仮想マシンを用いたネットワーク構築演習における正解トレースに基づく答案評価システムの提案

立岩 佑一郎[1]　高橋 直久[1]

概要：ネットワーク構築演習では「ネットワークが繋がらない／繋がってしまう」という，通信データの到達性に関する要件の達成に躓く学習者が多い．そして，自身のネットワーク（答案）の誤りを発見し修正できない学習者は，指導者に救援を仰ぐことになる．しかし，指導者であっても誤りを見つけ修正のためのヒントを出すのは時間がかかることが多い．これまでに，我々は仮想マシンによるネットワークを用いたネットワーク構築演習システム LiNeS を開発し，LiNeS のネットワークにおける通信データの到達性に関わる設定を評価する機能を開発した．しかし，評価にて使用される条件（以降，正解条件と呼ぶ）を指導者が演習問題毎に定義する必要がある上に，正解条件の導出法が確立されていないことと，ネットワークのノード数の増加に応じて正解条件の数が増大することから，指導者への負担が大きかった．そこで本稿では，通信データの通過したノードおよび各ノードでの通信データに対する処理の系列をトレースと呼び，教師により指定されたトレースに基づいて正解条件を機械的に導出し，導出した正解条件により答案を評価するシステムを提案する．

1. まえがき

大学や専門学校などでの伝統的なネットワーク構築演習では，Linux 計算機，ルータ，スイッチングハブなどのネットワーク実機を用いて，クライアント／サーバシステムの構築や，マルチセグメントネットワークの構築が行われてきた．このとき，複数の学習者でグループを組み機器を共有することで，数に限りのある機器での演習を可能にしている．

しかし，専用の機器を用意する経済的な負担や，機器の共有によるグループ内での操作時間の偏りが問題となっていた．これに対し，近年，仮想マシンを利用した演習が実施されるようになってきた．一台の計算機上で複数の仮想マシンを稼働させて仮想的なネットワーク（以降，VMN と呼ぶ）を実現できる．PC 演習室の共用 PC で VMN を用いた演習は，この問題を解決できる．

筆者らは VMN によるネットワーク構築演習システム LiNeS を開発した [1]．このシステムは，User-mode Linux[2]（以降，UML と呼ぶ）による VMN を実現する機能，学習者による VMN の操作を支援する GUI を持つ．LiNeS による演習用ネットワークは次のモデルとなる．ISO 七階層モデルにおいて第三層から第七層は，一般的な Linux マシンの実装と同様である．第二層がイーサネットのみである．また，第一層において，リピータとスイッチングハブは，サーバ，クライアント，L3 ファイアウォール（以降，L3FW と呼ぶ），L2 ファイアウォール（以降，L2FW と呼ぶ），ルータと半二重通信する．そして，通信速度は 100Mbps，信号の到達距離は無限大で，コリジョンは発生しない．学習者はネットワークトポロジーの設計，Linux コマンドの実行，Linux システムファイルの編集により要求を満たす VMN を構築する．この演習において学習者が指定可能な VMN の設定項目は，演習問題に指定されたもの（以降，解答項目と呼ぶ）に限定される．

一方で，指導者の指導対象がグループから個人へ変わり，その数が増加したために，救援を求める学習者への対応が不足するという問題（問題 2）が生じた．LiNeS による演習におけるこの問題は，通信データの到達性の実現に起因するものが多い．

到達の例では，ノード A からノード B を経由してノード C へ通信データの配送を実現する演習問題において，学習者の誤設定によりノード B までの配送となったり，ノード D 経由での配送となったりする．不達の例では，ノード A からノード C のアドレスへ通信データを送信し，途中のノード B においてそれを遮断するという演習問題において，学習者の誤設定により通信データがノード C へ配送されてしまう．

[1] 名古屋工業大学
Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya-shi, Aichi-ken 466–8555, Japan

