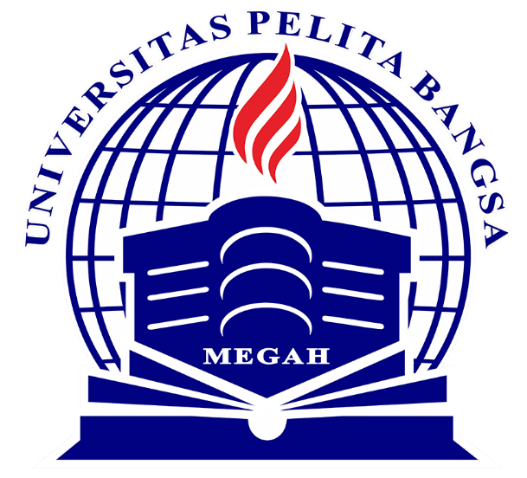
**MAKALAH**

**“EKUIVALENSI NFA KE DFA”**



**DOSEN PEMBIMBING :**

**Asep Muhidin, S.Kom, M.Kom**

**Disusun Oleh :**

* Nama : Diki setiawan
* Kelas : TI 18.D6

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS PELITA BANGSA**

**2020/2021**

**DAFTAR ISI**

**DAFTAR ISI**

[**KATA PENGANTAR** 2](#_Toc38827573)

[**BAB I** 2](#_Toc38827574)

[**PENDAHULUAN** 2](#_Toc38827575)

[**1.1 Latar Belakang** 2](#_Toc38827576)

[**1.2 Rumusan Masalah** 3](#_Toc38827577)

[**1.3 Tujuan dan Manfaat** 3](#_Toc38827578)

[**1.3.1 Tujuan** 3](#_Toc38827579)

[**1.3.2 Manfaat** 4](#_Toc38827580)

[**BAB II** 4](#_Toc38827581)

[**2.1.Pengertian Deterministic Finite Automata** 4](#_Toc38827582)

[**2.2.Pengertian Non-Deterministic Finite Automata** 4](#_Toc38827583)

[**2.3 Perbedaan Ekuivalensi NFA dan DFA** 4](#_Toc38827584)

[**2.4. Ekivalensi-Deterministic Finite Automata ke Deterministic FiniteAutomata** . 6](#_Toc38827585)

[**2.5 Konversi dari NFA ke DFA** 7](#_Toc38827586)

[**BAB II** 9](#_Toc38827587)

[**PENUTUP** 9](#_Toc38827588)

[**3.1 Kesimpulan** 9](#_Toc38827589)

[**3.2 Saran** 9](#_Toc38827590)

[**DAFTAR PUSTAKA** 10](#_Toc38827591)

# **KATA PENGANTAR**

Puji dan syukur kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmatnya penulis dapat menyelesaikan makalah Ekuivalensi Non Deterministic Finite Automata ke Deterministic Finite Automataini. Selain sebagai tugas, makalah ini dibuat untuk menambah pengetahuan dan ilmu kita tentang Finite Automata Banyak sekali hambatan dalam penyusunan makalah ini baik itu masalah waktu, sarana, dan lain lain. Oleh sebab itu, Selesainya Makalah ini bukan semata mata karena kemampuan penulis,banyak pihak yang mendukung dan membantu kami. Dalam kesempatan ini, penulis mengucapkan terima kasih banyak kepada pihak pihak yang telah membantu. Saya harapkan makalah ini nantinya akan berguna bagi para pembaca.jika ada kesalahan dalam makalah ini kami mengharapkan kritik dan saran yang membangun agar dapat lebih baik.

