令和 6 年度卒業研究発表会（日本大学工学部情報工学科）
A-3-19

授業支援を目的としたオートマトン作成システムの開発

Development of an Automaton Creation System to Support Classroom Teaching

20216136　野地　駆　[関澤研究室]

1.　はじめに

オートマトンはシステムの構造や動作を抽象化した数理モデルの一つであり、状態遷移図による視覚的表現はその理解を助ける。しかし、状態遷移図はオートマトンを構成する要素が増えるにつれ複雑さが増し、視覚的に整理された形にするには多くの時間と労力を要する。この問題はオートマトンについて学習する教育現場において、教員の資料作成や学生の学習効率に悪影響を及ぼす可能性がある。本研究ではこの問題を解決するために、オートマトンの状態遷移図を自動的に描画し、編集やシミュレーションを可能にするシステムの開発を行う。

2.　準備

2.1　Graphviz

Graphviz（Graph Visualization Software）[1]は AT&T 研究所が開発したオープンソースのグラフ視覚化ソフトウェアである。本研究ではオートマトンの状態遷移図を自動的にレイアウト計算するために Graphviz を使用する。

2.2　PyQt

PyQt[2]は GUI ツールキットである Qt を Python から使用できるようにするバインディングである。本研究では PyQt6 を用いて GUI の構築とインタラクティブに対応した状態遷移図の描画を行う。

3.　本研究の概要

本システムは、オートマトンの作成における負担を軽減するために以下の 4 つの機能の実装を最終目標としている。

1. 状態遷移図の自動描画
2. HID（Human Interface Device）による編集機能
3. 状態遷移図の保存および呼び出し
4. シミュレーション機能

本研究では決定性有限オートマトン（Deterministic Finite Automaton、以下 DFA と略記）を対象とした自動描画機能を実装する。なお、将来的には非決定性有限オートマトン（Nondeterministic Finite Automaton）や他のオートマトンへの対応も視野に入れている。

本システムの開発には Python を用い、状態遷移のレイアウト計算には Graphviz を、ユーザインタフェースの構築には PyQt を採用する。

4.　実装の概要

本章では、本研究で実装したシステムのインタフェースと状態遷移図の自動描画機能について述べる。

4.1　インタフェース

図 1 にオートマトン作成システムのインタフェースを示す。オートマトンを構成する要素である状態や遷移関数等の定義を行えるようになっており、入力パネル下部に配置された描画ボタンを押すことで、ウィンドウ左側のエリアに状態遷移図が自動的に描画される。また、ウィンドウ左上にはメニューバーを設置し、状態遷移図の保存および呼び出し機能の実装を進めており、作業の継続性を確保できるよう設計している。なお、シミュレーション機能およびそれに必要なウィジェット等の実装は現時点では未完了である。

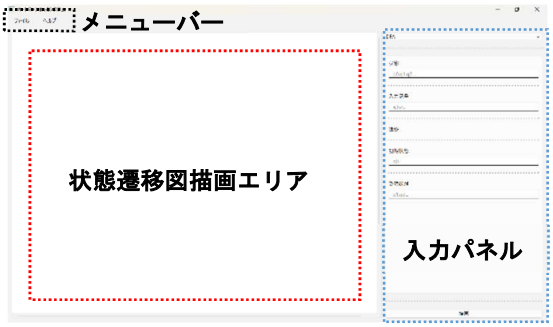

図 1.　メインウィンドウ

4.2　自動描画機能

自動描画機能はユーザが定義したオートマトンの状態遷移図を自動的に描画する機能である。ユーザから入力されたオートマトンの構成要素の情報は Graphviz のレンダリングエンジンによってレイアウト計算が行われ、レイアウト情報として出力される。その後、その情報を基に状態遷移図を描画する。

レイアウト情報は plain 形式で出力され、図 2 にその一部を示す。出力された情報では、状態は node（以下ノードと略記）、遷移は edge（以下エッジと略記）として扱われ、各行にノードやエッジのラベルや座標等が記述される。図 2 はノード q0 からノード q1 に向かうエッジついての情報であり、遷移先のノードやエッジの経路を示す座標が記されている。なお、Graphviz と PyQt の座標系には単位や座標系の原点の位置等に相違があるため、PyQt の座標系に変換する。

```
edge q0 q1 4 2.3453 0.33515 2.4719 0.33515
```

図 2.　出力されたオートマトンのレイアウト情報の一部

図 3 は実際に描画された DFA= (Q,Σ,δ,q_0,F)であり、DFA の構成要素は以下ように定義した。

$Q=\{q_0,q_1,q_2\},\ \Sigma=\{a,b\}$

$\delta:\ \delta(q_0,a)=q_0,\delta(q_0,b)=q_1,\delta(q_1,a)=q_0,\delta(q_1,b)=q_2$
$\delta(q_2,a)=q_1,\delta(q_2,b)=q_0$

$q_0=q_0,\ F=\{q_2\}$

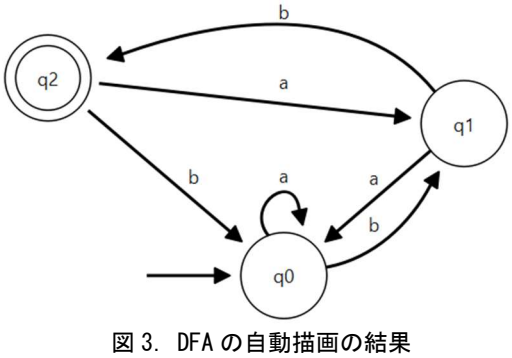

図 3.　DFA の自動描画の結果

ノードやエッジのラベルはレイアウト情報に基づいて描画されている。一方で、エッジの曲線については視覚的に見やすい状態遷移図を実現するために、ベジェ曲線を求める計算を行い、滑らかな曲線を生成している。ベジェ曲線とは$N+1$個の制御点から得られるN次曲線であり、以下の(1)で表される。