誤設定を学習者が修正できずに指導者へ救援を求めたとき，指導者は誤っている解答項目を発見し，その修正のためのヒントを学習者に伝える．ヒントは，例えば，「ノード A とノード C でネットワークアドレスが異ならなければならない」,「ノード C の属するネットワークへの経路をノード A の経路制御表に追加しなければならない」などである．

問題 2 の解決のために，筆者らは LiNeS の VMN における通信データの到達性に関わる設定を評価する機能を開発した [3]．この機能は，解答項目の値を収集する機能，到達性に関わる設定を評価する条件（以降，正解条件と呼ぶ）で解答項目の値を評価する機能，不成立となる正解条件を提示する機能から構成される．学習者は，不成立の正解条件を参考に，誤りを見つけて正しく修正する．

しかし，正解条件は指導者により定義されるものであり，これを演習問題毎に漏れなく定義することは指導者にとって負担が大きい（問題 3)．この理由は次の 2 つである；1) 正解条件の導出方法は確立されておらず，指導者による TCP/IP の解釈に基づくものである．2) ネットワークのノード数が増加するにつれ，定義する正解条件がその増加数が階差数列に近似するように増加する．

本稿では，通信データの通過したノードおよび各ノードでの通信データに対する処理の系列をトレースと呼び，教師により指定されたトレース（以降，正解トレース例と呼ぶ）に基づいて答案を評価するシステムを提案する．提案システムは，正解トレース例から正解条件を導出し，正解条件に基づいて答案を評価し，不成立となる正解条件を表示する．そして評価実験において，学習者の作成したネットワークを用いて，この正解条件の正当性の評価を論じる．

2. ネットワーク構築演習

2.1 演習内容

対象とする学習者は，TCP/IP を習得済みであるが，ネットワークを構築したことがない者とする．演習の達成目標は，学習者が下記の項目に関する要件を満たすネットワークを設計でき，その設計に従って，リピータハブ，スイッチングハブ，L3FW，L2FW，サーバ，クライアントを設定してネットワークを構築できるようになることである．ここで，L3FW と L2FW は Linux マシンで，iptables により実現されている．サーバは Linux マシンで Linux のサーバソフトウェアがインストールされている．そして，クライアントは Linux マシンである．

(1) IP アドレスとサブネットマスク
(2) 同一セグメントの IP ネットワーク
(3) デフォルトルート
(4) 静的経路制御
(5) 動的経路制御
(6) サーバサービス
(7) IP マスカレード
(8) NAT
(9) パケットフィルタリング
(10)DMZ

本稿では，通信プロトコル，ペイロード，ペイロードの送信元ノード，ペイロードの宛先アドレスを通信属性と呼び，ペイロードが滞在したノードを滞在ノードと呼ぶ．トレースは，ある通信におけるすべての滞在ノードとそれらに関する滞在順序，および滞在ノードでのペイロードとそれを含む通信データへの処理を表したものである．

演習問題は，通信属性とトレースの一部を満たす通信を可能とするネットワークの構築を要求するものである．そして，演習問題にて使用するノードは決定されている．また，フィルタリングによる通信データの破棄，または直前の滞在ノードがリピータである滞在ノードでの通信データの破棄以外を設定誤りと見なす．構築において，学習者は演習問題に指定された解答項目のみ設定可能である．

解答項目に指定可能なものは次の通りである．ルータでは，ルーティングテーブル，ARP テーブル，ネットワークデバイスである．ネットワークデバイスでは，IP，サブネットマスク，MAC，接続先ネットワークデバイスである．サーバとクライアントでは，ルータでの解答項目に加え，待受プロセスのリッスンプロトコル，リッスンポート，リッスン IP である．L3FW では，ルータでの解答項目に加え，NAT およびフィルタである．ブリッジおよびリピータでは，ネットワークデバイスの接続先ネットワークデバイスである．L2FW では，ブリッジでの解答項目に加え，NAT およびフィルタである．

