ - języki, które kodują instancje problemu $\pi$ i są rozpoznawane przez $P$ (osiągany jest stan $q_Y$).

$L(\pi)$ – języki kodujące instancje problemu $\pi$.

$$L(\pi) = L_N(P) \cup L_Y(P)$$

***Program $\boldsymbol{P}$ na DMT rozwiązuje problem $\boldsymbol{\pi}$ przy kodowaniu $\boldsymbol{e}$*** jeśli:

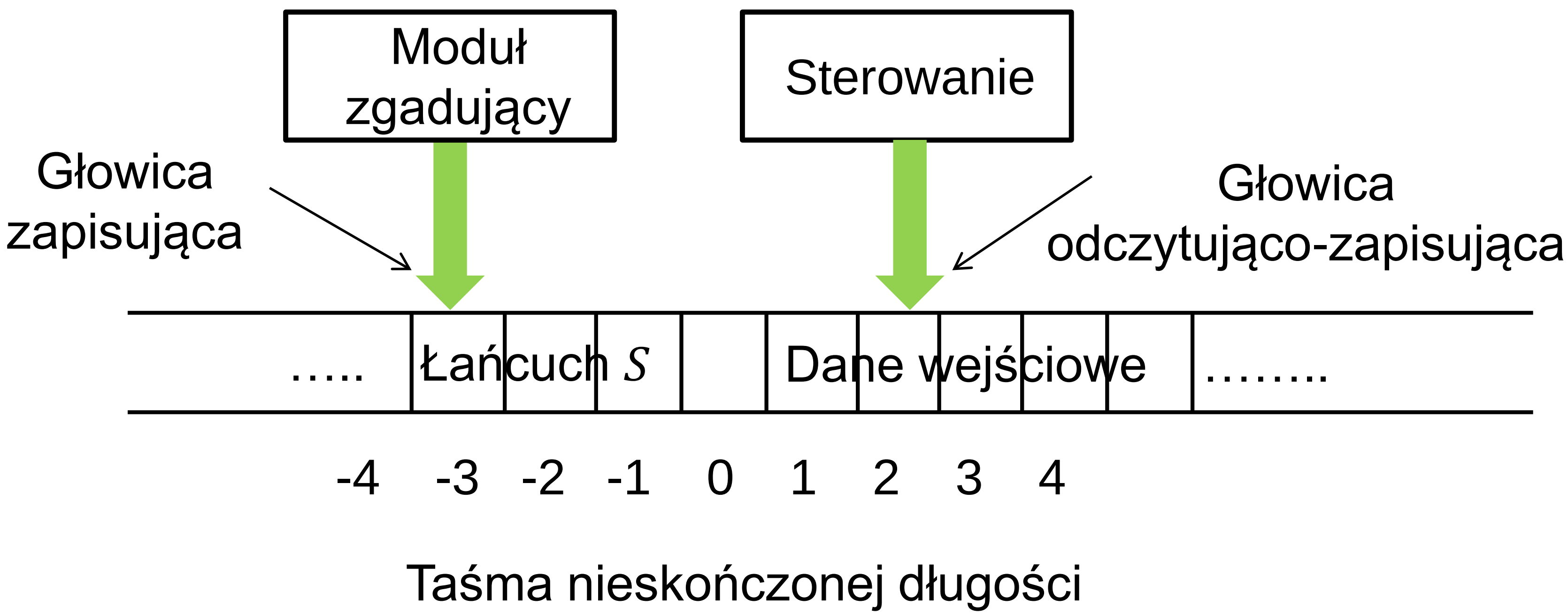
$L_N(P)$ zawiera wszystkie takie języki, które kodują instancje problemu $\pi$, dla których odpowiedź jest „nie" i tylko takie języki,

$L_Y(P)$ zawiera wszystkie takie języki, które kodują instancje problemu $\pi$, dla których odpowiedź jest „tak" i tylko takie języki.

Jeśli czas działania programu (algorytmu) $P$ na DMT rozwiązującego problem decyzyjny $\pi$ jest ograniczony od góry wielomianem zależnym od długości języka $L$, tzn. czas działania $t \leqslant p(|L|)$ dla każdego $L$ i pewnego wielomianu $p$, to algorytm $P$ jest ***algorytmem wielomianowym***.

