2020年度 卒業論文

ネットワークトポロジ可視化システムの開発

大阪産業大学 デザイン工学部 情報システム学科
情報教育システム研究室

17H027 尾白航樹

ネットワークトポロジ可視化システムの開発

17H027 尾白航樹

1 はじめに

システムの構築・設計において、構成予定図や設計書と実際に運用されている構成が異なってしまうことは、時に深刻な問題を引き起こす。2020 年 10 月 1 日、株式会社東京証券取引所にて富士通株式会社製の株式売買システム “arrowhead”のシステム障害により終日売買停止 [1] となり、マニュアルに記載されていた構築予定の仕様書と実際に運用されていた機器の仕様が異なっていたことが原因として各メディアに取り上げられたことは記憶に新しい。このような失態を防ぐためには、システム設計書通りの構築が実際に行われているかを確認する手段が必要であると考えられる。

2 目的

本研究の目的は、構築予定の設計と実際の構成が異なってしまう問題の解決である。そこで本研究では、ネットワークのトポロジを構成している Cisco Systems, Inc. の “Cisco Catalyst”の設定情報から、可視化されたトポロジ表示システムを開発し、トポロジの正常および異常を容易に確認することを目指す。本研究で開発したシステムにより構築予定図との差異を発見することで、運用後のトラブル防止に繋がると考える。

3 NTG

NTG とは、“Network Topology Grapher”の略で、本研究で開発したシステムの名称である。NTG は、config ファイルから JSON ファイルを作成し、トポロジ表示ツール “inet-henge-master”[*1] [2] に JSON ファイルを読み込ませ、トポロジ表示をするまでの一連の動作を自動で行う。

[*1] JSON ファイルを読み込み、ローカルホスト上に自動でトポロジ表示を行うツールのこと。

4 検証

トポロジ表示の自動化された NTG を用いて、正常な状態のトポロジに対し何らかのトラブルを与え、表示されたトポロジから異常が検知できるのかを検証する。演習室のトポロジは現在運用中であり、トラブルを避けるために本研究では、大阪産業大学デザイン工学部情報システム学科の 3 年時に履修できるネットワーク構築演習で実際に扱った構成を “Cisco Packet Tracer”[*2]を用いて config を作成し、意図的にエラーを起こし検証を行う。本研究では、構築時のエラーとしてポートのシャットダウンの検証を行った。

5 まとめ

本研究ではネットワークのトポロジを構成している “Cisco Catalyst”機器の設定情報から、トポロジの表示を行うシステム開発を行った。その結果、“running-config”と “cdp neighbors”の 2 つの設定情報から物理的なトポロジを可視化させることに成功し、ポートや物理層の異常によるトポロジの欠落を発見することができた。今後の課題としては、マルチベンダー間でのトポロジの表示、論理的なトポロジの表示、VLAN 等の情報の追加が挙げられる。

参考文献

[1] 10 月 1 日に株式売買システムで発生した障害について. https://www.jpx.co.jp/corporate/news/news-releases/0060/20201019-01.html, Oct 2020.

[2] inet-henge-master. https://github.com/codeout/inet-henge/, 2020.

[*2] Cisco Networking Academy が公開しているフリーのソフトウェアのこと。ソフトウェア上で PC・ルータ・L2 スイッチ・L3 スイッチなどを配置し、それらを接続してネットワークを構築することができる。

目次

1　はじめに

システムの構築・設計において、構成予定図や設計書と実際に運用されている構成が異なってしまうことは、時に深刻な問題を引き起こす。2020 年 10 月 1 日、株式会社東京証券取引所にて富士通株式会社製の株式売買システム “arrowhead”のシステム障害により終日売買停止 [1] となり、マニュアルに記載されていた構築予定の仕様書と実際に運用されていた機器の仕様が異なっていたことが原因として各メディアに取り上げられたことは記憶に新しい。このような失態を防ぐためには、システム設計書通りの構築が実際に行われているかを確認する手段が必要であると考えられる。

第 2 章では本研究の目的について述べる。第 3 章では本研究で用いる機器と既存のツールの概要を述べる。第 4 章では本研究で開発したトポロジ可視化システムの概要を述べる。第 5 章では本研究による検証について述べる。第 6 章では検証結果および本研究に関わる振り返りをその考察とともに述べる。第 7 章には研究の成果とともに今後の課題についてまとめる。

2 目的

本研究の目的は、構築予定の設計と実際の構成が異なってしまう問題の解決である。そこで本研究では、ネットワークのトポロジを構成している Cisco Systems, Inc.(以下、Cisco 社とする) の “Cisco Catalyst”の設定情報から、トポロジ表示を自動で行うシステムを開発し、トポロジの正常および異常を容易に確認することを目指す。本研究で開発したシステムにより構築予定図との差異を発見することで、運用後のトラブル防止に繋がると考える。

3 既存のシステム

本研究では、“Cisco Catalyst”の設定情報およびトポロジ表示ツール “inet-henge-master”を扱う。この章では、本研究で使用する “Cisco Catalyst”および “inet-henge-master”に関するコマンドやその実際の出力について説明を行う。

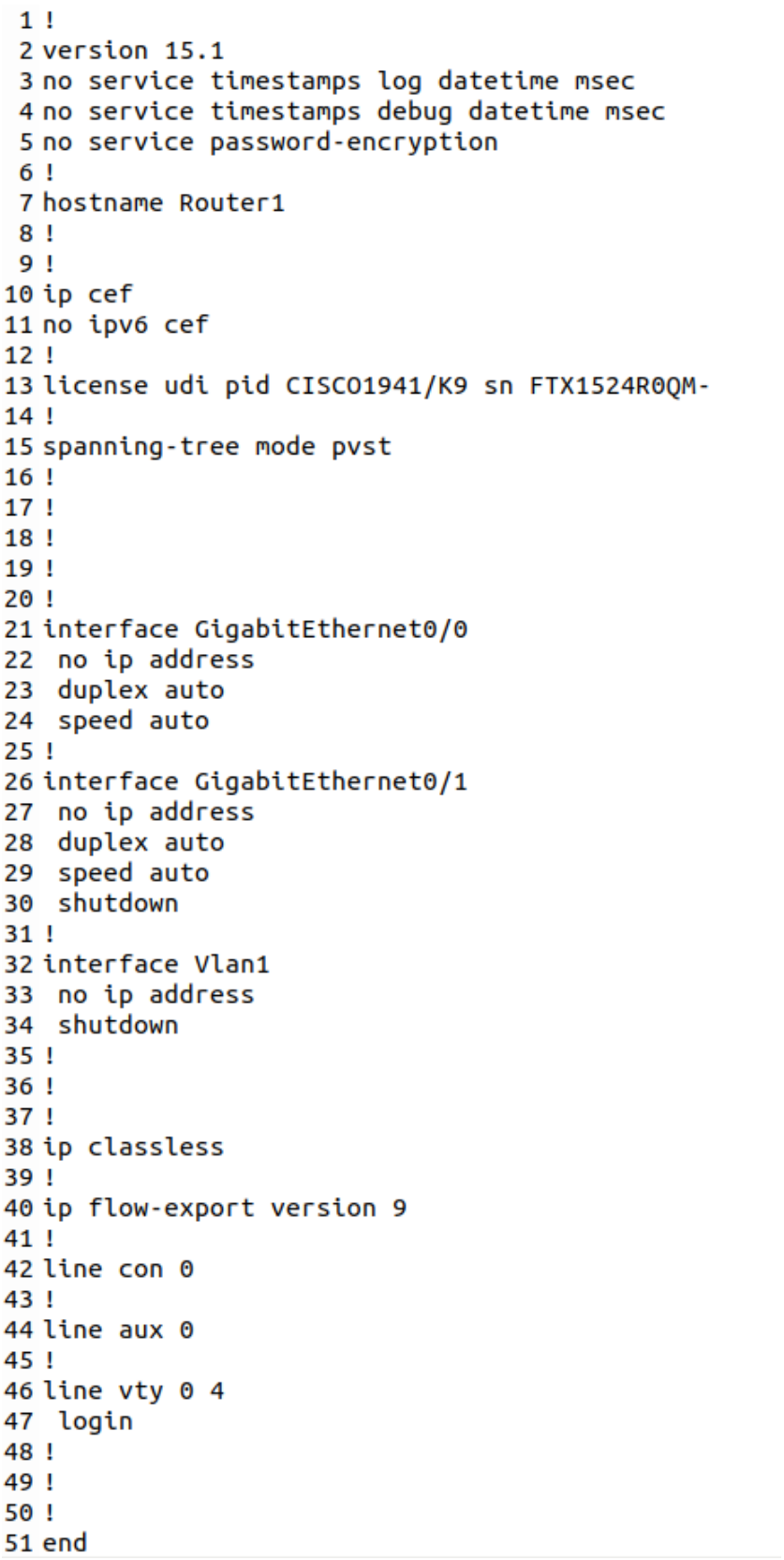
3.1 Cisco Catalyst

“Cisco Catalyst”とは、アメリカ合衆国カリフォルニア州サンノゼに本社を置く、世界最大のコンピュータネットワーク機器開発会社である Cisco 社製 LAN スイッチのフラッグシップブランドであり、ISP[*1]や大企業のコアネットワーク向けの大規模スイッチ、中小企業・部署内ネットワーク向けのスイッチ製品が含まれる。