演習の流れは次の通りである．演習開始前に，指導者は複数の演習問題を作成する．演習中，学習者は演習問題に対する答案を作成する．指導者は答案の正誤を判定する．答案が誤りの学習者は，答案が正解となるまで，答案を修正する．理解不足のため答案を完成できない場合や誤りを修正できず正解にならない場合，学習者は指導者の救援を仰ぐ．

2.2 演習システム LiNeS

UML は Linux 上で Linux をシミュレートする仮想マシンである．LiNeS は UML による VMN を実現する機能，VMN の操作支援 GUI，VMN の設定（解答項目）を定期的に取得し，その都度ネットワーク経由でデータベースへ保存する機能を持っている．LiNeS は LinuxPC 上で動作する．図 1 は，LiNeS の VMN の操作支援 GUI である．画面 (1) は VMN のネットワークトポロジーの編集フィールドで，画面 (2) は各々の UML の仮想コンソールである．

LiNeS では表 1 のノード種に示すノードを用いて VMN を構成できる．同時に稼働可能なノード数は稼働させている OS の空きメモリに依存し，例えば 2GB の空きメモリ

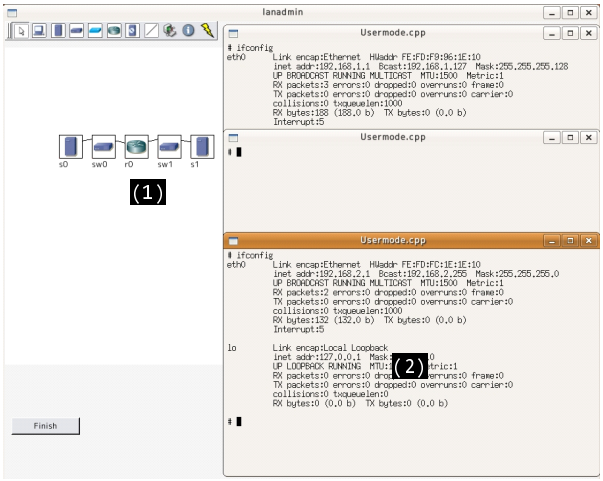

図 1　ネットワークの例

表 1　LiNeS のノード

ノード種	ネットワークデバイス数
サーバ	1
クライアント	1
ルータ	最大 9（設置時に指定する）
スイッチングハブ	5
リピータハブ	5
透過型ファイアウォール	最大 9（設置時に指定する
非透過型ファイアウォール	最大 9（設置時に指定する

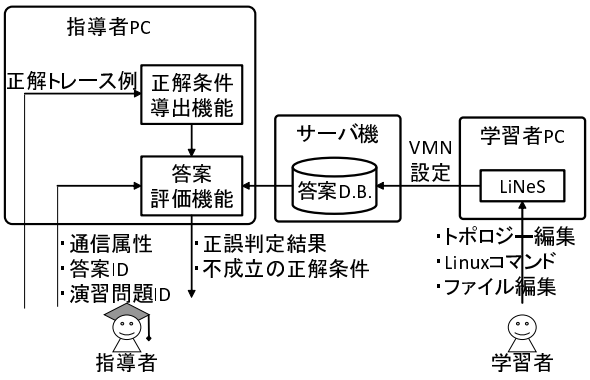

図 2　システム構成

であれば 10 ノード以上の実績がある．また，UML の仕様により，サーバ，クライアント，ルータ，透過型ファイアウォール，非透過型ファイアウォールは，スイッチングハブまたはリピータハブとしかネットワーク接続できない．ネットワーク接続は二つのノードにおいて各々のネットワークデバイスを相互に接続する．

3. 提案システムの実現法

3.1 システム構成

演習における LiNeS と提案システムの利用を図 2 に示す．提案システムは正解条件導出機能と答案評価機能から構成される．演習前に指導者は演習問題に対応した正解トレース例と通信条件を正解条件導出機能に与える．演習中，学習者は LiNeS の提供する VMN を操作することで，演習問題に解答する．また，LiNeS は VMN から設定を定期的に取得し，答案 D.B. にその都度保存する．

