部分グラフ同型アルゴリズムの実装と回路理解支援システムへの応用

秋山　耀[†1,a)]

概要：電子工作を行う際，web や本を用いることで目的の動きをする回路を探し出すことが可能である．しかし探しだした回路が「どういう動きをするのか」は理解できても「どうしてこの構造になっているのか」を理解するのは難しい．そこで回路図から回路の構造を解析し，ユーザに分かりやすく提示するシステムを提案する．回路中の決まった回路パターンを探し出し，それらを図示や文章で説明する．さらにそれらを矢印でつなげフローチャート化する．システムを実装するに従い，グラフの部分同型問題を解くために Ullmann のアルゴリズムを使用した．

1. はじめに

1.1 背景

電子工作を行う際，まずユーザは作りたい回路を本または web で検索する．そして本や web サイトに書かれている回路図，またはそれをより実際の配線に近い形で記した実体配線図を見ながら回路を作成する．このように「何がつくりたいのか」が自分の中で決まってさえすれば，回路図を探し出し，回路を作成することができる．しかし，「なぜその回路がそのような構成になっているのか」を理解しないで作成しているユーザは少なくないだろう．アナログ回路の場合は，デジタル回路に比べ特に回路の構成が難しいため，よりそのような傾向が強いように思われる．

そこで，本論では回路を学習するためのシステムではなく，回路を作成するときに，回路の動きを大雑把に把握するための手法を提案する．回路を役割ごとにユニットにまとめ，それらを何らかの形で簡易化して表示することでユーザの回路構成理解を支援する．今回はシステムの対象回路をギターのエフェクタ回路とし，回路は作れるが回路の意味を理解せず作成している人を対象ユーザとして設定する．さらに任意の回路図において提案手法を作成する部分を，グラフの部分同型問題を解くことに帰着させ，プロトタイプシステムを実装した．

1.2 目的

本論の目的は，回路をユニットにまとめて単純化する手法を用いてユーザの回路に対する理解を深めうるかの検証である．第 3 章で回路の可視化方法を提案し，第 4 章で提案した手法を用いた予備実験について述べる．第 5 章で，予備実験に基づいて改良した手法を用いたシステムについて述べる．第 6 章で議論及び制約について述べる．以降，これらの手法，評価，そして提案手法を用いたプロトタイプについて述べる．

2. 関連研究

電子回路について，学習支援を目的としたシステムが提案されている．例えば，豆電球を用いた並列，直列の回路を学習するためのシステムを高山らは提案している [2]．これは回路中における電圧を線の高さに，電流を線の太さとして表現して電圧・電流を可視化することで学習を支援しようとしている．また，順序回路について学習を深めるために，出口はフリップフロップの学習支援システムを開発している [3]．回路図だけでなく，タイミングチャートや状態遷移図，状態遷移表を状態ごとに同期させることにより学習効果を高めようとしている．これらの研究では回路を勉強する過程を支援している．しかし，ユーザが電子工作しようと web サイトや本に記された複雑な回路を読むときに使うことは難しい．

仮想上の回路をシミュレートする研究もされている．例えば，123D Circuits[4] はソフトウェア上にブレッドボードを使用した回路を製作，シミュレーションを行える．CCK[5] はブレッドボード上ではなく，自由に回路を製作して試行錯誤できる．また，123D Circuits では回路を作った後にそれをプリント基板などにおこすことが可能である．実際の Arduino と連携できるシステムとして Wakita

[†1] 現在，明治大学
Presently with Meiji Uniersity
[a)] ak@cs.meiji.ac.jp

らの Intuino[6] がある．Intuino はタイムライン上に出力値を描くシーケンサ的なインタフェースによって Arduino の動きの調整を容易にしている．

