

Automaty i Języki Formalne

Soma Dutta

Wydział Matematyki i Informatyki, UWM w Olsztynie
soma.dutta@matman.uwm.edu.pl

Wykład - 2: Automaty skończone

Semestr zimowy 2023

Podsumowanie

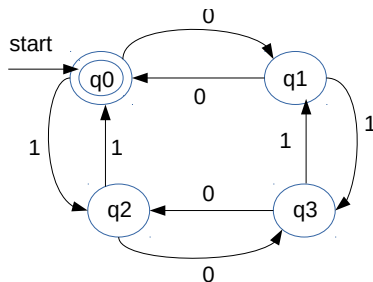
- ▶ Istnieją dwa rodzaje automatów skończonych - deterministyczne i niedeterministyczne.
- ▶ (**Deterministic Finite Automaton**): Deterministyczny automat skończony (DFA) to pięciokrotka $M = (Q, \Sigma, \delta, q_0, F)$ gdzie
 - (i) Q jest skończonym zbiorem stanów,
 - (ii) Σ to skończony zbiór symboli wejściowych, nazywany alfabetem,
 - (iii) $\delta : Q \times \Sigma \mapsto Q$ to funkcja przejścia, która określa następny stan, biorąc pod uwagę bieżący stan i symbol wejściowy,
 - (iv) $q_0 \in Q$ to stan początkowy, i
 - (v) $F \subseteq Q$ to zbiór zawierające wszystkie stany końcowe.
- ▶ Jak ciąg znaków jest przetwarzany w DFA: $\hat{\delta} : Q \times \Sigma^* \mapsto Q$
- ▶ Język DFA $M = (Q, \Sigma, \delta, q_0, F)$ to zbiór ciągów akceptowanych przez M. Tzn., $L(M) = \{w \in \Sigma^* : \hat{\delta}(q_0, w) \in F\}$.

Przykład: Projektowanie DFA akceptującego żądany język

- ▶ Załóżmy, że chcemy skonstruować DFA, który akceptuje język podany jako:
 $L = \{w \in \Sigma^* : w \text{ ma parzystą liczbę zer i parzystą liczbę jedynek}\}$

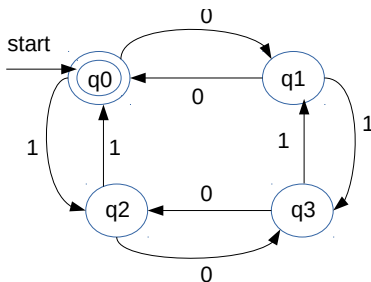
Przykład: Projektowanie DFA akceptującego żądany język

- ▶ Załóżmy, że chcemy skonstruować DFA, który akceptuje język podany jako:
 $L = \{w \in \Sigma^* : w \text{ ma parzystą liczbę zer i parzystą liczbę jedynek}\}$



Przykład: Projektowanie DFA akceptującego żądany język

- ▶ Załóżmy, że chcemy skonstruować DFA, który akceptuje język podany jako:
 $L = \{w \in \Sigma^* : w \text{ ma parzystą liczbę zer i parzystą liczbę jedynek}\}$



$M = (\{q_0, q_1, q_2, q_3\}, \{0, 1\}, \delta, q_0, \{q_0\})$ gdzie funkcja przejścia jest zdefiniowana poprzez diagram przejścia.

Niedeterministyczny automat skończony (Non-deterministic finite automaton)

- ▶ Niedeterministyczny automat skończony (Ndfa) różni się od DFA tym, że pozwala na wybór ruchów, zamiast mieć unikalny ruch stojąc w pewnym stanie po odczytaniu symbolu wejściowego. Formalnie osiągamy to poprzez zdefiniowanie przejścia funkcję tak, aby jej zakres był zbiorem możliwych stanów.
- ▶ Język akceptowany przez DFA jest również akceptowany przez Ndfa. Jednak zaprojektowanie Ndfa dla danego języka jest często znacznie łatwiejsze niż zaprojektowanie DFA dla tego samego języka. Co więcej, póki możemy, zawsze przekonwertować Ndfa na DFA, odpowiedni DFA może mieć wykładniczo więcej stanów niż Ndfa.