3.1.1 running-config

“Cisco Catalyst”の設定情報の一つとして “running-config”がある。“running-config”は、動作中に使用するコンフィグレーションファイルのことであり、設定した内容は自動的に “running-config”へ保存される。これらの設定情報を参照する際は、コンソール画面の特権 EXEC モードから “show running-config”で呼び出すことができる。“show running-config”による出力画面の一例を図 1 に示す。

[*1] Internet Service Provider(インターネットサービスプロバイダ) のこと。インターネットサービスプロバイダとは、インターネット接続の電気通信役務を提供する組織のことである。



```
 1 !
 2 version 15.1
 3 no service timestamps log datetime msec
 4 no service timestamps debug datetime msec
 5 no service password-encryption
 6 !
 7 hostname Router1
 8 !
 9 !
10 ip cef
11 no ipv6 cef
12 !
13 license udi pid CISCO1941/K9 sn FTX1524R0QM-
14 !
15 spanning-tree mode pvst
16 !
17 !
18 !
19 !
20 !
21 interface GigabitEthernet0/0
22  no ip address
23  duplex auto
24  speed auto
25 !
26 interface GigabitEthernet0/1
27  no ip address
28  duplex auto
29  speed auto
30  shutdown
31 !
32 interface Vlan1
33  no ip address
34  shutdown
35 !
36 !
37 !
38 ip classless
39 !
40 ip flow-export version 9
41 !
42 line con 0
43 !
44 line aux 0
45 !
46 line vty 0 4
47  login
48 !
49 !
50 !
51 end
```

図 1　“running-config”の例。コンソール画面の特権 EXEC モードから “show running-config”で確認することができる。バージョンやインタフェース、VLAN の情報など、機器内で動作している設定情報を参照することができる。本研究では、7 行目に記載されている “hostname”の情報をノード情報の一つとして扱う。行頭の行番号は説明のために追加したものであり、実際には行番号は表示されない。

3.1.2 cdp neighbors

CDP(Cisco Discovery Protocol) は Cisco 社独自のプロトコルであり、データリンク層で動作する。このプロトコルは、ケーブル接続している隣接する Cisco 機器の情報が得られるもので、Cisco 機器のどこのポートにどのような Cisco 機器が接続しているのかが分かる。CDP はデータリンク層で動作することから、隣接機器の IP アドレス情報などが間違っていたり設定がない場合であっても第三層に依存しないため、問題なく隣接する機器の情報が得られる。こちらの情報もコンソール画面の特権 EXEC モードから “show cdp neighbors” で確認することができる。“show cdp neighbors” による出力画面の一例を図 2 に示す。出力される情報は左から順に、対向機器のホストネーム (Device ID)、出力元インタフェース (Local Intrfce)、ホールドタイム (Holdtme)、対向機器のネットワーク機能 (Capability)、対向機器の型番 (Platform)、出力先インタフェース (Port ID) の 6 つである。

```
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
                  D - Remote, C - CVTA, M - Two-port Mac Relay

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
FSW-15-8.osaka-sandai.ac.jp        Gig 1/0/36        152           S I   WS-C2960X Gig 1/0/5
hubmain0         Gig 1/0/38        151               S I   C9200L-48 Gig 1/0/47
hubmain0         Gig 1/0/39        171               S I   C9200L-48 Gig 1/0/48
hubmain0         Gig 1/0/37        138               S I   C9200L-48 Gig 1/0/46
hubmain0         Gig 1/0/41        130               S I   C9200L-48 Gig 2/0/47
hubmain0         Gig 1/0/40        129               S I   C9200L-48 Gig 2/0/46
hubmain0         Gig 1/0/42        173               S I   C9200L-48 Gig 2/0/48
hubmain04        Gig 1/0/48        156               S I   WS-C2960G Gig 0/48
hubmain04        Gig 1/0/47        156               S I   WS-C2960G Gig 0/47
hubmain04        Gig 1/0/46        156               S I   WS-C2960G Gig 0/46
hubmain03        Gig 1/0/45        123               S I   WS-C2960G Gig 0/48
hubmain03        Gig 1/0/44        137               S I   WS-C2960G Gig 0/47
hubmain03        Gig 1/0/43        123               S I   WS-C2960G Gig 0/46
```

図 2　“cdp neighbors” の例。コンソール画面の特権 EXEC モードから “show cdp neighbors” で確認することができる。“show cdp neighbors” により出力される情報は左から順に、対向機器のホストネーム (Device ID)、出力元インタフェース (Local Intrfce)、ホールドタイム (Holdtme)、対向機器のネットワーク機能 (Capability)、対向機器の型番 (Platform)、出力先インタフェース (Port ID) である。

3.1.3 config

一般に config とは、インターネットに関連する機器の設定のことを意味する。コンピュータ・サーバ・ネットワーク機器などのハードウェアから OS などのソフトウェアまでを含み、これらの機器の設定全般のことを config と呼ぶ。また本研究では、前述 (“running-config” “cdp neighbors”) の設定情報を一つの config として扱う。config の一例を図 3 に示す。

```
 1 !
 2 version 15.1
 3 no service timestamps log datetime msec
 4 no service timestamps debug datetime msec
 5 no service password-encryption
 6 !
 7 hostname Router1
 8 !
 9 ip cef
10 no ipv6 cef
11 !
12 !
13 license udi pid CISCO1941/K9 sn FTX1524R0QM-
14 !
15 !
16 spanning-tree mode pvst
17 !
18 !
19 interface GigabitEthernet0/0
20  no ip address
21  duplex auto
22  speed auto
23 !
24 interface GigabitEthernet0/1
25  no ip address
26  duplex auto
27  speed auto
28  shutdown
29 !
30 interface Vlan1
31  no ip address
32  shutdown
33 !
34 ip classless
35 !
36 ip flow-export version 9
37 !
38 !
39 line con 0
40 !
41 line aux 0
42 !
43 line vty 0 4
44  login
45 !
46 !
47 end
48 Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
49                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone
50 Device ID    Local Intrfce   Holdtme    Capability   Platform    Port ID
51 Switch1      Gig 0/0          157             S       2960        Fas 0/1
```

図 3　本研究で扱う config の例。上部に “running-config”、下部に “cdp neighbors” を連結させた設定情報となっている。47 行目までが “running-config”、48 行目以降が “cdp neighbors” の設定情報である。コンソール画面のログをそのまま貼り付けることで容易に作成することが可能。行頭の行番号は説明のために追加したものであり、実際には行番号は表示されない。

3.2 トポロジ表示ツール

本研究でトポロジを表示する際に、トポロジ表示ツール “inet-henge-master” [2] を利用する。“inet-henge-master” とは、JSON ファイルを読み込みローカルホスト上に自動でトポロジ表示を行うツールである。トポロジを “inet-henge-master” で表示したい場合、該当トポロジのノード情報、リンク情報の JSON ファイルを作成し、ツールに読み込みをさせる。トポロジを “inet-henge-master” で表示させるプロセス例として、元のトポロジを図 4 に、ノード・リンクの JSON ファイルを図 5 に、“inet-henge-master” による出力を図 6 にそれぞれ示す。