教師が通信属性，答案 ID，演習問題 ID を入力として答案評価機能に答案評価を要求すると，正解条件導出機能が演習問題 ID に対応した正解条件を導出し，答案評価機能が答案 ID に対応した答案を D.B. から読み込み，正解条件により答案を評価する．すべての正解条件が成立した場合，システムは答案が正解であると表示する．そうでない場合，システムは答案が不正解であると表示し，不成立となる正解条件を表示する．

文献 [3] に述べられている答案評価機能は，サーバ，クライアント，スイッチングハブから構成されるシングルセグメントのネットワークを対象としている．本稿の答案評価機能は，この機能を拡張して，ルータによるマルチセグメントのネットワークも対象としている．しかし，この実現法は文献 [3] に基づいているので，本稿では省略する．

3.2 正解条件

ネットワーク通信において，各ノードにおけるペイロードとそれを含む通信データに対する処理を記録することを考える．これは，通信属性とネットワークを入力とし，トレースを出力とする処理（以降，主処理と呼ぶ）の実行により達成できると言える．主処理の実行において，ペイロードを滞在させ処理するノード（滞在ノード），そのノードでの処理（以降，副処理と呼ぶ），および副処理の実行順序が主処理の入力に依存して決定される．また，副処理は滞在ノードを出力の一つとする．副処理を実行順序に従ってすべて実行し終えるとトレースを得られる．

以上により，演習問題における到達性の要求は，あるトレース（出力値）となるためのネットワーク（入力値）を要求するものであるとみなせる．そして，正解条件は，主処理への入力項目，トレース，および副処理の三つの関係である．不成立の正解条件の表示は，この要求の達成への支援となりうる．これは，正解条件から正解値を導出するのは，問題文から正解値を導出するより簡単であると思われるためである．また，解答を入力として正解条件を評価し，その評価結果の表示も，この要求達成のための支援となりうる．これは，評価結果から誤りの潜在範囲を読み取れると思われるためである．

3.3 正解条件導出のための副処理の要件

前節で述べたように，正解条件は，主処理への入力項目，トレース，および副処理の三つの関係である．このことから，正解条件の導出には副処理の定義が必要となることがわかる．主処理の実行において，滞在ノード，副処理，および副処理の実行順序が主処理の入力に依存して決定される．副処理を実行順序に従ってすべて実行し終えるとト

レースを得られる．誤りを絞り込みやすくするために下記の A，B に，ヒントを生成するために C に留意し，副処理をいくつかの処理（以降，副々処理と呼ぶ）の系列として定義する．

A) ある通信において各ノードで一つの滞在ノードを出力する処理を定義する．先述のように，滞在ノード，滞在順序，ペイロードへの処理を表したものがトレースであり，トレースの一部は演習問題に指定される．答案のトレースと正解とするトレース（以降，正解トレースと呼ぶ）を比較することで，答案の正誤を判定するためである．

B) 一つの副々処理の入出力ができるだけ少なくなるように副々処理を定義する．これは，次の理由である．ある副々処理の出力した滞在ノードが正解と異なるとき，その副々処理の入力が誤りである可能性が出現する．一方，ある副々処理の出力した滞在ノードが正解と同じとき，その副々処理の入力が正しい可能性が出現する．ある副々処理の入力は別の副々処理の出力か解答項目である．このため，一つの副々処理の入出力を少なくすることは，正誤の可能性が混在する副々処理を減らすことになり，誤りを絞り込みやすくする．

C) 実行される副々処理とその順序（以降，副々処理の実行系列と呼ぶ）が出力から導出できるように副々処理を定義する．入力にのみ依存して副々処理の実行系列が決定されると，誤った入力（すなわち誤答案）が誤った副々処理の実行系列を導出してしまう．この場合，正解の出力と副々処理により実行を遡って表された入力は，誤ったものとなってしまう．一方，正解の出力から導出できた副々処理の実行系列であれば，正しく入力を表すことができる．