また，ソフトウェアシミュレータの中だけでなく，物理的なブロックオブジェクトを用いて電子工作を支援しようという研究も存在する．LightUp は電子部品のセットされたブロックを磁石でつなげることにより回路を作成できるシステムである [7]．回路をスマートフォンなどのカメラを通して見ることで，電子の流れを可視化する事ができる．このシステムであれば回路を作成する過程に置いてどのように電気が流れるかを把握することが可能である．システムはブロックと電子部品の対応関係における透明性を確保するために，回路ダイアグラムを参考にシステムのデザインをしている．回路ダイアグラムは回路をわかりやすくするための手段として使用される．回路を機能毎に切り分けてブロックで表現する．ブロック毎の相関関係を矢印で示す回路ダイアグラムもある．littleBits は LightUp と同じように電子回路をブロックを用いて作成できるシステムである [8]．LightUp と違い，ブロック中に複数の電子部品や回路が含まれ，ブロック毎に 1 つの機能を持つ．そのため，littleBits を用いて作られるのは回路ダイアグラムそのものといえる．

回路理解のための研究は電気の理解，または回路構築の容易さのどちらを支援しているかで分けることができる．例えば Conradi の Flow of Electron[9] は前者として挙げられる．Flow of Electron はドラッグ操作で仮想的に配線できるテーブルトップインタフェースであるが，チュートリアル機能や電子の動きのシミュレーション機能を持つ．また，先述の LightUp もこちらに分類されるだろう．逆に，先述の littleBits は回路を理解することではなく，とりあえず動く回路が作れるようになることを重視している．落合らの VisibleBread board は，回路に流れている電気をリアルタイムで AR 的に可視化することで，新たな発見と感動を共有し電気回路を触ることの喜びを与えることを目的としている [10]．本研究はどちらかと言えば前者にあたるだろう．

今回は回路ダイアグラムのように回路を機能毎に自動でまとめ，それをユーザに分かりやすく提示することを目的としている．

3. 回路図の可視化

多くの複雑な回路は，機能が単純な小さい回路のブロック（回路ブロック）に分けて考えることが可能である．回路が読める老練者は複雑な回路を回路ブロックに分けて回路を読み解いている [1]．そこで，これら老練者が行っている作業をコンピュータが行うことで，初学者の回路理解支援を行うシステムを提案する．読みたい対象の回路（クエリ回路）の中に存在する，回路ブロックのパターン（回路パターン）を見つけ出す．そしてユーザが回路パターンを理解しやすいように提示する．今回，このように回路中をいくつかの回路ブロックにまとめて単純化することを「回路の可視化」と呼ぶことにする．

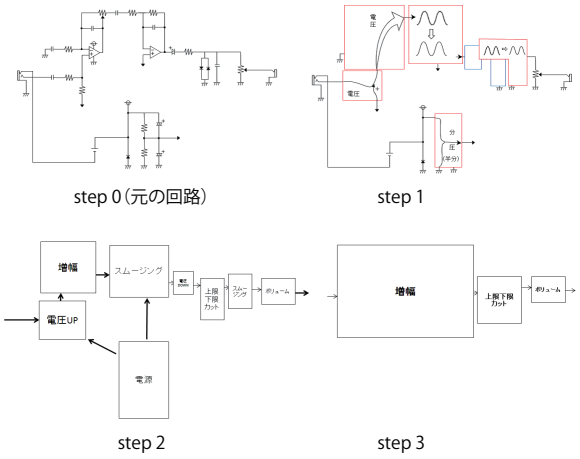

図 1 回路の可視化

可視化は 3 つの種類を提案し，それぞれを 1 つの step として 1 から 3 まで順に見ていく方式を提案する．また，「可視化が進むごとに元の回路の形から解離することになるが，単純で理解しやすくなる」という仮説を立てた．

step1 回路図中から回路ブロックを枠で囲い，図で説明

step2 回路図中から回路ブロックを枠で囲い，単語で説明したフローチャートにまとめる

step3 タイプ 3 中の回路ブロックを更に大きくまとめ，根幹だけ残して単語で説明したフローチャートにまとめる

3.1 step1

クエリ回路内の回路ブロックを枠で囲い，その回路パターンがどのような回路なのかをイラストで表現し，回路図上に表示する．例として図 2 はノイズ除去の回路パターンを表したイラストである．