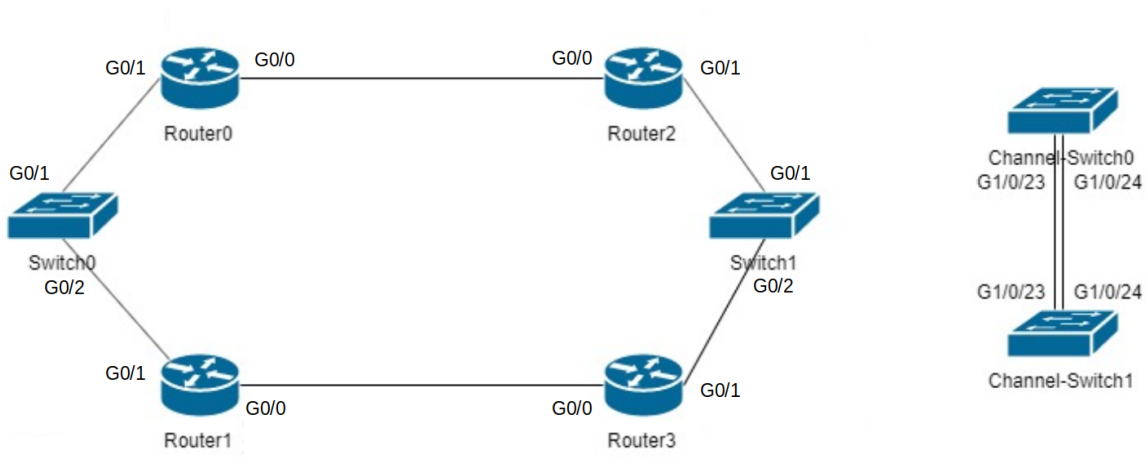


図 4　トポロジの例。左側のトポロジが 6 台でのリング型トポロジ。右側はスイッチ 2 台での直線的なトポロジである。“G0/0” や “G0/1” はポートを表している。

```
{
  "nodes": [
    { "name": "Router0", "icon": "./images/router.png" },
    { "name": "Router1", "icon": "./images/router.png" },
    { "name": "Router2", "icon": "./images/router.png" },
    { "name": "Router3", "icon": "./images/router.png" },
    { "name": "Switch0", "icon": "./images/switch.png" },
    { "name": "Switch1", "icon": "./images/switch.png" },
    { "name": "Channel-Switch0", "icon": "./images/switch.png" },
    { "name": "Channel-Switch1", "icon": "./images/switch.png" }
  ],

  "links": [
    { "source": "Router0", "target": "Switch0" ,
      "meta": { "interface": { "source": "ge0/1", "target": "ge0/1"} } },
    { "source": "Router1", "target": "Switch0" ,
      "meta": { "interface": { "source": "ge0/1", "target": "ge0/2"} } },
    { "source": "Router0", "target": "Router2" ,
      "meta": { "interface": { "source": "ge0/0", "target": "ge0/0"} } },
    { "source": "Router1", "target": "Router3" ,
      "meta": { "interface": { "source": "ge0/0", "target": "ge0/0"} } },
    { "source": "Router2", "target": "Switch1" ,
      "meta": { "interface": { "source": "ge0/1", "target": "ge0/1"} } },
    { "source": "Router3", "target": "Switch1" ,
      "meta": { "interface": { "source": "ge0/1", "target": "ge0/2"} } },
    { "source": "Channel-Switch0", "target": "Channel-Switch1",
      "meta": { "interface": { "source": "ge-1/0/23", "target": "ge-1/0/23"} } },
    { "source": "Channel-Switch0", "target": "Channel-Switch1",
      "meta": { "interface": { "source": "ge-1/0/24", "target": "ge-1/0/24"} } }
  ]
}
```

図 5　JSON ファイルの例。上部にノード情報、下部にリンク情報が記載されている。

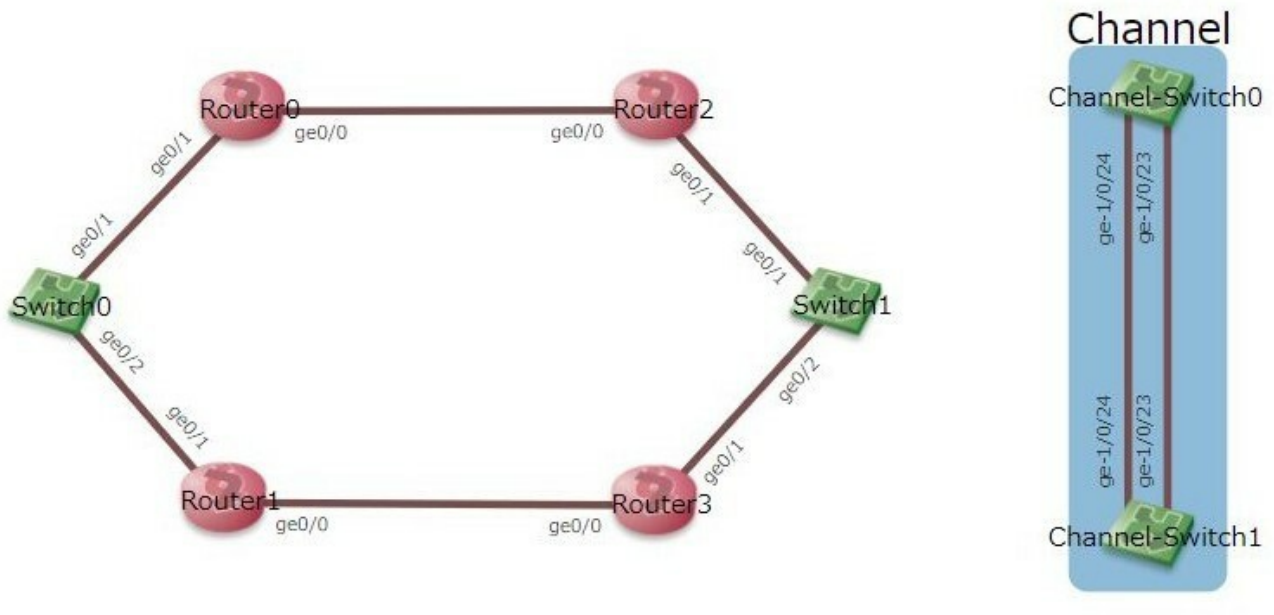


図 6　“inet-henge-master”によるトポロジ表示。図 4 と同じ構成のトポロジが表示されていることが確認できる。JSON ファイルに画像を参照させることで、ノードを好きな画像に置き換えることもできる。

3.3 既存システムの課題

“inet-henge-master”によるトポロジ表示を自動化するにあたり、JSON ファイルを手入力で作成しなければならないことが 1 番の課題として挙げられる。トポロジの情報が多くなると、前述図 5 のような JSON ファイルを手入力する際、膨大な時間と労力を費やすことになってしまう。現行の “inet-henge-master”使用時のプロセスを図 7 に示す。小規模なネットワークトポロジの表示に関しては、現行のプロセスのままで利用することが容易だが、大規模なネットワークトポロジを “inet-henge-master”で表示させるには “手動”という点が課題となり、JSON ファイルを作成しなければいけないことを考えると理想的ではない。

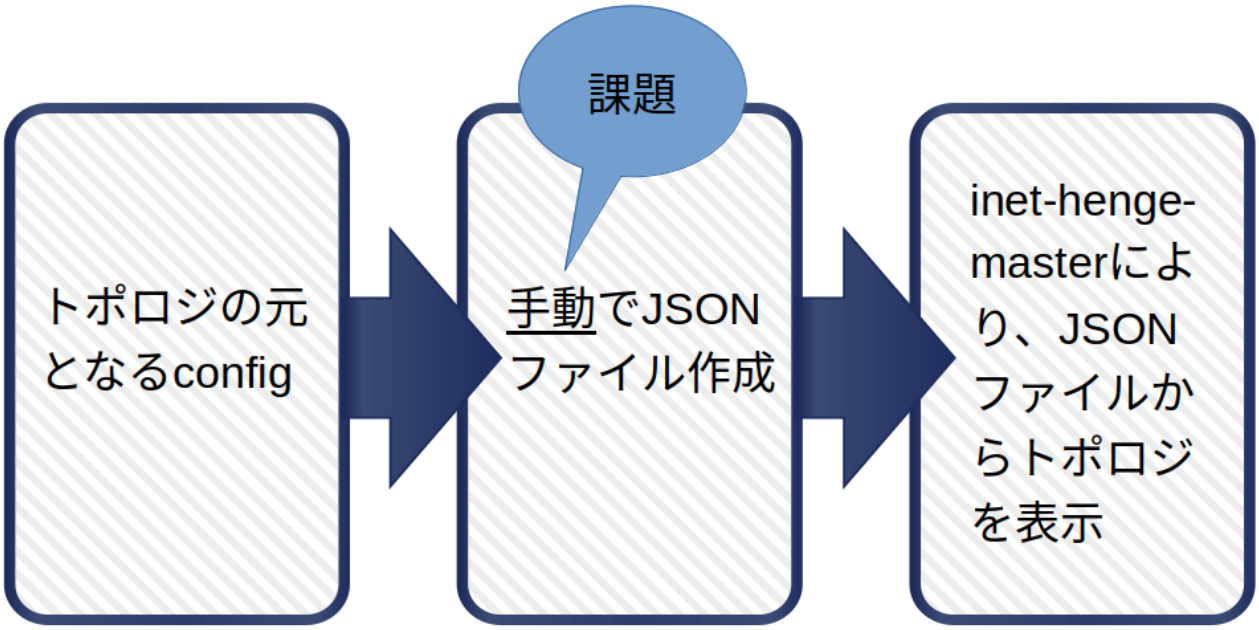


図 7　“inet-henge-master”の動作プロセス。小規模なネットワークトポロジの表示に関しては、現行のプロセスのままで利用することが容易だが、大規模なネットワークトポロジを “inet-henge-master”で表示させるには JSON ファイルの手入力が膨大な数になる。すなわち “手動”という点が、“inet-henge-master”を活用する際の一番の課題となっている。