# **BAB I**

# **PENDAHULUAN**

## **1.1 Latar Belakang**

Dalam hierarki kelas-kelas bahasa menurut Chomsky, kelas bahasa yang paling sederhana adalah kelas bahasa reguler (regular languages). Bahasa reguler dapat dengan tepat di deskripsikan dengan menggunakan finite automata (FA); dengan kata lain bahasa yang dapat diterima oleh suatu finite automata adalah bahasa reguler. Finite automata merupakan mesin abstrak yang berupa sistem model matematika dengan masukan dan keluaran diskrit yang dapat mengenali bahasa paling sederhana (bahasa reguler) dan dapat diimplementasikan secara nyata di mana sistem dapat berada di salah satu dari sejumlah berhingga konfigurasi internal disebut state. Banyak model perangkat keras dan perangkat lunak yang menggunakan finite automata sebagai penerapannya. Beberapa contoh penerapan finite automata dalam perangat keras dan perangkat lunak adalah dalam perancangan dan pemantauan perilaku rangkaian digital, scanning dokumen teks dalam halaman web guna menemukan kesamaan kata, frase dan bentuk lain (Hopcroft et al., 2007). Terdapat dua jenis finite automata, yaitu deterministik finite automata (DFA) dan nondeterministik finite automata (NFA). Perbedaan di antara kedua jenis finite automata tersebut terletak pada kontrol terhadap finite automata tersebut (Hopcroft et al., 2007). Deterministik finite automata (DFA) bersifat deterministik, yang berarti bahwa automata tersebut tidak dapat berada di lebih dari satu state pada saat yang bersamaan, sedangkan non-deterministik finite automata (NFA) bersifat non-deterministik, yang berarti bahwa 2 automata tersebut dapat berada di beberapa state pada saat yang bersamaan atau dengan kata lain NFA dapat menebak di state mana dia berikutnya akan berada (Hopcroft et al., 2007). Ada banyak bahasa yang apabila digunakan akan membuat NFA lebih mudah dibangun dibandingkan jika dibangun menggunakan DFA. Suatu bahasa yang dibangun menggunakan NFA ternyata tidak lebih powerful dibandingkan dengan ketika dibangun menggunakan DFA.

## **1.2 Rumusan Masalah**

Berikut ini adalah perumusan masalah yang dihadapi untuk diselesaikan adalah :

1. Apa pengertian dari NFA?

2. Apa pengertian dari DFA?

3. Apa perbedaan NFA dengan DFA?

4. Bagaimana terjadinya ekuivalensi NFA ke DFA?

5. Bagaimana konversi NFA ke DFA yang ber ekuivalen?

## **1.3 Tujuan dan Manfaat**

### **1.3.1 Tujuan**

Berdasarkan masalah-masalah di atas maka cakupan tujuan penelitian ini secara rinci dapat dirumuskan sebagai berikut :

1. Mengetahui lebih dalam tentang NFA dan DFA dan bisa mengerti alur dari konversi NFA ke DFA

### **1.3.2 Manfaat**

Dengan mengacu pada tujuan penelitian di atas, maka manfaat penelitian meliputi hal-hal sebagai berikut :

1. Sebagai bahan penelitian dan pengembangan

# **BAB II**

## **2.1.Pengertian Deterministic Finite Automata**

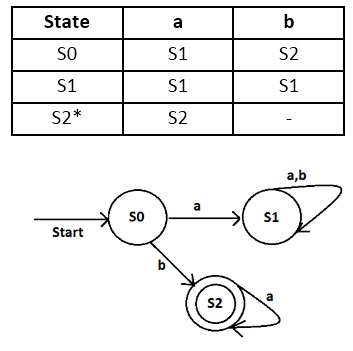
Deterministic Finite Automata merupakan sebuah fungsi yang harus terdefinisi untuk semuapasangan state-input yang ada didalam Q X ∑. Deterministik finite automata (DFA) bersifat deterministik, yang berarti bahwa automata tersebut tidak dapat berada di lebih dari satu state pada saat yang bersamaan.

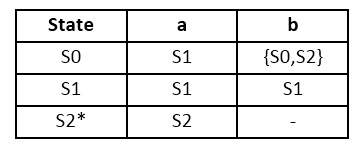
## **2.2.Pengertian Non-Deterministic Finite Automata**

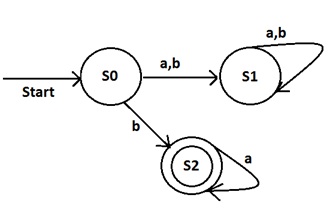
Non-deterministik finite automata (NFA) bersifat non-deterministik, yang berarti bahwa automata tersebut dapat berada di beberapa state pada saat yang bersamaan atau dengan kata lain NFA dapat menebak di state mana dia berikutnya akan berada (Hopcroft et al., 2007).Pada Non-Determinaistic Finite Automata (NFA) dari suatu state bisa terdapat 0,1, atau lebih busur keluar (transisi) berlabel simbol input yang sama. Non-Determinaistic Finite Automata didefenisikan pula dengan lima(5) M=(Q , Σ , δ , S , F )dengan arti yang serupa pada Deterministic Fnite Automata. Disini perbedaan ada padafungsi transisinya, dimana untuk setiap pasangan state-input, kita bisa memiliki 0 (nol) atau lebih pilihan untuk state berikutnya.