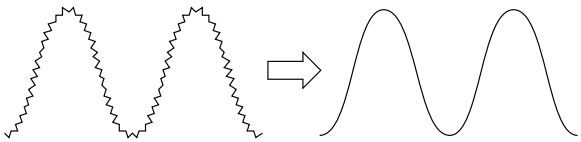

図 2 提示方法イラスト例

3.2 step2

クエリ回路内の回路ブロックを枠で囲い，その回路パターンがどのような回路なのかを単語で表現する．そして回路ブロックがどのように関係しあっているかをフローチャートとして表示する．この時，回路ブロックの位置は元の回路図上にあった位置からずらさない．

3.3 step3

step2 の回路ブロック群を，さらに役割に合わせてユニットにまとめ，それがどのような役割をするユニットなのかを単語で表現する．そしてユニットがどのように関係しあっているかをフローチャートとして表示する．この時，ユニットの位置は元の回路図上にあった位置からできるだけずらさない．

4. 予備実験

第 3 章で述べた提案手法が適した方法であるか，また第 3 章で述べた仮説を検証するため，男 6 人を対象に予備実験を行った．予備実験にはギターのエフェクター回路，ディストーションとフェイザーの 2 つの回路を用いた．ディストーションもフェイザーも音を歪ませるための回路である．これら 2 つの回路について各種可視化段階の画像を予備実験用に作成した．ラジオボタンで表示される画像が変わるような web ページを作成し，異なる step の画像を表示することで，可視化の段階をユーザが自由に変更できるようにした．前述の 3 つの可視化 step に加え，可視化前の回路図 step0 を含めた 4 つの step を用いて実験を行った（図 1）．step2 においては，ユニット同士が重なってしまう部分が見られたため，それぞれ上になるユニットが違う画像を作成し，それら 2 枚で 1 つの step として扱った．

4.1 タスク

実験参加者にはまず電子工作・楽器・シンセサイザー経験の有無を聞くアンケートを行った．参加者それぞれに回路の素子についての最低限の知識を書いたプリントを配った．その後で，予備実験システムを使って step1 から step4 までの画像を順に見てもらう．回路についてはそれがなんの回路であるか，どんな動きをする回路であるかを明かない．回路名もディストーション回路を回路 A として説明し，フェイザー回路を回路 B として名前を伏せて説明した．回路 A，回路 B それぞれについて以下のことを答えさせた．

- どの step で回路を理解できたか．ここで言う「理解」とは，回路のすべてを知り尽くすことではなく回路がどのような働きをするかを大雑把にわかることを指す．
- 理解するのにどのぐらい時間がかかったか．
- この回路について何かわかったことはあるか．
- この回路について詳しく知りたくなったか．

また，回路 A と回路 B について，2 つの回路の共通点と相違点についてもどの機能が共通しているのか，相違しているのか 2 つ挙げさせ，その共通点・相違点に気づいた可視化 step も答えさせた．タスク終了後に実験の所感を尋ねるアンケートを行った．

4.2 結果

参加者のうち楽器経験者は 3 人（うち 1 人がギター経験者），4 人は電子工作の経験があった（うち 1 人はギターのエフェクタ作成経験者）が，残りの 2 人はほぼ電子工作に触れたことがない初学者であった (表 1)．回路の理解した step について，人数の分布は表 2 のような結果になった．step1 で回路 A, 回路 B を理解した参加者 A は同一人物であり，ギターエフェクタ作成経験者であった．回路 A で全部見てもわからなかったと回答した参加者 B は「step1,step2 で用いていた単語の意味がわからなかった」と意見を述べている．

表 1 参加者ごとの各回路の理解した step

参加者	楽器経験	電子工作経験	回路 A を理解した段階	回路 B を理解した段階
A	あり	あり	step1	step1
B	なし	なし	わからなかった	step3
C	あり	あり	step2	わからなかった
D	なし	なし	step2	step2
E	なし	あり	step2	step2
F	なし	あり	step2	step2