4　NTG(Network Topology Grapher)

前章では本研究で用いる既存のシステムについて説明をした。この章では、前章で提起した問題点を解決し、本研究の目的を達成するために開発したシステムの説明を行う。

4.1　概念図

NTG とは、“Network Topology Grapher”の略で、本研究で開発したシステムの名称である。NTG は、config ファイルから JSON ファイルを作成し、“inet-henge-master”に JSON ファイルを読み込ませ、トポロジ表示をするまでの一連の動作を自動で行う。NTG の概念図を図 8 に示す。

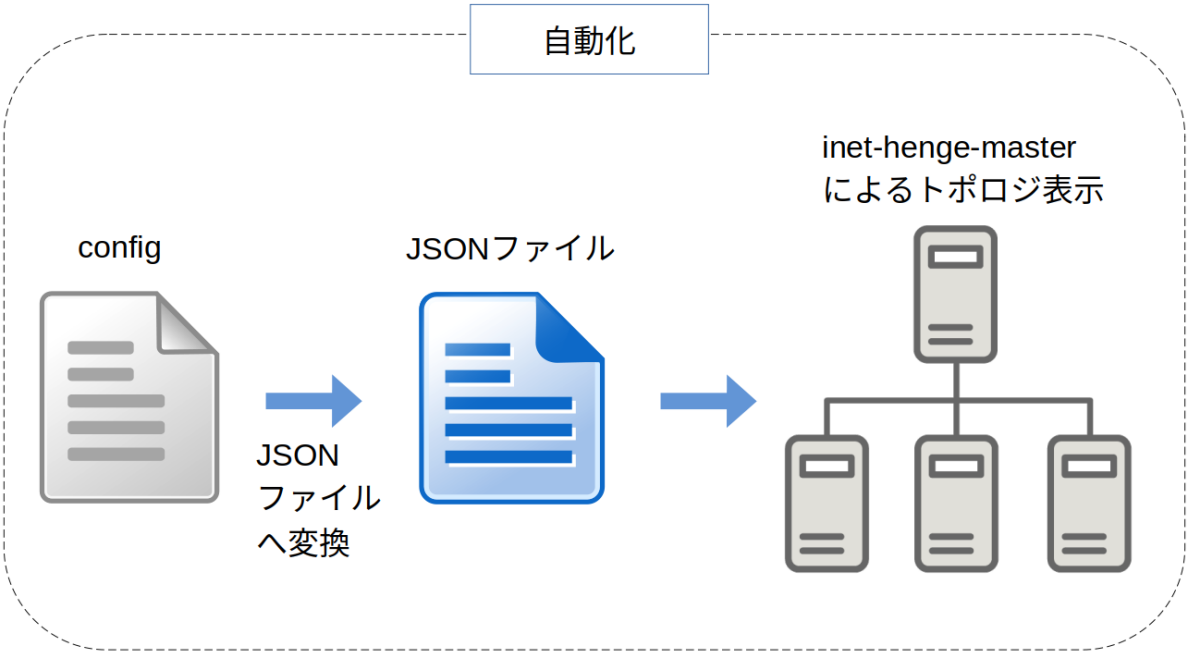


図 8　NTG の概念図。NTG(Network Topology Grapher) とは、本研究で開発したシステムで、config ファイルから JSON ファイルを作成し、“inet-henge-master”に JSON ファイルを読み込ませ、トポロジ表示をするまでの一連の動作を自動で行うシステムである。

4.2　15501 演習室

NTG の正常動作を確認するために大阪産業大学 15 号館 5 階 15501 演習室 (以下、「演習室」という) のトポロジ表示を試みた。演習室のトポロジは、15501_L3SW、Hubmain、hubmain0、hubmain03、hubmain04 の 5 台の機器から構成されている。演習室のトポロジ例を図 9 に示す。トポロジを構成している 15501_L3SW の “cdp neighbors” を図 10 に、Hubmain の “cdp neighbors” を図 11 に、hubmain0 の “cdp neighbors” を図 12 に、hubmain03 の “cdp neighbors” を図 13 に、hubmain04 の “cdp neighbors” を図 14 にそれぞれ示す。図 9 における “FSW-15-8” はフロアスイッチといい、主にフロアごとに設置して複数のエッジスイッチ[*2]を束ねるスイッチのことである。

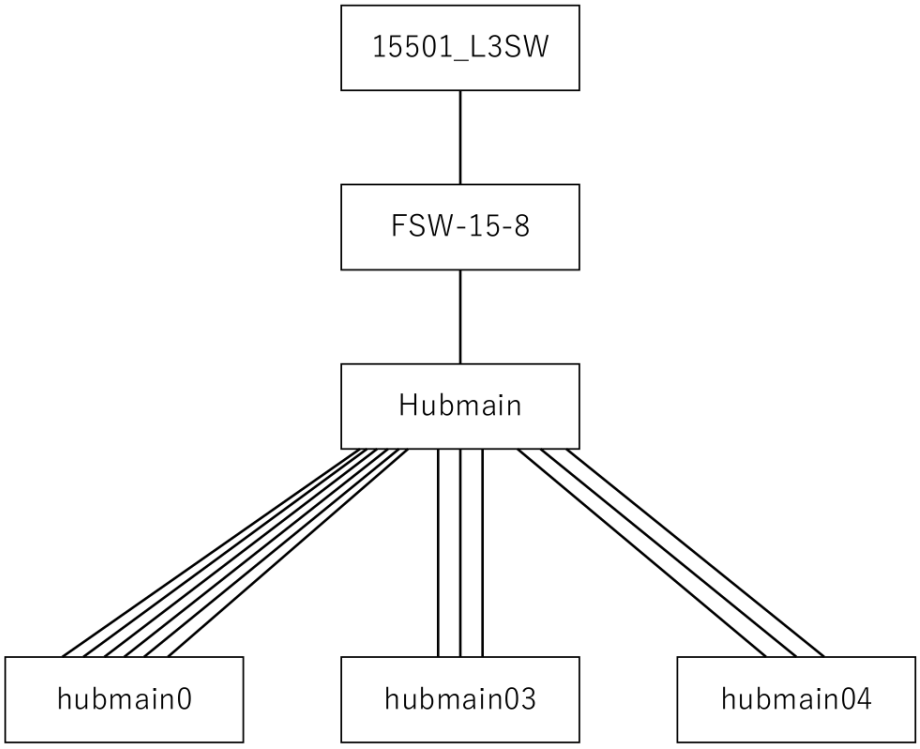


図 9　15501 のトポロジ構成図。演習室のトポロジは、15501_L3SW、Hubmain、hubmain0、hubmain03、hubmain04 の 5 台の機器から構成されている。“FSW-15-8” はフロアスイッチといい、主にフロアごとに設置して複数のエッジスイッチを束ねるスイッチのことである。

```
1 Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
2                   S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
3                   D - Remote, C - CVTA, M - Two-port Mac Relay
4
5 Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
6 FSW-15-8.osaka-sandai.ac.jp        Gig 0/48          145               S I   WS-C2960X Gig 1/0/2
```

図 10　15501_L3SW の “cdp neighbors”。6 行目の “FSW-15-8.osaka-sandai.ac.jp” は図 9 における “FSW-15-8” と同義である。行頭の行番号は説明のために追加したものであり、実際には行番号は表示されない。