3.4 正解条件導出のための副処理

通信データは，通信プロトコル，宛先 IP，送信元 IP，宛先ポート，送信元ポート，ICMP タイプ，宛先 MAC，送信元 MAC，ペイロードから構成される．

表 2 は，副処理をノード毎に分類して表したものである．列タイトル上段の ♢ はルータ，♣ はサーバかクライアント，♡ は透過型ファイアウォール，♠ は非透過型ファイアウォール，♮ はスイッチングハブかリピータハブである．列タイトル下段は，通信データに対する処理の種別を指しており，S は送信元ノードとしての送信処理，R は中継ノードとしての中継処理，D は宛先ノードとしての受信処理，F は中継ノードとしての破棄処理，O は送信元ノードとしての破棄処理，I は宛先ノードとしての破棄処理，B は中継ノードまたは宛先ノードとしての破棄処理である．項目値は各役割における実行順序である．例えば，ルータにおける通信データの中継は，「CheckDstMAC → L3Routing → SetDstMAC → Transmit」の順で副々処理が行われる．

SetSrcPort は通信データを入力とし，通信データの送信元ポートにランダム値を設定し，通信データを出力する．SetSrcIP は通信データ，L3Routing の結果である出力デバイス名，および滞在ノードを入力とし，通信データの送信元 IP に出力デバイスの IP を設定し，通信データを出力する．SetDstMAC は通信データ，ネットワーク，L3Routing の結果であるネクストホップ IP，および滞在ノードを入力とし，ARP テーブル検索または ARP 通信を実行して得られた MAC を，通信データの宛先 MAC に設定し，通信データを出力する．

L3Routing は，通信データ，滞在ノードを入力とし，通信データの宛先 IP をキーとしてルーティングテーブルを検索し，ネクストホップ IP と出力デバイス名を出力する．L2Routing は，通信データ，滞在ノード，ネットワークを入力とし，滞在ノードのデバイスの中から，通信データの宛先 MAC を持つ外部ノードのデバイスにネットワーク接続されているデバイスを見つけ，それを出力する．L1Routing は，通信データ，滞在ノード，通信データの滞在ノードへの入力デバイスを入力とし，滞在ノードのデバイスの中から，入力デバイス以外のデバイスを見つけ，それを出力する．

通信データ，滞在ノード，通信データの滞在ノードへの入力デバイス，通信データの滞在ノードからの出力デバイスを入力とし，iptables における filter テーブルにおいて，OFilter は OUTPUT チェインのルールを，IFilter は INPUT チェインのルールを，FFilter は Forward チェインのルールを評価し，フィルタリングの結果（適応または不適応）と滞在ノードを出力する．

また，同入力にて iptables における nat テーブルにおいて，ONAT は OUTPUT チェインのルールを，PostNAT は POSTROUTING チェインのルールを，PreNAT は PREROUTING チェインのルールを通信データに適用し，通信データを出力する．

CheckDstMAC は，通信データ，滞在ノード，そのノードへの入力デバイスを入力とし，入力デバイスの MAC と通信データの宛先 MAC の比較結果を出力する．FindProcess は，通信データと滞在ノードを入力とし，通信データを受信可能なリッスンプロセスを見つけ，滞在ノードとしてそのプロセス名を出力する．Transmit は，通信データ，滞在ノード，ネットワーク，出力デバイスを入力として，滞在ノードとして通信データの次の滞在ノード，そのノードへの入力デバイスを出力する．

3.5 トレースの指定法

演習問題にて指定されるトレースは自然言語であるため，これをそのままでプログラムに理解させることは難しい．本節では，トレースをプログラムに理解させやすく，指導者が簡潔に指定できる指定法を述べる．