表 2 step ごとの各回路の理解人数

	step0	step1	step2	step3	全部見てもわからなかった
回路 A	0	1	4	0	1
回路 B	0	1	3	1	1

2 つの回路の類似点・相違点を見つけるタスクにおいて，類似点は step1 と step2 を見て全員が正解をしている．相違点については，答えられなかった参加者 B と，ミスをした参加者 C の 2 人以外は全員正解をしていた．参加者 C は回路 A の step2 と回路 B の step3 を見て相違点を探したため，違う step 同士を見て答えたためのミスだったと考えられる．参加者 B については類似点は答えられていたため，実験中に心が折れた可能性が考えられる．

step1 の所感についてはイラストの抽象化がわかりづらくアニメーションを使って説明した方が良いという意見があった．step2 においてはわかりやすいという意見が多数であったが，フローチャートにすると回路という実感がなく感じたという意見もあった．ギターエフェクタ作成経験者である参加者 A は，波形（イラスト）での説明がなくなったのがわかりづらかったと意見している．step3 については簡略化しすぎててわからないという意見が大多数を占めていた．また，実験に対して，「どういう作用をするのかは理解できたが，何のためにその作用をするのかがわからない」という意見があった．

4.3 手法の改良

実験結果を受けて，step1 についてはイラストによる抽象化をよりわかりやすいものに変更する必要があると考え，アニメーションを付加したものに変更する方針にした．step2 に対しての意見に単語がわかりづらいというものがあったため，step2 に用いる説明を単語ではなく，文という形にする．また，フローチャートをどこから見て良いのかわからなかった可能性があるため，フローチャートを整列して表示するという改良を行う（図 3）．step0 では誰も回路を理解することができていなかったが，step1 と step2 で半数以上の参加者が理解できていたため，step1・step2 によって参加者の回路理解を支援できたと考えられる．また，前述した「step が進むごとに元の回路の形から解離することになるが，単純で理解しやすくなる」という仮説は step1 より step2 の方が理解した参加者が多かったため，step2 までは成り立っていたと考えられる．step3 は大幅な簡易化は逆に使用者に混乱を招くことがわかったため，step3 は削除する方針とする．

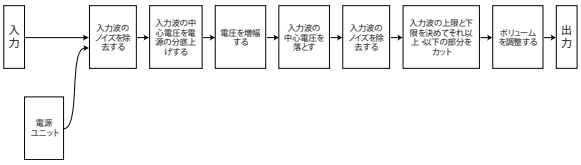

図 3　step2 の改良案

5. 実装とアルゴリズム

回路図をコンピュータ上で扱えるようにするために，回路図をグラフ構造に変換する（図 4）．今回は素子と交点をノードとし，それらの接続関係をエッジとしたグラフ構造に変換した．ノードには id が割り当てられており，id は素子の種類を表している．交点もノードとして表現することで，ノードの次数は，その素子の使っている端子数として表現することが可能になる．

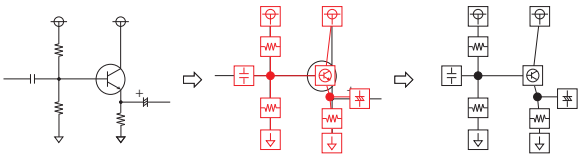

図 4　回路のグラフ化

回路図をグラフ構造に変換して考えることで，クエリ回路から回路パターンを見つけ出す問題は，クエリ回路を変換したグラフ（クエリ回路グラフ）の部分グラフのうち，回路パターンを変換したグラフ（回路パターングラフ）と同型であるものを探す問題（グラフの部分同型問題）に置き換えて考える事が可能である．グラフの部分同型問題は，グラフ B がグラフ A と同型な部分グラフを持つかどうかという判定問題と定義されている（図 5）．グラフ A と B が同型であるとは，A と B が同じラベルの頂点を持ち，同じような辺のつながり方をすることを挿す．ここで言うグラフ A とは回路パターングラフであり，グラフ B とはクエリ回路グラフである．回路パターンは多数あるため，その数だけグラフの同型判定をする必要がある．