[*2] 主に拠点内の接続に使用するスイッチのこと。ここでは Hubmain、hubmain0、hubmain03、hubmain04 を指す。

```
1 Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
2                   S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
3                   D - Remote, C - CVTA, M - Two-port Mac Relay
4
5 Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
6 FSW-15-8.osaka-sandai.ac.jp        Gig 1/0/36        152           S I   WS-C2960X Gig 1/0/5
7 hubmain0         Gig 1/0/38        151              S I   C9200L-48 Gig 1/0/47
8 hubmain0         Gig 1/0/39        171              S I   C9200L-48 Gig 1/0/48
9 hubmain0         Gig 1/0/37        138              S I   C9200L-48 Gig 1/0/46
10 hubmain0        Gig 1/0/41        130              S I   C9200L-48 Gig 2/0/47
11 hubmain0        Gig 1/0/40        129              S I   C9200L-48 Gig 2/0/46
12 hubmain0        Gig 1/0/42        173              S I   C9200L-48 Gig 2/0/48
13 hubmain04       Gig 1/0/48        156              S I   WS-C2960G Gig 0/48
14 hubmain04       Gig 1/0/47        156              S I   WS-C2960G Gig 0/47
15 hubmain04       Gig 1/0/46        156              S I   WS-C2960G Gig 0/46
16 hubmain03       Gig 1/0/45        123              S I   WS-C2960G Gig 0/48
17 hubmain03       Gig 1/0/44        137              S I   WS-C2960G Gig 0/47
18 hubmain03       Gig 1/0/43        123              S I   WS-C2960G Gig 0/46
```

図 11　Hubmain の “cdp neighbors”。6 行目の “FSW-15-8.osaka-sandai.ac.jp” は図 9 における “FSW-15-8” と同義である。行頭の行番号は説明のために追加したものであり、実際には行番号は表示されない。

```
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
                  D - Remote, C - CVTA, M - Two-port Mac Relay

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Hubmain          Gig 1/0/47        174             R S I  WS-C3850- Gig 1/0/38
Hubmain          Gig 1/0/48        133             R S I  WS-C3850- Gig 1/0/39
Hubmain          Gig 1/0/46        140             R S I  WS-C3850- Gig 1/0/37
Hubmain          Gig 2/0/47        161             R S I  WS-C3850- Gig 1/0/41
Hubmain          Gig 2/0/46        162             R S I  WS-C3850- Gig 1/0/40
Hubmain          Gig 2/0/48        141             R S I  WS-C3850- Gig 1/0/42
```

図 12　hubmain0 の “cdp neighbors”。

```
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
                  D - Remote, C - CVTA, M - Two-port Mac Relay

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Hubmain          Gig 0/48          144             R S I  WS-C3850- Gig 1/0/45
Hubmain          Gig 0/47          166             R S I  WS-C3850- Gig 1/0/44
Hubmain          Gig 0/46          146             R S I  WS-C3850- Gig 1/0/43
```

図 13　hubmain03 の “cdp neighbors”。

```
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
                  D - Remote, C - CVTA, M - Two-port Mac Relay

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Hubmain          Gig 0/48          156             R S I  WS-C3850- Gig 1/0/48
Hubmain          Gig 0/47          164             R S I  WS-C3850- Gig 1/0/47
Hubmain          Gig 0/46          159             R S I  WS-C3850- Gig 1/0/46
```

図 14　hubmain04 の “cdp neighbors”。

4.3　演習室のトポロジ表示

NTG によるトポロジ自動表示の出力結果を図 15 に示す。またこの出力結果は、図 9 における演習室のトポロジと一致しており、動作の正常性が証明された。

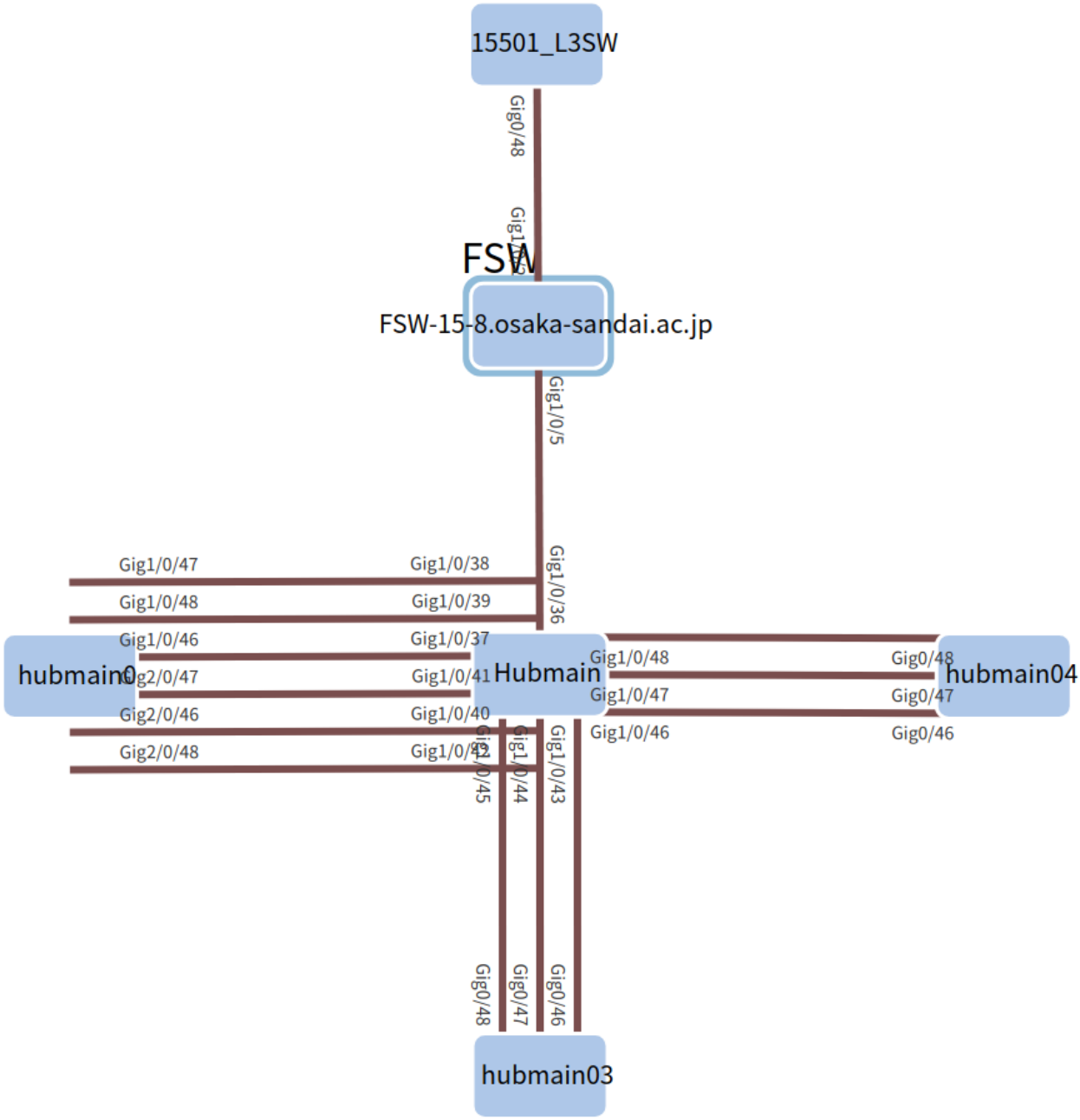


図 15　NTG を使用し、config から可視化させた 15501 演習室のトポロジ。図 9 の構成および前述のリンク情報と一致していることが確認できる。

5 NTG の検証

この章では、トポロジ表示の自動化された NTG を用いて、正常な状態のトポロジに対し何らかのトラブルを与え、表示されたトポロジから異常が検知できるのかを検証する。演習室のトポロジは現在運用中であり、トラブルを避けるために、本研究では本学科[*3]の 3 年時に履修できるネットワーク構築演習の自由設計課題で実際に扱った構成を “Cisco Packet Tracer”[*4]を用いて config を作成し、意図的にエラーを起こし検証を行う。本研究では、構築時のエラーとしてポートのシャットダウンの検証を行う。また、“Cisco Packet Tracer”により構築した正常な状態のトポロジを図 16 に、NTG を用いたトポロジ表示を図 17 にそれぞれ示す。これらをデフォルトの状態として、以下検証を行っていく。