演習問題では，送信元ノード，宛先アドレス，通信プロトコル，および送信するペイロードによる通信において，ペイロードが滞在したノードとその滞在順序に加えて，ノード

表 2 正解条件導出のための副処理

	♢				♣			♡		♠							♮
	S	R	D	B	S	D	B	R	F	S	R	D	B	O	F	I	R
SetSrcPort	1				1					1				1			
SetSrcIP	3				3					3,6				3,6			
SetDstMAC	4	3			4			5		9	6						
L3Routing	2	2			2					2,5	3			2,5	3		
L2Routing								2	2								
L1Routing																	1
OFilter										7				7			
IFilter												3				3	
FFilter								3	3		4				4		
ONAT										4				4			
PostNAT								4		8	5						
PreNAT								1	1		2	2			2	2	
FindProcess			2			2						4					
CheckDstMAC		1	1	1		1	1				1	1	1		1	1	
Transmit	5	4			5			6		10	7						2

内でプロセスによりペイロードが破棄されることまたは読み取られることがトレースとして指定される．ここでの破棄は，正しい設定でのフィルタリングによるペイロードの破棄，または直前の滞在ノードがリピータである滞在ノードにおいて，ペイロードに付帯する MAC とそれを受信したデバイスの MAC が異なることによるペイロードの破棄であり，ここでの読み取りは，プロセスによるペイロードの読み取りである．

我々は，「滞在ノード，処理の種別，プロセス名」を節とする木によるトレースの指定を提案する．ただし，プロセス名はペイロードを受け取るプロセスの名前を表す文字列であり，処理の種別が D のとき以外は空文字列となる．そして，根から葉までの経路は，ペイロードの滞在順に節が構成される．木構造とする理由は，リピータハブによりペイロードのコピーがブロードキャストされるためである．

図 3 は，六つのノードから構成されるネットワークである．角丸四角形がノードを表し，その内側の文字列の書式は「ノード種 ノード ID」である．そして，角丸四角形同士を結ぶ線はノード間のリンクを表す．送信元ノードを n0，宛先アドレスを 192.168.1.1（n5 の IP アドレス），通信プロトコルを ICMP，ペイロードを ICMP の echo-request とする通信を考える．図 4 が，この通信によるペイロードのトレースの図解である．ペイロードが n0 から送り出され，n3 で破棄され，n5 で読み取られることを示している．

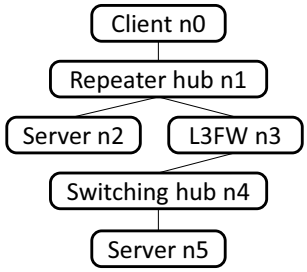

図 3 ネットワークの例

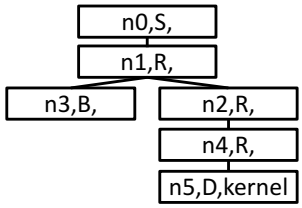

図 4 トレースの例

3.6 正解条件の導出法

本節では，正解条件の導出アルゴリズムを述べる．このアルゴリズムでは，次の関数を定義する．トレースの木の節を v とするとき，関数 mid(v) は v の第一要素を返し，関数 mtype(v) は v の第一要素が指しているノードの種別を返し，関数 ptype(v) は v の第二要素を返し，関数 pname(v) は v の第三要素を返し，関数 children(tree, v) は tree における v の子の集合を返す．トレースの木を tree，その節を v，ネットワークを nw，ノードを m，デバイスを d，通信データを p とするとき，正解条件の集合を返す関数 BuildC(tree, v, nw, m, d, p) を以下のように定義する．

S1. ptype(v)=S または ptype(v)=R であれば，表 2 の mtype(v) で ptype(v) の処理系列に対して nw, m, d, p を入力として，その実行結果として次の滞在マシン m2，次の滞在マシンへの入力デバイス d2, ネットワーク nw2，通信データ p2 を得て，条件「m2=children(tree, v)」を C に追加し S2 へ進み，そうでなければ S3 へ進む．