Przykład: potrzeba projektowania NDFA

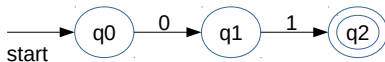
- ▶ Załóżmy, że chcemy skonstruować automat skończony, który akceptuje wszystkie ciągi zer i jedynek zakończone na 01.

Przykład: potrzeba projektowania NDFA

- ▶ Załóżmy, że chcemy skonstruować automat skończony, który akceptuje wszystkie ciągi zer i jedynek zakończone na 01.
- ▶ Najprostszy przypadek to zaprojektowanie diagramu przejścia akceptującego 01.

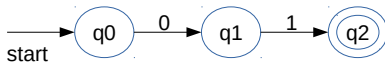
Przykład: potrzeba projektowania NDFA

- ▶ Załóżmy, że chcemy skonstruować automat skończony, który akceptuje wszystkie ciągi zer i jedynek zakończone na 01.
- ▶ Najprostszy przypadek to zaprojektowanie diagramu przejścia akceptującego 01.



Przykład: potrzeba projektowania NDFA

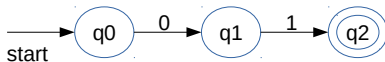
- ▶ Załóżmy, że chcemy skonstruować automat skończony, który akceptuje wszystkie ciągi zer i jedynek zakończone na 01.
- ▶ Najprostszy przypadek to zaprojektowanie diagramu przejścia akceptującego 01.



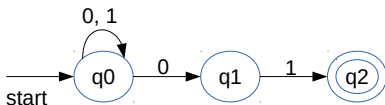
- ▶ Ale teraz chcemy, aby akceptowane były wszystkie ciągi znaków w postaci $\omega 01$, gdzie ω może być dowolną kombinacją zer i jedynek. Najprostszy sposób to dodanie pętli do q_0 z obydwoma wejściami 0 i 1.

Przykład: potrzeba projektowania NDFA

- ▶ Załóżmy, że chcemy skonstruować automat skończony, który akceptuje wszystkie ciągi zer i jedynek zakończone na 01.
- ▶ Najprostszy przypadek to zaprojektowanie diagramu przejścia akceptującego 01.

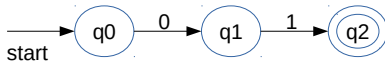


- ▶ Ale teraz chcemy, aby akceptowane były wszystkie ciągi znaków w postaci $\omega 01$, gdzie ω może być dowolną kombinacją zer i jedynek. Najprostszy sposób to dodanie pętli do q_0 z obydwoma wejściami 0 i 1.

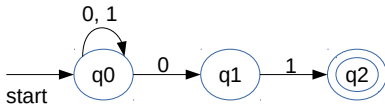


Przykład: potrzeba projektowania NDFA

- ▶ Załóżmy, że chcemy skonstruować automat skończony, który akceptuje wszystkie ciągi zer i jedynek zakończone na 01.
- ▶ Najprostszy przypadek to zaprojektowanie diagramu przejścia akceptującego 01.



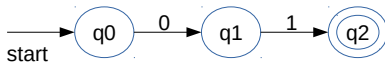
- ▶ Ale teraz chcemy, aby akceptowane były wszystkie ciągi znaków w postaci $\omega 01$, gdzie ω może być dowolną kombinacją zer i jedynek. Najprostszy sposób to dodanie pętli do q_0 z obydwoma wejściami 0 i 1.



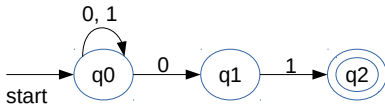
- ▶ Nie wolno nam jednak tego robić w przypadku DFA bo od węzła q_0 z wejściem 0 mamy wtedy dwa możliwe stany do przejścia. Ponadto funkcja przejścia nie jest zdefiniowana dla przypadków, gdy automat jest w stanie q_1 i czyta symbol 0, oraz w stanie q_2 i czyta symbol albo 0, albo 1.