[*3] 大阪産業大学 デザイン工学部 情報システム学科

[*4] Cisco 社の教育機関である、Cisco Networking Academy が公開しているフリーのソフトウェアのこと。ソフトウェア上で PC・ルータ・L2 スイッチ・L3 スイッチなどを配置し、それらを接続してネットワークを構築することができる。

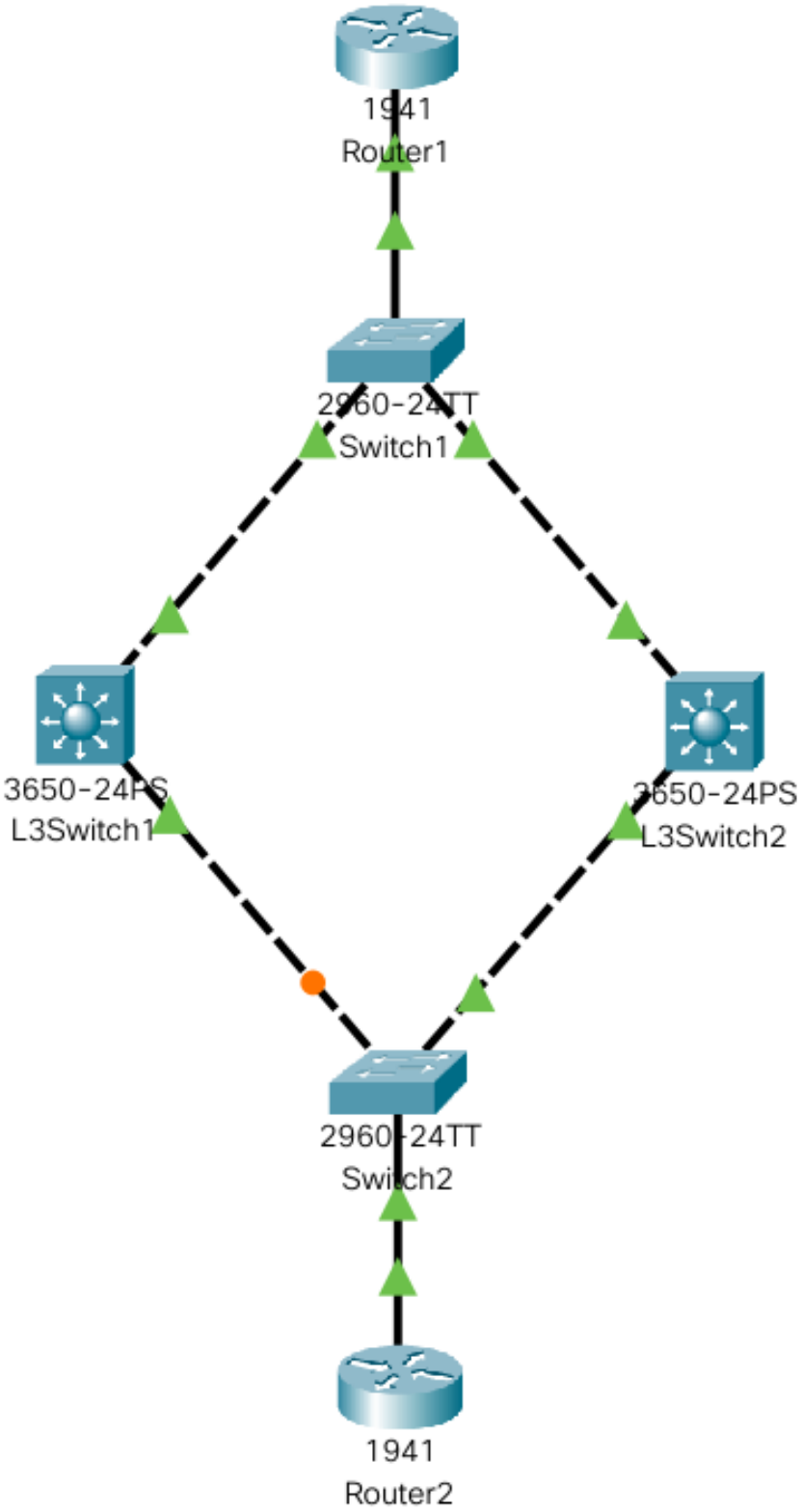


図 16　“Cisco Packet Tracer”で構築した検証用のトポロジ図。リンク上の三角形は疎通が行えていることを示している。丸印は、STP によるブロックポートとしてポートが閉じた状態を示している。Switch1、L3Switch2、Switch2 間で疎通不能のトラブルが起こった場合にはブロックポートが開き、Switch1、L3Switch1、Switch2 間で疎通が再開される。リンクの実線と破線はケーブルの違いである。