## **2.3 Perbedaan Ekuivalensi NFA dan DFA**

Finite Automata adalah mesin automata dari suatu Bahasa regular. Finite Automata memiliki jumlah state yang banyaknya berhingga dan dapat berpindah-pindah dari suate state ke state yang lainnya. Finite Automata dibagi menjadi Deterministic Finite Automata (DFA) dan Non Deterministic Finite Automata (NFA).

[](https://socs.binus.ac.id/files/2019/12/vn-1.jpg)**Berikut contoh dari Deterministic Finite Automata :**

[](https://socs.binus.ac.id/files/2019/12/vn-2.jpg)**Berikut contoh dari Non Deterministic Finite Automata (NFA) :**

[](https://socs.binus.ac.id/files/2019/12/vn-3.jpg)

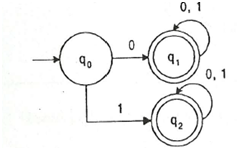
Berdasarkan contoh Deterministic Finite Automata (DFA) dan Non Deterministic Finite Automata (NFA) yang ada di atas, terlihat perbedaan antara DFA dan NFA yaitu :

* Pada Deterministic Finite Automata, jika suatu state diberi inputan maka state tersebut akan selalu tepat menuju satu state
* Pada Non Deterministic Finite Automata, jika suatu state diberi inputan maka mungkin saja bisa menuju ke beberapa state berikutnya. Dapat dilihat di S0, jika diberi inputan b bisa menuju ke S1 dan S2.

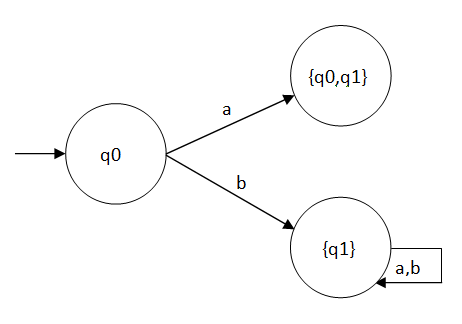
## **2.4. Ekivalensi-Deterministic Finite Automata ke Deterministic FiniteAutomata** Ada banyak bahasa yang apabila digunakan akan membuat NFA lebih mudah dibangun dibandingkan jika dibangun menggunakan DFA. Suatu bahasa yang dibangun menggunakan NFA ternyata tidak lebih powerfull dibandingkan dengan ketika dibangun menggunakan DFA. Setiap bahasa yang dapat dideskripsikan oleh suatu NFA ternyata dapat pula dideskripsikan oleh satu DFA. Bukti bahwa DFA dapat melakukan apa saja yang dapat dilakukan NFA melibatkan suatu konstruksi yang disebut dengan subset construction. Subset construction adalah prosedur untuk mentransformasikan suatu NFA menjadi DFA (Hopcroft et al., 2007).

Jumlah state yang dimiliki oleh DFA maupun oleh NFA kurang lebih sama pada kebanyakan kasus tetapiberbeda dalam jumlah transisi yang dimiliki oleh keduanya. Pada sebagian kecil kasus, untuk membuat suatu DFA yang mengungkapkan bahasa yang sama dengan suatu NFA dengan  jumlah state n, bisa jadidalam kasus terburukdiperlukan 2 state (Hopcroft et al., 2007). Hopcroft et al. (2007) menyatakan bahwa salah satu bentuk perluasan dari finiteautomata adalah finite automata dengan transisi epsilon (ǫ). NFA yang memiliki ǫ (ǫ-NFA)memungkinkan NFA tersebut untuk menerima transisi ǫ atau string kosong.Lebih lanjut efeknya pada NFA adalah memungkinkan terjadinya transisi spontan tanpa menerima simbol masukan. Seperti halnya sifat non-deterministik pada finite automata, penambahan transisi ǫ ini tidak memperluas kelas bahasa yang dapat diterima oleh suatu finite automata. Perluasan ini hanya akan memberikan kemudahan dalam membangun suatu automata. DFA hasil transformasi dari suatu NFA bukanlah suatu DFA yang minimal. Untuk suatu DFA, dapat menemukan DFA yang ekuivalen yang memiliki jumlah state yang lebih sedikit atau sama dengan semua DFA yang menerima bahasa yang sama (Hopcroft et al., 2007).