Przykład: potrzeba projektowania NDFA

- ▶ Załóżmy, że chcemy skonstruować automat skończony, który akceptuje wszystkie ciągi zer i jedynek zakończone na 01.
- ▶ Najprostszy przypadek to zaprojektowanie diagramu przejścia akceptującego 01.



- ▶ Ale teraz chcemy, aby akceptowane były wszystkie ciągi znaków w postaci $\omega 01$, gdzie ω może być dowolną kombinacją zer i jedynek. Najprostszy sposób to dodanie pętli do q_0 z obydwoma wejściami 0 i 1.



- ▶ Nie wolno nam jednak tego robić w przypadku DFA bo od węzła q_0 z wejściem 0 mamy wtedy dwa możliwe stany do przejścia. Ponadto funkcja przejścia nie jest zdefiniowana dla przypadków, gdy automat jest w stanie q_1 i czyta symbol 0, oraz w stanie q_2 i czyta symbol albo 0, albo 1.
- ▶ To jest to, na co pozwala nam NDFA. Podczas gdy DFA ma dokładnie jeden łuk z każdego stanu dla każdego symbolu wejściowego, NDFA nie ma takich ograniczeń.

Definicja niedeterministycznego automatu skończonego

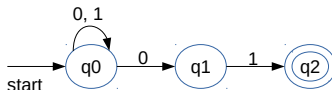
- ▶ Niedeterministyczny automat skończony (N DFA) to pięciokrotka

$M = (Q, \Sigma, \delta, q_0, F)$ gdzie

- (i) Q jest skończonym zbiorem stanów,
- (ii) Σ to skończony zbiór symboli wejściowych, nazywany alfabetem,
- (iii) $\delta : Q \times \Sigma \mapsto P(Q)$ to funkcja przejścia, która przyjmuje stan w Q i symbol wejściowy jako argumenty i zwraca podzbiór Q - czyli określa następne możliwe stany, biorąc pod uwagę bieżący stan i symbol wejściowy,
- (iv) $q_0 \in Q$ to stan początkowy, i
- (v) $F \subseteq Q$ to zbiór zawierające wszystkie stany końcowe.

Przykład: tabela przejścia NDFA

$$M = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$$



	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	ϕ	$\{q_2\}$
q_2	ϕ	ϕ

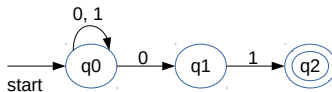
- ▶ Zatem widzmy, że w przeciwieństwie do DFA, w przypadku NDFA mamy nawet pusty zbiór jako wyjście funkcji przejściowej, gdy automat jest w pewnym stanie i czyta jakieś symbole, a w niektórych przypadkach wyjście jest zbiorem kilku możliwych stanów.
- ▶ Tzn., że w przypadku pewnego stanu, jeśli automaty odczytają określony symbol, może nastąpić więcej niż jeden ruch lub nawet z pewnego stanu nie ma ruchu dla niektórych określonych symboli.

NDFA: rozszerzenie funkcji przejścia na ciągi znaków

- ▶ Podobnie jak w przypadku DFA, również w przypadku NDFA należy rozszerzyć funkcję przejścia δ do odpowiedniej $\hat{\delta}$ tak, że mając stan q i ciąg symboli ω , zwraca zbiór możliwych stanów gdzie automat może dotrzeć po przetworzeniu wszystkich symboli w ciągu ω .

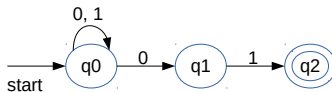
NDFA: rozszerzenie funkcji przejścia na ciągi znaków

- ▶ Podobnie jak w przypadku DFA, również w przypadku NDFA należy rozszerzyć funkcję przejścia δ do odpowiedniej $\hat{\delta}$ tak, że mając stan q i ciąg symboli ω , zwraca zbiór możliwych stanów gdzie automat może dotrzeć po przetworzeniu wszystkich symboli w ciągu ω .
- ▶ Zobaczmy, jak należy zdefiniować na przykładzie.