$$P(t) = (1-t)^3P_0 + 3t(1-t)^2P_1 + 3t^2(1-t)P_2 + t^3P_3 \ (1)$$

ここで$t \in \mathbb{R}, 0 \leq t \leq 1$はパラメータ、$\{P_i | i \in \{0,1,\dots,N\}\}$は制御点である。PyQt には制御点を指定することでベジェ曲線を描くメソッド用意されている。しかしレイアウト情報には曲線が通過する複数の座標点が記録されているものの、ベジェ曲線を描画するための制御点の情報は含まれていない。このため、本システムではレイアウト情報に含まれる座標点から制御点を推定するアルゴリズムを実装した。図 4 は、始点と終点以外の制御点およびレイアウト情報に記録されていた座標点を可視化した状態遷移図を示している。この図では、制御点を緑、座標点を青で表示しており、曲線がどのように生成されているかを示している。

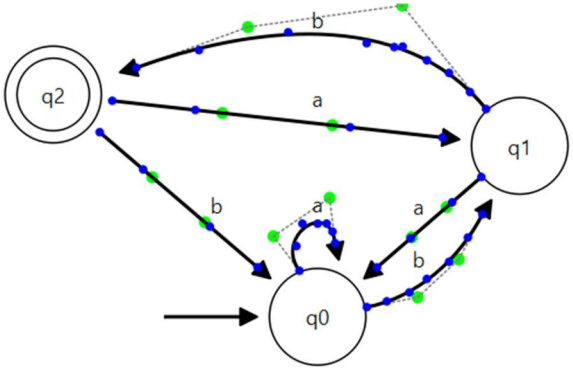

図 4. 制御点(緑)と座標点(青)を可視化した DFA

描画された曲線と座標点との誤差が小さく、レイアウト情報に基づき、全体として適切な曲線を描画できている。以上の描画結果より、自動描画機能では DFA の基本構造を正確に反映しつつ、視覚的にも高い状態遷移図を描画できている。

5. 課題と今後の展望

本研究では DFA の自動描画機能の実装を行うことができた。この機能により、状態や遷移を手動で配置する必要がなくなり、ラベル一つ一つを入力する手間も省かれることで、オートマトンの作成にかかる時間を大幅に短縮することができた。このことにより教員の資料作成にかかる負担の軽減や学生の学習効率の向上が見込まれる。しかしながら、改善が必要な点や未実装の機能が多く残されており、さらなるシステムに拡張が求められる。

まず、自動描画機能の課題について述べる。この機能ではオートマトンの構成要素を手動で入力する必要があるが、オートマトンが複雑化するにつれ入力作業にかかる時間と労力が増大し、その過程で入力ミスが発生するリスクも高まる。特に、本システムでは遷移関数を定義する際に図 5 に示す通り各状態に対して入力欄が設けられ、そこに「入力記号:状態」のペアをカンマで区切って入力する形式を採用している。しかし、この形式では状態や入力記号が増加するにつれて必要な入力数が増え、煩雑さが課題となる。これを解決するために、定義作業をより直感的かつ効率的に行えるインタフェースの導入を計画している。具体的には、キーボード入力による定義は状態と遷移のみにし、チェックボックスやプルダウンメニューなどを用いた選択式フォームに変更する予定である。このインタフェースを導入することで、入力ミスを減らし、オートマトンの作成に要する時間をさらに短縮できると考えられる。

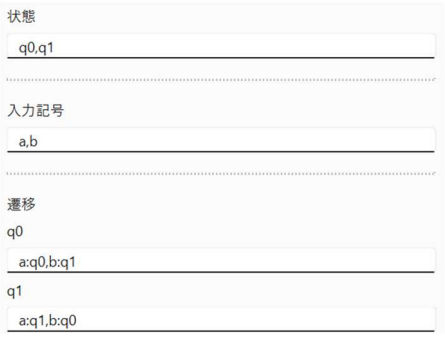

図 5. 状態、入力記号、遷移の入力欄

次に未実装機能の実装である。現状ではオートマトンの作成自体は可能なものの、手動での状態や遷移の配置変更、追加・削除する機能は実装されていない。HID による編集機能はオートマトンの作成プロセスを柔軟にするための重要な機能であり、有用性の向上において不可欠である。この機能が実装することで、オートマトンの作成や編集がより直感的かつ効率的に行えるようになり、システムの実用性が大幅に向上すると期待される。

また、シミュレーション機能についても現時点では未実装である。この機能は、オートマトンの作成・編集に比べると実装の優先度は低いものの、オートマトンの動作を視覚的かつ直感的に確認できるようにする点で非常に重要であると考えている。特に、作成したオートマトンの動作確認を行う手段として、シミュレーション機能は有用であり、この機能が実装されることで教員は授業準備の際に教材の動作確認を簡単に行えるようになり、教材の品質向上が期待される。また、学生にとっては、作成したオートマトンの動作を視覚的に追うことで、抽象的な概念を具体的に理解する助けとなる。

6. おわりに

本研究ではオートマトンの状態遷移図を自動描画するシステムを開発した。このことにより、教員の負担軽減や学生の理解促進が期待できる。しかしながら、未実装の機能が課題として残る。これらの機能拡張を進めていくことにより、教育現場での有用性が更に高くなると考えている。

参考文献

[1] Graphviz Development Team. “Graphviz”. Graph Visualization Software. 2021-08-10.https://graphviz.org/, (参照 2024-12-10)

[2] Riverbank Computing. “PyQt Components”. Riverbank Computing | Introduction. https://riverbankcomputing.com/software/pyqt/intro, (参照 2024-12-10)