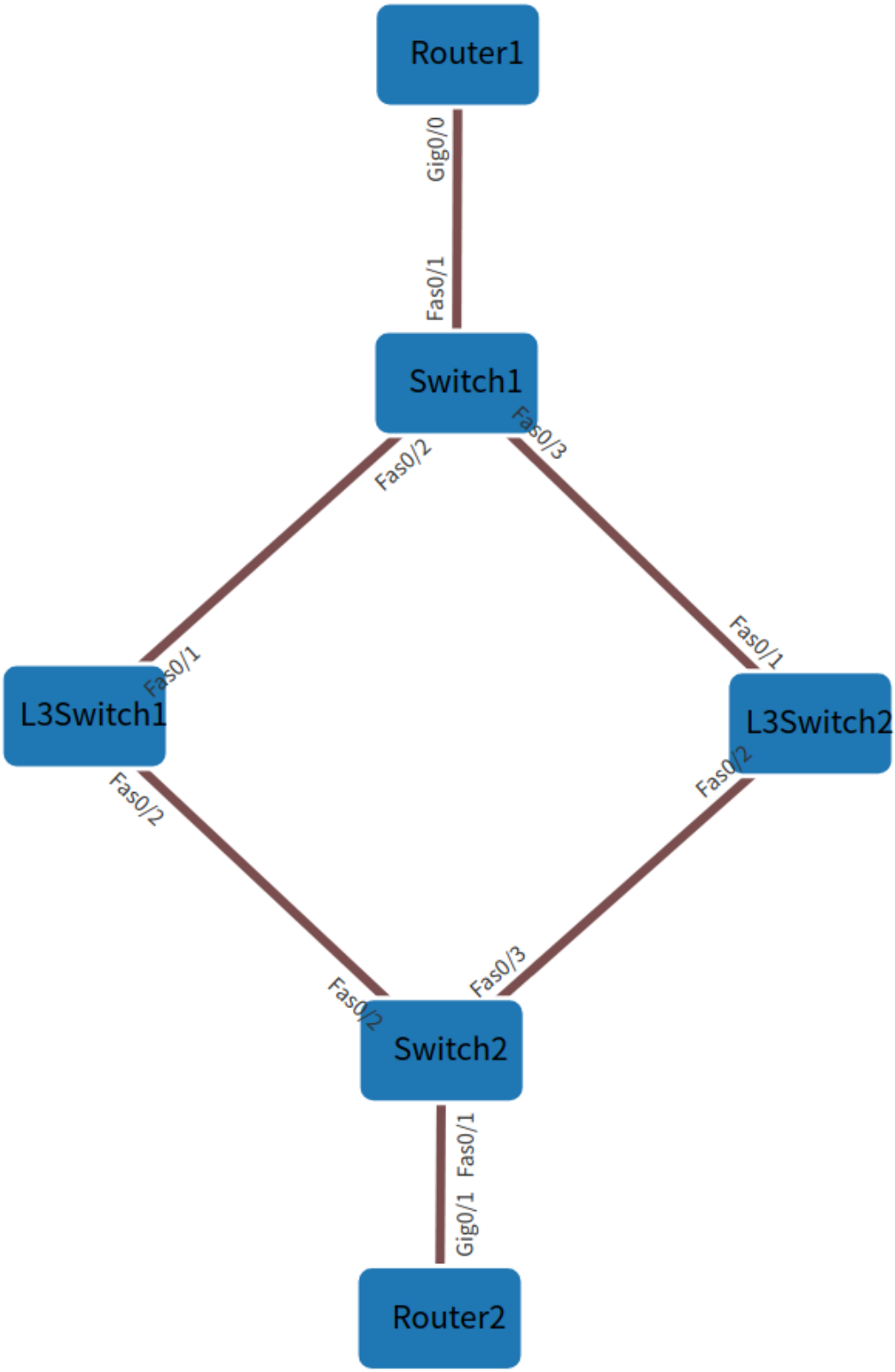


図 17　NTG を用いて表示した検証用のトポロジ図。

5.1 ポートのシャットダウン

本研究では、検証としてケーブルの接続されているポートのシャットダウンを行う。今回、ポートのシャットダウンを行ったのは、Switch1 の Gig0/1 のポートである。ポートのシャットダウンは、コンソール画面のグローバルコンフィギュレーションモードから該当ポートを選択し、“shutdown”コマンドで実行できる。解除したい場合は、“no shutdown”で解除できる。そして、ポートをシャットダウンすると “cdp neighbors”による隣接機器の情報が更新される。また、今回隣接関係を所持していた Router1 も Switch1 のポートがシャットダウンされることで、ホールドタイムが経過した後 “cdp neighbors”の情報から Switch1 が削除される。ポートのシャットダウンを行った際の “Cisco Packet Tracer”による Router1 および Switch1 の出力を図 18 に示す。Router1 と Switch1 間の疎通が行えていないことが確認できる。また、ポートシャットダウン前の Switch1 の “cdp neighbors”出力結果を図 19 に、シャットダウン後の Switch1 の “cdp neighbors”出力結果を図 20 に、ポートシャットダウン前の Router1 の “cdp neighbors”出力結果を図 21 に、シャットダウン後の Router1 の “cdp neighbors”出力結果を図 22 にそれぞれ示す。こちらもシャットダウン前後で、互いの機器から隣接情報が削除されていることが確認できる。

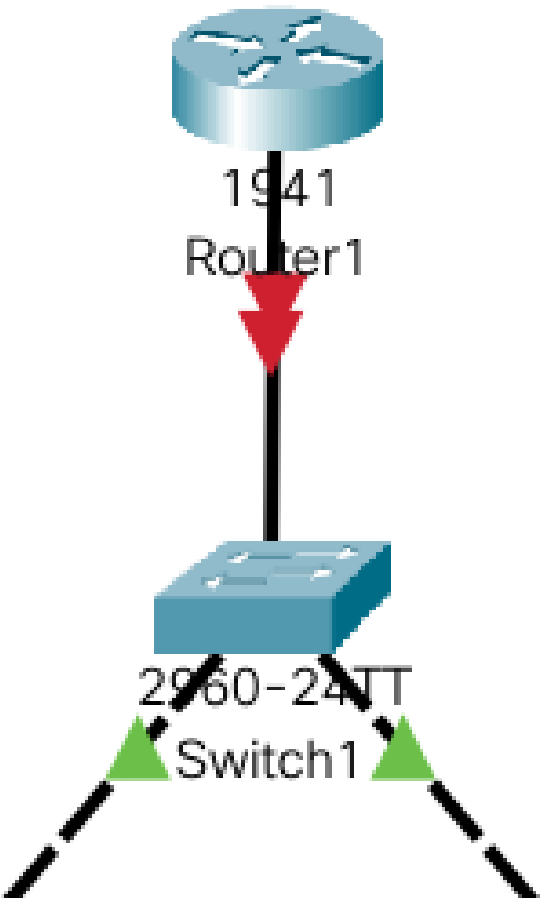


図 18 “Cisco Packet Tracer”によるシャットダウン後のトポロジ表示を拡大したもの。Router1 および Switch1 間の逆三角形は疎通ができていないことを示している。前述したデフォルトの状態と比較すると三角形から逆三角形へ変化しており、ポートのシャットダウンによって疎通が途絶えたことが確認できる。

```
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone
Device ID     Local Intrfce   Holdtme    Capability   Platform    Port ID
L3Switch1       Fas 0/2         167           S       3560        Fas 0/1
L3Switch2       Fas 0/3         135           S       3560        Fas 0/1
Router1       Fas 0/1         150           R       C1900       Gig 0/0
```

図 19　シャットダウン前の Switch1 の “cdp neighbors”。

```
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone
Device ID     Local Intrfce   Holdtme    Capability   Platform    Port ID
L3Switch1       Fas 0/2         167           S       3560        Fas 0/1
L3Switch2       Fas 0/3         135           S       3560        Fas 0/1
```

図 20　シャットダウン後の Switch1 の “cdp neighbors”。

```
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone
Device ID     Local Intrfce   Holdtme    Capability   Platform    Port ID
Switch1       Gig 0/0         157           S       2960        Fas 0/1
```

図 21　シャットダウン前の Router1 の “cdp neighbors”。

```
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone
Device ID     Local Intrfce   Holdtme    Capability   Platform    Port ID
```