NDFA: rozszerzenie funkcji przejścia na ciągi znaków

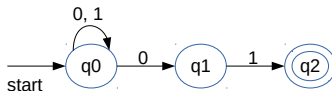
- ▶ Podobnie jak w przypadku DFA, również w przypadku NDFA należy rozszerzyć funkcję przejścia δ do odpowiedniej $\hat{\delta}$ tak, że mając stan q i ciąg symboli ω , zwraca zbiór możliwych stanów gdzie automat może dotrzeć po przetworzeniu wszystkich symboli w ciągu ω .
- ▶ Zobaczmy, jak należy zdefiniować na przykładzie.



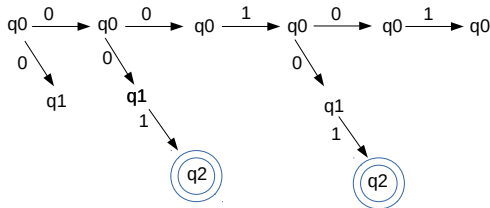
- ▶ Zobaczmy, jak przetwarzany jest ciąg 00101.

NDFA: rozszerzenie funkcji przejścia na ciągi znaków

- ▶ Podobnie jak w przypadku DFA, również w przypadku NDFA należy rozszerzyć funkcję przejścia δ do odpowiedniej $\hat{\delta}$ tak, że mając stan q i ciąg symboli ω , zwraca zbiór możliwych stanów gdzie automat może dotrzeć po przetworzeniu wszystkich symboli w ciągu ω .
- ▶ Zobaczmy, jak należy zdefiniować na przykładzie.

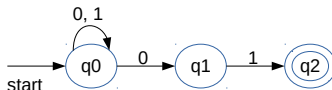


- ▶ Zobaczmy, jak przetwarzany jest ciąg 00101.

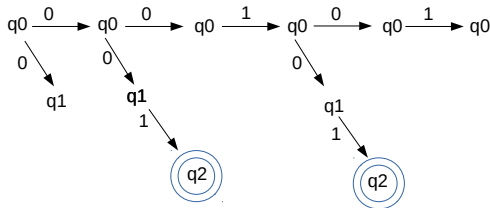


NDFA: rozszerzenie funkcji przejścia na ciągi znaków

- ▶ Podobnie jak w przypadku DFA, również w przypadku NDFA należy rozszerzyć funkcję przejścia δ do odpowiedniej $\hat{\delta}$ tak, że mając stan q i ciąg symboli ω , zwraca zbiór możliwych stanów gdzie automat może dotrzeć po przetworzeniu wszystkich symboli w ciągu ω .
- ▶ Zobaczmy, jak należy zdefiniować na przykładzie.



- ▶ Zobaczmy, jak przetwarzany jest ciąg 00101.



- ▶ Zatem zaczynając od q_0 po przetworzeniu całego ciągu 00101 automat może dojść do jednego ze stanów z $\{q_0, q_2\}$.

Podsumowanie etapów przetwarzania ciągu 00101

- ▶ $\hat{\delta}(q_0, \Lambda) = \{q_0\}$
- ▶ $\hat{\delta}(q_0, 0) = \{q_0, q_1\}$
- ▶ $\hat{\delta}(q_0, 00) = \hat{\delta}(q_0, 0) \cup \hat{\delta}(q_1, 0) = \{q_0, q_1\} \cup \phi$
- ▶ $\hat{\delta}(q_0, 001) = \hat{\delta}(q_0, 1) \cup \hat{\delta}(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$
- ▶ $\hat{\delta}(q_0, 0010) = \hat{\delta}(q_0, 0) \cup \hat{\delta}(q_2, 0) = \{q_0, q_1\} \cup \phi = \{q_0, q_1\}$
- ▶ $\hat{\delta}(q_0, 00101) = \hat{\delta}(q_0, 1) \cup \hat{\delta}(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

Stan końcowy to $\{q_2\}$. Więc, $\hat{\delta}(q_0, 00101) = \{q_0, q_2\} \cap \{q_2\} \neq \phi$. To oznacza, 00101 jest akceptowany przez dany NFA.