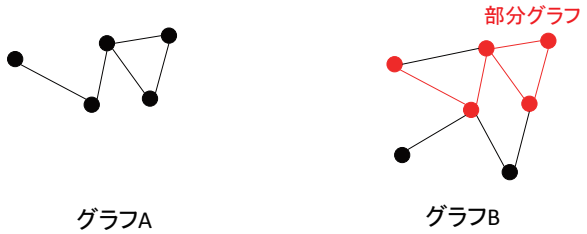

図 5　グラフの部分同型問題

5.1 バックトラックの部分同型問題アルゴリズム

部分同型問題では，グラフ A とグラフ B の頂点の対応付けを，探索木を用いて決めることができる（図 6）．例えば，図上の青線で表されたルートではグラフ A の頂点 1 とグラフ B の頂点 1，グラフ A の頂点 2 とグラフ B の頂点 4，グラフ A の頂点 3 とグラフ B の頂点 3 を対応付けている．探索時間はグラフ A の超点数を n，グラフ B の頂点数を m とした時，$\Updownarrow\backslash$ であるため，指数時間アルゴリズムとなる．

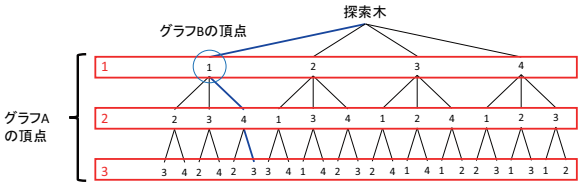

図 6　探索木

5.2 Ullmann の部分同型問題アルゴリズム

グラフの部分同型問題は NP 完全であるため，多項式時間では答えが求まらない可能性がある．そこで，Ullmann のアルゴリズム [11] を用いて，グラフの部分同型問題を回路に適応した際に実用時間内に解けるか簡易実験を行った．Ullmann のアルゴリズムはグラフの部分同型問題の有名な改良アルゴリズムである．探索木を探索する際に常に同型可能性（Refinement Procedure）を計算し，探索空間の削減を行う．

グラフ A は回路パターングラフより，ノード数は回路パターンの大きさに依存する．回路パターン自体はそれほど大きい回路ではないため，ここではグラフ A を 20 頂点として固定した．それに対し，グラフ B はクエリ回路グラフであるため，ノード数はユーザの調べたい回路に依存する．

そこで今回の簡易実験ではグラフ A のノード数を固定し，グラフ B のノード数を 20 から 200 まで 10 ノードずつ増やして実行時間を調べた（図 7，表 3）．グラフのノードに id が振ってある場合のグラフ部分同型判定のために Ullmann のアルゴリズムを変更したものを用いて実験を行った．また，頂点 id の種類は回路中の素子の種類に依存するため，0〜7 の 8 種類としてグラフを作成した．実験に用いたグラフはほぼ部分同型判定の false 例である．グラフ B の同じノード数に対してそれぞれ 20 回施行を行い，その平均値をプロットしている．使用言語は Java である．

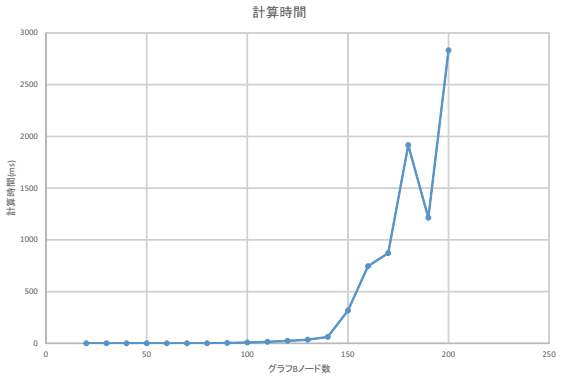

図 7　簡易実験結果

実験結果より，約 100 頂点までなら 0.01 秒以内，約 140 頂点までなら約 0.1 秒以内に答えが出ると判明した．パターン回路が 100 個存在すれば，それらの大体 100 倍の時間がかかる．並列化して処理を行えばそれらの何分の 1 程度に計算時間を抑えることが可能である．パターン予備実験で用いたディストーション回路とフェイザー回路をグラフ構造に変換した場合，それぞれノード数は約 45 頂点，約 100 頂点であり，比較的小規模な回路であれば Ullmann のアルゴリズムを使用しても問題は無いと考えられる．