S2. children(v) の要素を child とするとき，すべての child

に対して BuildC(tree, v, nw2, m2, d2, p2) を実行し，その結果を C に追加後に復帰する．

S3. ptype(v)=D であれば，「mtype(v) で ptype(v) の処理系列に対して nw, m, d, p を入力として，FindProcess の実行結果が pname(v) となる」を C に追加後に関数から復帰し，そうでなく，mtype(v)=F であれば，「mtype(v) で R の処理系列に対して nw, m, d, p を入力として FFilter の実行結果が (適応, v) となる」を C に追加後に復帰し，そうでなく，mtype(v)=I であれば，「mtype(v) で D の処理系列に対して nw, m, d, p を入力として，IFilter の実行結果が (適応, v) となる」を C に追加後に復帰し，そうでなく，mtype(v)=O であれば，「mtype(v) で S の処理系列に対して nw, m, d, p を入力として OFilter の実行結果が (適応, v) となる」を C に追加後に復帰し，そうでなく，ptype(v)=B であれば，「mtype(v) で D の処理系列に対して nw, m, d, p を入力として CheckDstMAC の実行結果が失敗となる」を C に追加後に復帰する．

トレースの木を tree，その根を root として，BuildC(tree, root, NULL, NULL, NULL, NULL) を実行すると，正解条件の集合を得る．

4. プロトタイプシステム

正解条件導出機能および答案評価機能は C++，答案 D.B. は MySQL により実装されている．また，正解トレース例の指定，通信属性の指定はこれらのソースコードへのハードコーディングによる．ただし，L3FW と L2FW に関する副々処理は未実装である．

図 5 に示す演習問題は，第 2.1 節にて述べた演習内容 1〜5 の理解を確認するためのものである．問題文で指定された「通常の経路」のトレースを図 6 に示す．

図 6 に示すトレースから導出した正解条件は八つあり，そのうちの一つを図 7 に示す．各行の書式は「行番号:正解条件に関する文字列」で，説明の利便性のため著者が行番号を加えた．また，ブラケットは変数を意味し，その書式は「副々処理の種別 (副々処理の ID). その処理の出力名」である．第 2 行目にて，shb1 が FindNDev の結果のノード集合と等しいことを条件として示しており，以降の行は，まず副々処理の ID を示す行，その次にその副々処理の概要を示す行の繰り返しである．第 3 行目は ID が 12 の FindNDev で，その概要が第 4 行目に書かれている．第 6 行目にて，送信元ノード（問題文より svr1 となる）のルーティングテーブル（解答項目）を，通信属性の宛先アドレスで検索し，第 4 行目にて，その検索結果の出力デバイスの接続先（解答項目）を求め，第 2 行目にて，その結果が shb1 でなければならないことがわかる．すなわち，ある答案に対してこの正解条件が不成立となることは，その答案に誤りが存在し，その誤りの潜在範囲は先述の二つの解答項目であり，その解答項目は shb1 を導くものでなければならないことを意味する．

下図に示すネットワークトポロジーのネットワークを構築せよ．rtr2は動的経路制御をサポートしていないものとする．また，通常の経路は，「rtr1－shb3－rtr3－shb6－rtr4」，それが使えないときは，「rtr1－shb4－rtr4」，それも使えないときは，「rtr1－shb2－rtr2－shb5－rtr4」と自動的に切り替えられるようにせよ．

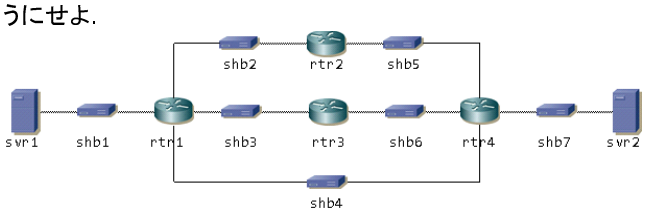

図 5 演習問題の例

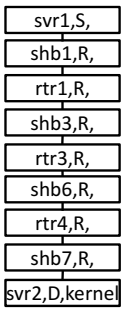

図 6 トレースの例

```
 1: VerifyNode(13)
 2: -shb1==[FindNDev(12).ノード集合]
 3:   FindNDev(12)
 4:   -[L3Routing(8).デバイス]の接続先のデバイスと
    ノードを求める
 5:     L3Routing(8)
 6:     -[APPSending(0).ノード]のルーティングテーブル
    を[InitDstIP(3).dip]で検索する
 7:       APPSending(0)
 8:       -通信条件の送信元ノード
 9:       InitDstIP(3)
10:       -通信条件の宛先アドレス
```