図 22　シャットダウン後の Router1 の “cdp neighbors”。

5.2　出力結果

ポートのシャットダウン後の NTG による出力結果を図 23 に示す。Router1 と Switch1 間のリンクが失われたことが図から読み取れる。

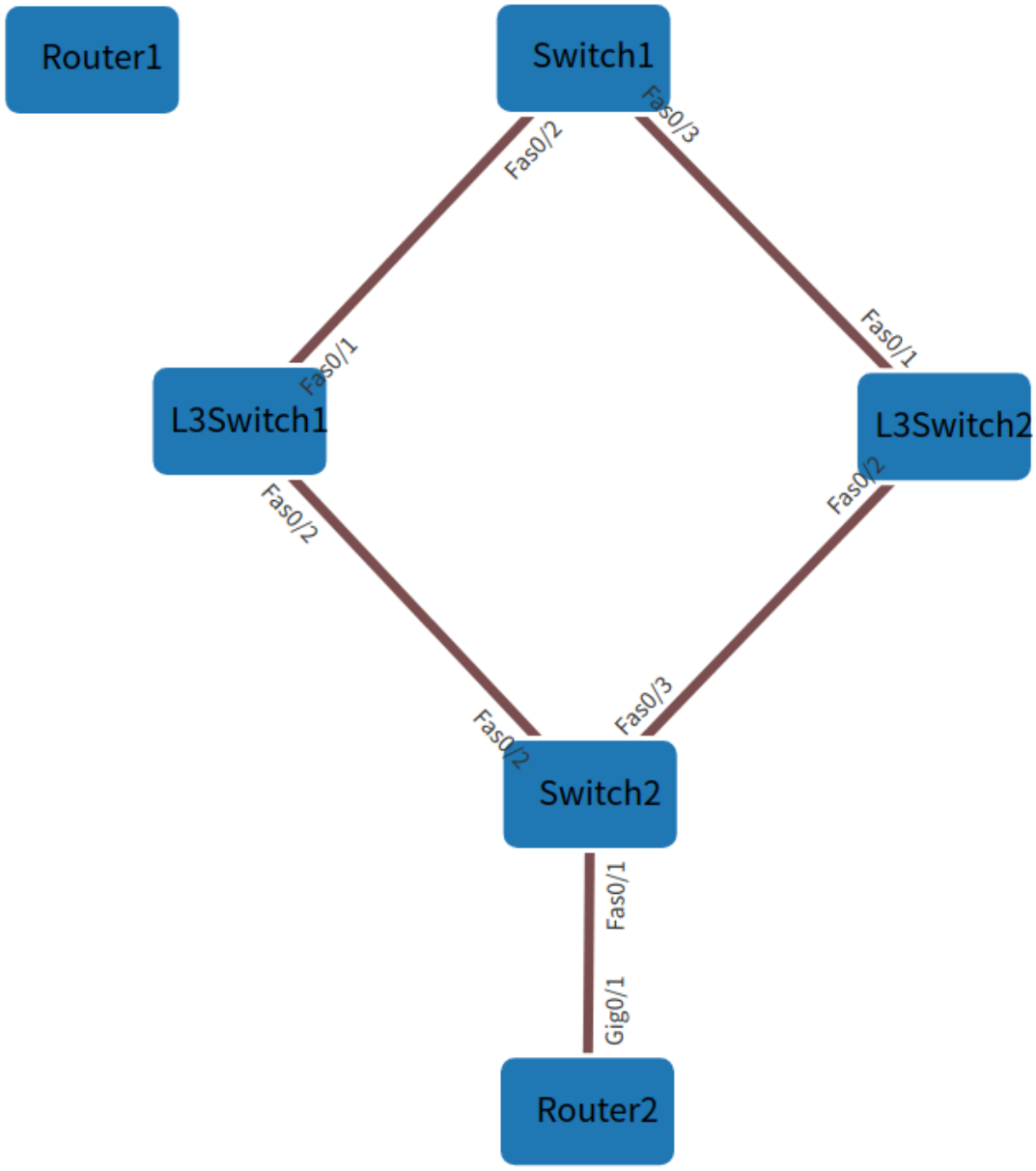


図 23　NTG によるシャットダウン後のトポロジ表示。Router1 と Switch1 間のリンクが失われたことが図から読み取れる。

6 結果と考察

この章では、本研究で開発した NTG のプログラム、および検証結果に関する考察を行う。

6.1 NTG の考察

本研究では、config を元にトポロジを表示させるというシステムを開発した。config といういわゆるテキスト形式の情報からトポロジを構成するために必要な情報を抜き出し、“inet-henge-master”に対応した JSON ファイルへと変換するところに一番時間を要した。また、NTG の正常動作および実用性を確認するために、自分で用意した config だけでなく演習室の config を用いて確認をした際、NTG のプロトタイプは “cdp neighbors”の情報からリンク情報を誤って取得してきてしまった。“cdp neighbors”の “Capability”の項目が複数表示されていたため、レコード毎で見たときの全体のフィールド数が異なってしまい、求めていた情報とは違うものを取得していたことが原因だった。プロトタイプはフィールドの前から何番目という情報の取得方法のみでプログラムを動かしていたが、前から数える方法で正常動作していない箇所に関しては、後ろから何番目のフィールドという指示で情報を取得することにした。これにより演習室のような実際に運用・動作している config でも、リンク情報を取得することができるようになった。

6.2 検証結果と考察

NTG では物理的なトポロジを表示させている。正常に接続・運用されているトポロジは正しく表示する前提が必要であったため、本研究では演習室のトポロジを表示し、正常動作を証明した。そして、本研究のもう一つの目的であるトポロジ異常の発見である。NTG は、物理的なトポロジを表示させており、リンクアグリゲーション等の論理的なトポロジの表示まではサポートしていない。このため検証で行うトラブル例も物理的にリンクがダウンするポートのシャットダウンを行った。ポートのシャットダウンを行うと、該当ポートのリンクが絶たれる。NTG でもリンクが失われたことが表示され、物理的なエラーを図として発見することができるようになった。

7 結論

本研究ではネットワークのトポロジを構成している “Cisco Catalyst”の設定情報から、トポロジ表示を自動で行うシステム開発を行った。その結果、“running-config”と “cdp neighbors”の 2 つの設定情報から物理的なトポロジを表示させることに成功し、ポートや物理層の異常によるトポロジの欠落を発見することができた。今後の課題としては、マルチベンダー間でのトポロジの表示、リンクアグリゲーション等の論理的なトポロジの表示、VLAN 情報の追加が挙げられる。

謝辞

本研究を進めていく上で、担当教員の大垣 斉准教授からご指導およびご協力を頂きました。また、ご協力頂いた情報教育システム研究室所属の学生および卒業生の方々に深く感謝いたします。

参考文献

[1] 10 月 1 日に株式売買システムで発生した障害について. https://www.jpx.co.jp/corporate/news/news-releases/0060/20201019-01.html, Oct 2020.

[2] inet-henge-master. https://github.com/codeout/inet-henge/, 2020.

付録 A　ソースコード

A.1　hostname と cdp 情報整形用の View

Listing 1　hostname と cdp 情報抽出用の View(cdp_arrange.sh)

```
#!/bin/bash

for i in `ls ../config/`
  do
    sed -e '/hostname/p' -e '1,/Device/d' ../config/$i
  done | perl -npe 's/\r\n/\n/' > cdp.conf
```

A.2 ノード情報整形用の View

Listing 2 ノード情報抽出用の View(nodes_arrange.sh)

```
#!/bin/bash

cat cdp.conf | awk '
  $1 == "hostname"{
    print $2
  }
  $1 != "hostname"{
    print $1
  }'| sort | uniq > node.conf
```

A.3 リンク情報作成に使用する CSV 出力用の View

Listing 3 リンク情報作成に使用する CSV 出力用の View(make_csv.sh)

```
#!/bin/bash

cat cdp.conf | awk '

  $1 == "hostname"{
    host = $2
    #print host
  }

  $1 != "hostname" {
    if(NF != 0){
      neighbor = $1
      local_if = $2$3
      neighbor_if = $(NF-1)$NF
      cdp_csv = ""host","neighbor","local_if","neighbor_if""
      print cdp_csv
    }
  }' > cdp.csv
```

A.4 リンク情報の重複回避用の View

Listing 4 リンク情報の重複回避用の View(csv_filter.sh)

```
#!/bin/bash

FILE="cdp.csv" 入力ファイル名#
OUT="result.csv" 出力ファイル名#

FFLAG=1 最初の一行目だけ無条件で追加するためのフラグ変数#

if [ -e $OUT ] 出力ファイルが実行時にあった場合削除#
then
  rm -fr $OUT
fi

for i in `cat $FILE`
do
  if [ $FFLAG = 1 ] 最初の一行目だけ無条件で追加する#
  then
    echo $i >> $OUT
    FFLAG=0
    continue
  fi
  SFLAG=0 比較後破棄するか追加するかを判定する変数#

  for k in `cat $OUT` #
  do
    for j in `seq 1 4` 4つの要素を比較するためのループ#
    do
      if [ $(($j % 2)) = 1 ] 比較元の要素番号が奇数の場合は比較先の要素番号を、偶数の場合は
#+1-1
      then
        SUM=`echo $((j+1))`
      else
        SUM=`echo $((j-1))`
      fi

      A=`echo $i | awk -F'[,]' -v "num=${j}" '{print $num}'` 比較元の要素抽出#
      B=`echo $k | awk -F'[,]' -v "num=${SUM}" '{print $num}'` 比較先の要素抽出#
```

```
        if [ "$A" != "$B" ] 一つでも不一致なら#break
        then
          break
        fi

        if [ $j = 4 ] すべて一致で破棄するフラグをたてる#
        then
          SFLAG=1
        fi
      done

      if [ $SFLAG = 1 ] 破棄フラグがたっているのでこれ以上の比較は不要なので#break
      then
        break
      fi
    done

    if [ $SFLAG = 0 ] 破棄フラグがたっていないので追加#
    then
      echo $i >> $OUT
    fi
  done
```

A.5 ノード情報取得用の View

Listing 5 ノード情報取得用の View(nodes.sh)

```
#!/bin/bashノード作り

#
cat node.conf | awk '
  BEGIN{
    nodes_json = "{\"nodes\":[";
  }

  NF != 0{
    name = "\"name\":\"" $1 "\""
    nodes = "{" name "}";
    nodes_json = nodes_json nodes ","
  }

  END{
    sub(/,$/, "", nodes_json); # 末尾の余分なを排除","
    nodes_json = nodes_json "],";
    print nodes_json
  }'> nodes.txt
```

A.6 リンク情報取得用の View

Listing 6 リンク情報取得用の View(links.sh)

```
#!/bin/bash

cat result.csv | awk -F '[,]' '
  BEGIN{
    links_json = "\"links\":[";
  }

  NF != 0{
    source1 = "\"source\":\"" $1 "\""
    target1 = "\"target\":\"" $2 "\""
    source2 = "\"source\":\"" $3 "\""
    target2 = "\"target\":\"" $4 "\""
    links = "{" source1 "," target1 ",\"meta\":{\"interface\":{" source2 "," target2 "}}}"
    links_json = links_json links ","
  }

  END{
    sub(/,$/, "", links_json); # 末尾の余分なを排除","
    links_json = links_json "]}"
    print links_json
  }'> links.txt
```

A.7 JSON 出力用の View

Listing 7 JSON 出力用の View(make_JSON.sh)

```
#!/bin/bash

cat nodes.txt > ../topology/topology.txt

cat links.txt >> ../topology/topology.txt

cat ../topology/topology.txt > ../inet-henge-master/example/topology.json
```

A.8 トポロジ出力用の View

Listing 8 トポロジ出力用の View(topology.sh)

```
#!/bin/bash

xdg-open http://0.0.0.0:8000/inet-henge-master/example/topology.html

cd ..

python3 -m http.server
```

A.9 NTG 実行用の View

Listing 9 NTG 実行用の View(application.sh)

```
#!/bin/bash

bash cdp_arrange.sh

bash nodes_arrange.sh

bash make_csv.sh

bash csv_filter.sh

bash nodes.sh

bash links.sh

bash make_JSON.sh

bash topology.sh
```