5.3　step2 の実装

回路をフローチャートにする際に，回路グラフの始点を設定する必要がある．回路は左から右にデータが流れていくように書かれることから，今回は一番左にある端子（グラフのノード）を回路グラフの始点ノードを設定している．そして回路グラフの始点ノードからの距離で各ノードの流れを順位付けしていく．

5.4　制約

回路をグラフに変換する際につながっている端子や極性を区別していないため，ダイオードなどが逆になっていても回路パターンにマッチングしてしまう場合がある．これは生成するグラフを有向グラフにすることで対応ができるのではないかと考えている．

回路パターンは登録されているものしか対応できない．

表 3　バックトラックアルゴリズムと Ullmann のアルゴリズムの比較

ノード数	バックトラックアルゴリズム	Ullmann のアルゴリズム
20	0ms	0ms
25	0ms	—
30	51ms	0ms
31	215ms	—
32	1289ms	—
33	17219ms	—
34	9621ms	—
35	92598ms	—
40	—	0ms
50	—	0ms
80	—	0ms
90	—	4ms
100	—	8ms
110	—	15ms
120	—	24ms
130	—	35ms
140	—	62ms
150	—	317ms
160	—	746ms
170	—	872ms
180	—	1916ms
190	—	1214ms
200	—	2832ms

例えば，回路パターンを改良した形もすべて登録する必要がある．

また，現時点では回路パターン中に回路パターンが含まれる場合に両方ヒットしてしまう場合がある．これは回路パターン同士の部分同型問題を前もって計算しておくことで小さい方の回路パターンを弾くことが可能である．

また，今回のシステムは，初心者がパターンマッチングを行う上で便利な，回路をブロック分けする 3 つのルール [1] のうち 1 つ（複数ブロックで同じ回路を共有することがある）を満たしているが，残りの 2 つ（インピーダンスの変わる箇所を見つけ出す，帰還ループを分割しない）を考慮しないシステムとなっているため，それらのルールを守れない箇所が出てくる場合がある．これらもグラフの扱いを工夫し対策する必要があると考えている．

5.5　プロトタイプシステム

作成したプロトタイプシステムは以下の図のようになっている（図 8）．各 step は左右キーで行き来することが可能である．step1 はアニメーションとなっている．また，0〜2 までのキーを入力することで，好きな可視化段階を重ねて表示することも可能である．1，2 キーはそれぞれ可視化段階 step1，2 に対応する．0 キーは元々の回路を表示することが可能である．可視化段階 step3 は step2 までと違

い，ユニットの位置情報を無視して整形しているため，今回は重ねて表示できないようにしている．

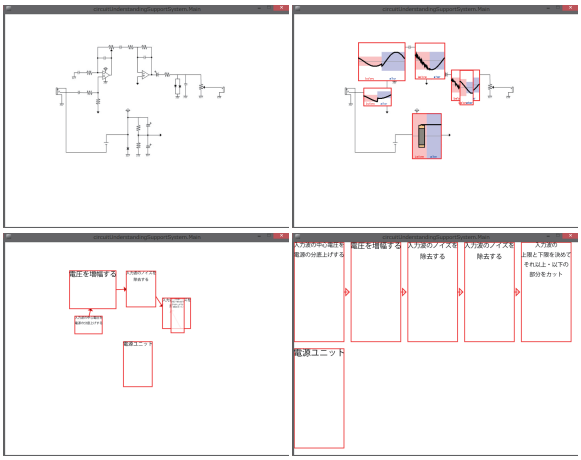

図 8　プロトタイプシステム

6. 議論・制約

本提案手法では，素子の挙動そのものまで理解できることが目的ではなく，あくまで回路全体として電気がどのような流れで処理されているのかを理解できるようにするのが目的である．そのため，提案手法をもちいて可視化した回路を見て「素子がどのように動いているか気になった」とユーザが素子レベルでの働きに興味を抱けば提案手法は成功といえる．第 4.3 章で出た意見より提案手法のみで回路全体が何なのかを理解するのは難しく，提案手法は何の回路かわかっているものについて使用すべきである．これらの理由により提案手法はトップダウン式に電子回路学習および電子回路理解を行う際の手法となるのではないだろうか．