Jeżeli algorytm nie jest algorytmem wielomianowym, to jest nazywamy ***algorytmem ponadwielomianowym***.

# *Niedeterministyczna jednotaśmowa maszyna Turinga (NDMT)*



# *Niedeterministyczna maszyna Turinga (NDMT)*

**Moduł zgadujący tylko zapisuje na taśmie odgadnięte rozwiązanie** (np. podzbiór zbioru elementów w problemie podziału, kolejność odwiedzania wszystkich miast w problemie komiwojażera, podzbiór elementów do upakowania w plecaku problemu plecakowego).

***Wykonanie programu*** składa się z wielu sekwencyjnych wykonań pary działań:

1. Zgadywania rozwiązania – generowania łańcucha $S$ symboli,
2. Sprawdzania jaka jest odpowiedź na pytanie problemu decyzyjnego dla wygenerowanego rozwiązania.

**Niedeterministyczna jednotaśmowa maszyna Turinga** (NDMT) składa się z DMT i modułu zgadującego (generującego).

Wykonanie programu NDMT dla łańcucha danych $x(I)$ instancji $I$ przebiega następująco:

1. Moduł zgadujący zapisuje na taśmie łańcuch $S$ symboli ze skończonego zbioru symboli taśmy,
2. NDMT sprawdza, tak jak wykonywany jest program przez DMT, czy wygenerowany łańcuch $S$ spełnia warunki pytania instancji $I$.

Dla jednej instancji $I$ może istnieć wiele łańcuchów $S$ reprezentujących rozwiązania. ?

# NDMT dla Problemu podziału zbioru

***Problem podziału zbioru***

Dane:

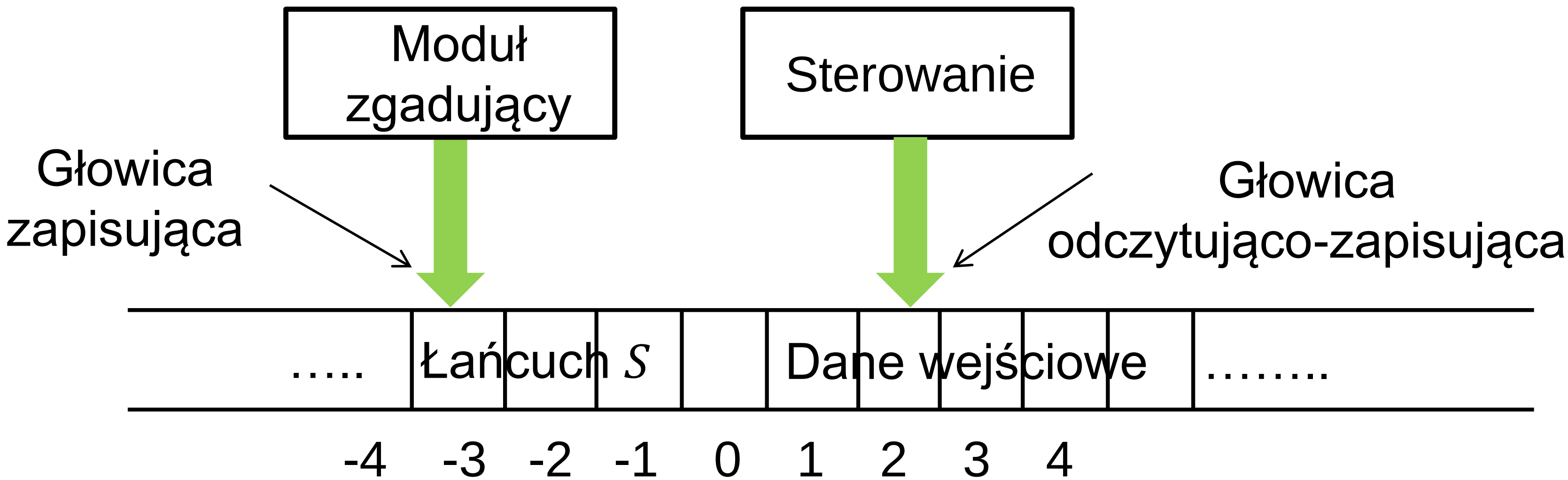
- $X = \{x_1, \ldots, x_i, \ldots, x_k\}$ - zbiór $k$ elementów $x_i \in N_+$, gdzie $N_+ = \{1,2, \ldots\}$,
- $B \in N_+$,
- $\sum_{i=1}^{k} x_i = 2B$.