NDFA: definicja funkcji przejścia na ciąg znaków

- ▶ $\hat{\delta} : Q \times \Sigma^* \mapsto P(Q)$ jest zdefiniowany rekurencyjnie w następujący sposób.
 - (i) $\hat{\delta}(q, \Lambda) = \{q\}$.
 - (ii) Dla $wa \in \Sigma^*$ gdzie $a \in \Sigma$, jeśli $\hat{\delta}(q, w) = \{q_1, q_2, \dots, q_k\}$,
wtedy $\hat{\delta}(q, wa) = \cup_{i=1}^k \delta(q_i, a)$.
- ▶ Oznacza to, że po odczytaniu pierwszej części ciągu, czyli w , jeśli automat może dotrzeć jeden ze stanów $\{q_1, q_2, \dots, q_k\}$, to musimy wziąć pod uwagę wszystkie te stany, z których automat może dotrzeć każdy z tych stanów $\{q_1, q_2, \dots, q_k\}$ po odczytaniu a .
- ▶ **Definicja:** Język NDFA $M = (Q, \Sigma, \delta, q_0, F)$ to zbiór ciągów akceptowanych przez M . Tzn.,
 $L(M) = \{w \in \Sigma^* : \hat{\delta}(q_0, w) \cap F \neq \emptyset\}$.

Równoważność pomiędzy DFA i NDFA

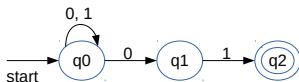
- ▶ Chociaż istnieje wiele języków, dla których NDFA jest łatwiejszy do skonstruowania niż DFA, taki jak język ciągów znaków kończących się na 01, jest to zaskakujący fakt, że każdy język, który może zostać opisany przez jakiś NDFA, może to być również opisywane przez niektóre DFA. Co więcej, DFA w praktyce ma około wielu stanów jak NDFA, chociaż często ma więcej przejść. W najgorszym przypadku jednak najmniejszy DFA może mieć 2^n stanów, a najmniejszy NDFA na ten sam język ma tylko n stanów.
- ▶ Aby znaleźć DFA, który akceptuje ten sam język, co język danego NDFA, musimy najpierw nauczyć się techniki zwanej konstrukcją podzbioru (subset construction), aby przekonwertować NDFA na odpowiedni DFA.

Etapy konwersji NDFA na odpowiedni DFA

- ▶ Załóżmy, że mamy NDFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$.
- ▶ Chcemy skonstruować odpowiedni DFA $D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ w taki sposób, że $L(N) = L(D)$.
- ▶ Zauważmy najpierw, że alfabet wejściowy zarówno dla N , jak i D jest taki sam, a stan początkowy DFA D jest zbiorem zawierającym tylko stan początkowy NDFA N .
- ▶ Q_D jest zbiorem wszystkich podzbiorów Q_N : Tzn., $Q_D = P(Q_N)$. To znaczy, jeśli $|Q_N| = n$ wtedy $|Q_D| = 2^n$. Jednakże nie wszystkie stany Q_D są osiągalne ze stanu początkowego Q_D i dlatego nie są brane pod uwagę w praktyce.
- ▶ F_D jest zbiorem wszystkich podzbiorów S z Q_N takich, że S ma co najmniej jeden wspólny stan F_N : Tzn.
$$F_D = \{S \subseteq Q_N : S \cap F_N \neq \emptyset\}$$
- ▶ Dla każdego $S \subseteq Q_N$ i $a \in \Sigma$, $\delta_D(S, a) = \cup_{q \in S} \delta_N(q, a)$
Tzn., sprawdzamy wszystkie przejścia z każdego stanu S w N w odniesieniu do symbolu wejściowego a i rozważamy zbiór wszystkich tych stanów, do których N może dotrzeć po odczytaniu a z każdego stanu S .