図 7 正解条件の例

表 3 答案の正誤判定結果

		指導者	
		正	誤
正解条件	正	4	0
	誤	0	4

5. 評価実験

正解条件による正誤評価の正当性評価を実験目的とする．図 5 の演習問題に対する答案のうち，指導者により正解と判定された四つと不正解と判定された四つにより正解条件を評価した．表 3 がその評価結果である．正解条件による判定が指導者によるものと同じ結果となっていることから，この八つの答案においては正当性があることがわかった．

6. 関連研究

文献 [4] はネットワーク構築演習システムであるが，演習環境の実現にとどまっている．文献 [5] のシステムは答案の自動評価機能を有している．しかし，正誤判定が学習者のネットワークの設定内容（答案）と正解との文字列比較に基づいているため，構築するネットワークの自由度が大幅に制限されてしまう．これに対し，提案手法は正解条件に基づいた正誤判定を行うため，この問題を解決できる．また，不成立の正解条件を活用することで誤り修正のヒントを生成できる．

7. むすび

本稿では，正解トレースから正解条件を機械的に導出する手法を提案し，その正解条件に基づいて答案を評価するシステムを提案した．ネットワーク構築演習の演習問題を一例として，それの正解トレース例を新規に定義した．そして，正解条件の導出法を C++により実装し，この正解トレース例に対する正解条件を導出した．この演習問題への八つ答案に対し，正解条件と指導者による正誤判定を比較し，両者が同じ結果となることを確認した．

今後の課題として次の二つをあげる．一つ目は，トレースと通信属性の指定方法の改善である．通常，指導者とシステム開発者は異なるため，提案法は運用上の利便性に欠ける．そこで，ソースコードから独立したファイルに指定するようにする．これにあたり，指定しやすい書式を考案したり，直感的な編集を可能にするエディタを開発したりする．もう一つは，誤り潜在範囲の表示の改善である．現在，不成立の正解条件がテキストで表示されるが，これは解答項目間の関係を捉えにくいもので，学習者は修正する解答項目を探しづらいと思われる．このため，修正する解答項目数と解答項目の修正難易度とのバランスを考慮して，解答項目間の関係を直感的に捉えるための表示を考案する．

謝辞

本研究は JSPS 科研費 23500084 および 25750082 の助成を受けたものです．

参考文献

[1] Yuichiro Tateiwa, Takami Yasuda, Shigeki Yokoi, “A System for Providing A Training Environment for Linux Networks using Virtual Environment Softwares,” IJCSNS International Journal of Computer Science and Network Security, vol.6, no.8A, pp.166-173, 2006.

[2] The User-mode Linux Kernel Home Page, http://user-mode-linux.sourceforge.net/.

[3] 中松智昭，立岩佑一郎，片山喜章，高橋直久，“仮想マシンネットワークを用いた初学者向け IP ネットワーク構築演習の自動評価システムの実現”，電子情報通信学会教育工学研究会技術研究報告，pp.169-174，2011.

[4] Muhammad Wannous, et al; NVLab, a Networking Virtual Web-Based Laboratory that Implements Virtualization and Virtual Network Computing Technologies, IEEE Trans. on Learning Technologies, Vol.3, No.2, pp.129-138, 2010.

[5] Nobukazu Iguchi, et al; Development of Hands-on IP Network Practice System with Automatic Scoring Function, Proc. of 2013 Seventh International Conference on Complex, Intelligent, and Software Intensive Systems, pp.704-709, 2013.