Pytanie:

Czy istnieje podzbiór $X_1 \subset X$ taki, że

$$\sum_{x_i \in X_1} x_i = B \text{ ?}$$

# NDMT dla Problemu podziału zbioru




Łańcuch $S$ - liczba binarna, której $i$-ta pozycja wskazuje czy $i$-ty element zbioru $X$ należy do wygenerowanego rozwiązania $X_1$

Dane wejściowe

$$x1 \sqcup x2 \sqcup \ldots xk \sqcup B$$

## NDMT dla Problemu podziału zbioru

$2^k - 2$ - liczba generowanych łańcuchów reprezentujących zbiór $X_1$

Czy koszt sprawdzenia relacji
$\sum_{x_i \in X_1} x_i = B$
jest ograniczony od góry przez wielomian od $k$ ?

Jaka byłaby złożoność generowania łańcuchów i sprawdzania, gdyby program był wyrażony w języku wysokiego poziomu.

**NDMT rozwiązuje problem decyzyjny $\pi$** , jeśli dla każdej instancji $I \epsilon D_{\pi}$ są spełnione warunki:

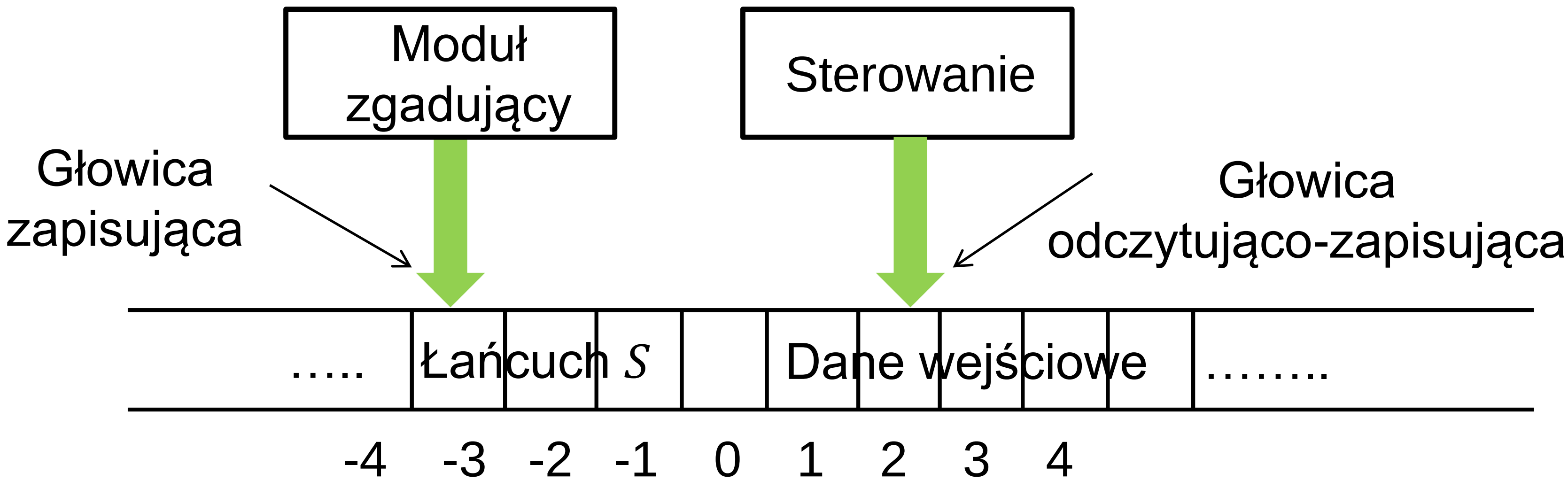
- Jeśli odpowiedź dla $I$ brzmi „tak”, to zostanie wygenerowany łańcuch $S$ , który wraz z $x(I)$ spowoduje, że po wykonaniu programu przez NDMT maszyna ta osiągnie stan końcowy $q_{tak}$ ,
- Jeśli odpowiedź dla $I$ brzmi „nie”, to dla każdego wygenerowanego łańcucha $S$ albo NDMT osiągnie stan końcowy $q_{nie}$, albo etap sprawdzania nie zostanie zakończony.