Selain itu juga, untuk membantu mahasiswa dan dosen dalam hal pengujian DFA dan NFA maka dibuatlah sebuah compiler yang dapat menunjukkan perubahan suatu finite automata dari suatu bentuk representasi ke bentuk representasi yang lain.Tahapan Pengubahan Non-Determinaistic Finite Automata ke Deterministic Finite Automata Dari semua mesin Non-deterministic Finite Automata dapat dibuat mesin Deterministic Finite Automata-nya yang ekivalen (bersesuaian).Ekivalensi disini artinya mampu menerima bahasa yang sama.

[](http://1.bp.blogspot.com/-hP2KU2iJ8V4/VUpFTxoiLdI/AAAAAAAABJE/32GMI1LK11Q/s1600/Untitled.png)  
**Gambar 2.1 Mesin DFA**

## **2.5 Konversi dari NFA ke DFA**

[](http://1.bp.blogspot.com/-kDZCrR1blGo/UJwCmerVcYI/AAAAAAAAANA/De0VBrdRHsE/s1600/NFA-DFA1.png)Berdasarkan *tabel transisi* pada NFA, kita gambarkan diagram transisi DFA nya terlebih dahulu.

|  |
| --- |
|  |
|  |  |

Kemudian tentukan arah dua busur keluar untuk ***State {q0,q1}*** yaitu busur arah ‘**a**’ dan ‘**b**’. Hal ini karena untuk DFA masing masing state harus memiliki pasangan busur, dalam soal ini busur ‘**a**’ dan ‘**b**’.

***Busur arah ‘a’ :***

d ({q0,q1}, a)   = {q0,a} È {q1,a}

                        = {q0,q1} È {q1}

                        = {q0,q1}

***Busur arah ‘b’ :***

d ({q0,q1}, b)   = {q0,b} È {q1,b}

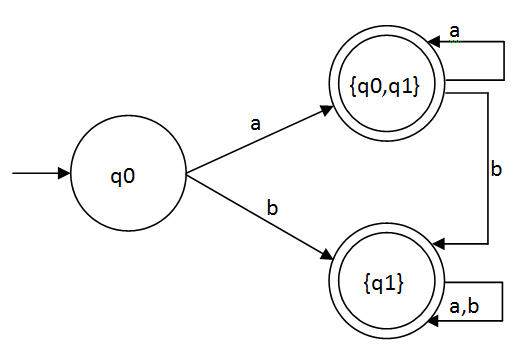
                        = {q1} È {q1}

                        = {q1}

Selanjutnya menentukan state akhir, yaitu kita ingat bahwa **F = {q1}** ketika masih NFA maka himpunan state akhir (**F**) sekarang adalah semua yang mengandung state **q1**.

Maka,**F = {{q1}, {q0, q1}}**

**Gambar Diagram Transisi Akhir setelah di konversi ke DFA**



|  |
| --- |
|  |
|  |  |

# 

# **BAB II**

# **PENUTUP**

## **3.1 Kesimpulan**

Suatu bahasa yang dibangun menggunakan NFA ternyata tidak lebih powerfull dibandingkan dengan ketika dibangun menggunakan DFA.

## **3.2 Saran**

Penulis menjelaskan NFA dan DFA nya harus lebih detail lagi dan kurang dalam menyampaikan materinya, mungkin kedepan nya penulisan makah NFA dan DFA ini biar lebih baik lagi dan lebih detail

# **DAFTAR PUSTAKA**

1. <http://raudhatuljannah97.blogspot.com/2017/05/nfa-ke-dfa-teori-bahasa-otomata.html>
2. <https://socs.binus.ac.id/2019/12/21/teknik-kompilasi-konversi-nfa-ke-dfa/>