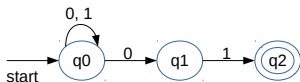
Przykład konwersji NDFA na odpowiedni DFA

- Mamy NDFA: $N = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta_N, q_0, \{q_2\})$ gdzie δ_N jest opisana na poniższym schemacie przejścia.



Przykład konwersji NDFA na odpowiedni DFA

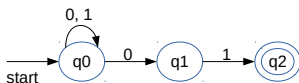
- ▶ Mamy NDFA: $N = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta_N, q_0, \{q_2\})$ gdzie δ_N jest opisana na poniższym schemacie przejścia.



- ▶ Odpowiedni DFA: $D = (Q_D, \{0, 1\}, \delta_D, \{q_0\}, F_D)$ gdzie

Przykład konwersji NDFA na odpowiedni DFA

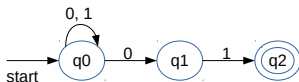
- ▶ Mamy NDFA: $N = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta_N, q_0, \{q_2\})$ gdzie δ_N jest opisana na poniższym schemacie przejścia.



- ▶ Odpowiedni DFA: $D = (Q_D, \{0, 1\}, \delta_D, \{q_0\}, F_D)$ gdzie
 $Q_D = \{\emptyset, \{q_0\}, \{q_1\}, \{q_2\}, \{q_0, q_1\}, \{q_0, q_2\}, \{q_1, q_2\}, \{q_0, q_1, q_2\}\}$

Przykład konwersji N DFA na odpowiedni DFA

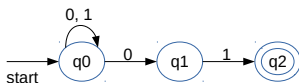
- Mamy N DFA: $N = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta_N, q_0, \{q_2\})$ gdzie δ_N jest opisana na poniższym schemacie przejścia.



- Odpowiedni DFA: $D = (Q_D, \{0, 1\}, \delta_D, \{q_0\}, F_D)$ gdzie
 $Q_D = \{\phi, \{q_0\}, \{q_1\}, \{q_2\}, \{q_0, q_1\}, \{q_0, q_2\}, \{q_1, q_2\}, \{q_0, q_1, q_2\}\}$
 $F_D = \{\{q_2\}, \{q_0, q_2\}, \{q_1, q_2\}, \{q_0, q_1, q_2\}\}$

Przykład konwersji N DFA na odpowiedni DFA

- Mamy N DFA: $N = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta_N, q_0, \{q_2\})$ gdzie δ_N jest opisana na poniższym schemacie przejścia.



- Odpowiedni DFA: $D = (Q_D, \{0, 1\}, \delta_D, \{q_0\}, F_D)$ gdzie
 $Q_D = \{\phi, \{q_0\}, \{q_1\}, \{q_2\}, \{q_0, q_1\}, \{q_0, q_2\}, \{q_1, q_2\}, \{q_0, q_1, q_2\}\}$
 $F_D = \{\{q_2\}, \{q_0, q_2\}, \{q_1, q_2\}, \{q_0, q_1, q_2\}\}$

	0	1
ϕ	ϕ	ϕ
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	ϕ	$\{q_2\}$
δ_D $\{q_2\}$	ϕ	ϕ
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1, q_2\}$	ϕ	$\{q_2\}$
$\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$

Rewizja tabeli przejść w oparciu o znaczące przejścia

- W tabeli przejść ze stanu początkowego $\{q_0\}$ możliwe jest dotarcie tylko do kilku stanów (podświetlonych na niebiesko). Zatem pozostałe stany nie mają znaczenia dla rozpatrywanego DFA.

	0	1
ϕ	ϕ	ϕ
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	ϕ	$\{q_2\}$
δ_D * $\{q_2\}$	ϕ	ϕ
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
* $\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
* $\{q_1, q_2\}$	ϕ	$\{q_2\}$
* $\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$

- Więc mamy odpowiedni DFA $D = (\{\{q_0\}, \{q_0, q_1\}, \{q_0, q_2\}\}, \{0, 1\}, \delta_D, \{q_0\}, \{q_0, q_2\})$ i poniższy diagram przejścia dla δ_D .