**NDMT rozwiązuje problem decyzyjny $\pi$ w** (co najwyżej) **wielomianowym czasie**, jeśli dla każdej instancji $I \epsilon D_{\pi}$, dla której odpowiedź brzmi „tak”, zostanie wygenerowany taki łańcuch $S$ , że czas wykonania etapów zgadywania i sprawdzania zakończonego odpowiedzią „tak” przez NDMT (dla $I$ oraz $S$) jest $O\left(p\left(N(I)\right)\right)$ dla pewnego wielomianu $p$ .

Twierdzenie

Jeśli jednotaśmowa NDMT rozwiązuje decyzyjny problem $\pi$ w czasie wielomianowym, to istnieje wielomian $p$ taki, że jednotaśmowa DMT rozwiązuje ten problem w czasie $O(2^{p\left(N(I)\right)})$, gdzie $I \epsilon D_{\pi}$ a $N(I)$ jest rozmiarem danych wejściowych instancji $I$.

# Interpretacja Twierdzenia o relacji między NDMT a DMT




Łańcuch $S$ - liczba binarna, której $i$-ta pozycja wskazuje czy $i$-ty element zbioru $X$ należy do wygenerowanego rozwiązania $X_1$

Dane wejściowe

$$x_1 \sqcup x_2 \sqcup \ldots x_k \sqcup B$$

# Interpretacja Twierdzenia o relacji między NDMT a DMT

Działanie DMT

DMT powinna generować kolejno rozwiązania np. poprzez tworzenie kolejnych liczb binarnych $k$ cyfrowych i sprawdzać jaka jest odpowiedź na pytanie.

# Adekwatność NDMT jako modelu obliczeń

- Sekwencyjny dostęp do danych wejściowych i wyników pośrednich (ze względu na organizację taśmową).
- Nie istnieje rzeczywisty odpowiednik NDMT rozwiązujący problemy decyzyjne w prezentowany sposób.
- NDMT obrazuje zdolność weryfikacji pozytywnej odpowiedzi dla rozwiązania (wygenerowanego łańcucha $S$) instancji $I \epsilon D_\pi$.

**Klasę $\boldsymbol{P}$** tworzą wszystkie problemy decyzyjne, które w co najwyżej wielomianowym czasie może rozwiązać DMT.

**Klasa $\boldsymbol{NP}$** zawiera wszystkie problemy decyzyjne, które w co najwyżej wielomianowym czasie może rozwiązać NDMT.

$$P \subseteq NP$$

Ze względu na wiele lat nieudanych prób udowodnienia relacji $\mathrm{P} = NP$, jest prawie pewne, że:

$$P \subset NP$$

(jest prawie pewne, że $P$ jest właściwą podklasą klasy $NP$).

Jednak czy $P \subset NP$ jest problemem otwartym. ?

- Pytanie, czy problemy NP-zupełne można rozwiązywać w czasie wielomianowym, jest największą zagadką informatyki teoretycznej.
- Problem $P = NP$ czy $P \neq NP$ jest problemem otwartym umieszczonym na liście Problemów milenijnych.

- **Problemy milenijne** (ang. Millennium Prize Problems) - zestaw siedmiu zagadnień matematycznych ogłoszonych przez Instytut matematyczny Claya 24 maja 2000 roku.

- Za rozwiązanie każdego z nich wyznaczono milion dolarów nagrody.

- Do dziś rozwiązano tylko jeden: hipoteza Poincarego została potwierdzona w 2006 przez rosyjskiego matematyka Grigorija Perelmana.