多くの回路理解支援システムは電流や電圧の振る舞い，そして素子がどのように動くかなど，回路を通して電気を勉強することを目的としている．そのため，単純な回路を対象とするにとどまっている．電子工作をするにあたって，ユーザが作るのは web サイトや本に記された複雑な回路であり，教科書などに書いてある基礎的で単純な回路ではない．もちろん，これら回路の理解支援システムを用いて好きな回路を解析させることは可能である．しかしシステムを通して理解できるのは素子自体の働き及び回路中の電圧電流など細かい事象である．複雑な回路を理解するにあたって必要とされているのは，回路中のすべての導線の電圧を把握することではなく，回路を大雑把に分けてそれらの機能が分かることではないだろうか．

第 5 章で述べた回路のグラフ化では，電子部品の向きは考慮していなかった．よって違う回路でも同じ回路グラフに変換されてしまう可能性がある．また，回路図においてグラウンド（0v）や電源は，電子部品に電源を通すのは自明なものとして省略されることがある．この場合，省略の有無で生成される回路グラフが変わる可能性がある．また，フローチャート化に際して，回路図が信号の流れが左から右になるように記述されていなければ適切にフローチャートに直すことは出来ないという制約がある．step1 と step2 において，ユニット同士重なって表示される場合がある．これは現在，マウスオーバーしているユニットを上に表示することで対応している．しかし，重なった場合のフローチャート化が現在難しく，表示が崩れてしまう可能性がある．

7. まとめ

本論では回路の理解支援手法としていくつかの可視化手法をあげ，それらが理解支援につながるか予備実験を行った．予備実験の結果を踏まえ，可視化手法を改良を行い，プロトタイプシステムを作成した．これにより回路グラフから自動でユニットを解析し提示することに成功した．またいくつかの制約を述べ，議論を行った．

参考文献

[1] 鈴木雅臣. 回路の素 101. CQ 出版, pp.2-29, 2012.

[2] 高山透, 平井佑樹, 金子敬一. 対話型回路作成による電流・電圧・抵抗の関係学習システムの提案. 情報教育シンポジウム 2013 論文集, pp.19-26, 2013.

[3] 出口幸子. フリップフロップの学習支援システムの開発と利用. 日本教育工学会論文誌, Vol.38, No.1, pp.73-77, 2014.

[4] 123D Circuits　https://123d.circuits.io/

[5] Wieman, C.E. and Perkins, K. K. A powerful tool for teaching science. Nature Physics, 2, 5, pp.290-292, 2006.

[6] Wakita, A. and Anezaki, Y. Intuino: An Authoring Tool for Supporting the Prototyping of Organic Interfaces. In *Proc. of DIS '10* (the 8th ACM Conference on Designing Interactive Systems), pp.179-188, 2010.

[7] Chan, J., Pondicherry, T., and Blikstein, P. LightUp: an augmented, learning platform for electronics. In *Proc. of IDC '13*, pp.491-494, 2013.

[8] Bdeir, A., and Rothman, P. Electronics as material: littleBits. In *Proc. of TEI '12*, pp.371-374, 2012.

[9] Conradi, B., Lerch, V., Hommer, M. et al. Flow of Electrons: An Augmented Workspace for Learning Physical Computing Experientially. In *Proc. of ITS '11*, pp.182-191, 2011.

[10] 落合陽一.「電気がみえる」デバイス VisibleBread board. 日本バーチャルリアリティ学会論文誌, Vol.15, No.3, pp.463-466, 2010.

[11] J. R. Ullmann. An Algorithm for Subgraph Isomorphism. J. of the ACM, Volume 23 Issue 1, pp.31-42, 1976.