社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2004－OS－96　(17)
2004／6／18

AnyBed: クラスタ環境に依存しない実験ネットワーク構築支援機構の設計と実装

鈴木 未央 †　　櫨山 寛章 †　　門林 雄基 †

† 奈良先端科学技術大学院大学 情報科学研究科

要 旨

ノード数に伴う物理資源割り当ての難しさ，設定項目数の多さにより，クラスタ環境上への実験ネットワーク構築は困難である．また，ソフトウェア・ハードウェア毎の記述形式の違いにより，別環境への設定ファイルの再利用が難しい．本稿で提案する実験ネットワーク構築支援機構では実験ネットワークトポロジを論理トポロジと物理トポロジに分離した XML 記述を用いる．これら二つの XML 記述を基に支援機構はクラスタ環境の差異によらず論理トポロジを反映する．また本稿では実装の評価を通して本提案の有用性を示す．

AnyBed: Design and implementation of experimental network building system with independence from specific cluster environments

Mio SUZUKI†　　Hiroaki HAZEYAMA†　　Youki KADOBAYASHI†

†Graduate School of Information Science, Nara Institute of Science and Technology

Abstract

Complexity of the physical resource assignment and numerous configurations makes it difficult for an experimenter to build an experimental network. Also, the differences of description among configuration files used on each software and hardware makes it difficult to reuse these files. In this paper, we propose the AnyBed: the experimental network building system using the logical and physical network topologies which are described in XML. Mapping these two topologies, AnyBed can build intended logical network topology on any PC clusters. We have also evaluated an implementation of AnyBed using two distinct clusters.

1　はじめに

近年，インターネットに関する研究では，実運用されているネットワーク以外の環境で検証・評価を行う手法が利用されている．その中の一手法として，計算機とレイヤ 2 スイッチを用いて，実運用されているネットワークのエミュレーションを行い，そのネットワーク上で実験を行うものがある．このようなネットワークエミュレーションを行う環境はネットワークエミュレーションテストベッドと呼ばれる．近年，計算機の低価格化・省スペース化により，ネットワークエミュレーションテストベッドを用いて大規模なネットワーク実験を行うことが可能となっている．

ネットワークエミュレーションテストベッドにおける一般的な実験手順は次の通りである．

第一に，実施しようとする実験を計画する．実験の目的により目標となるレイヤ 3 の実験ネットワークトポロジを決定する．また，実験ネットワークトポロジ上のどのノードにおいてプログラムを動かし，

結果を収集するかを決定する．

第二に，実験計画に基づき実験ネットワークトポロジのノードに対して，ネットワークエミュレーションテストベッドに存在する物理的なノードを割り当てる．その後，実験ノードのネットワークインタフェイスに対して物理的なネットワークインタフェイス，IP アドレスを割り当てる．最後に，実験ネットワークトポロジ中のサブネットに対し VLAN とサブネットアドレスを割り当てる．この際，物理配線，ネットワークインタフェイス数・帯域や CPU 能力といったノードの能力を考慮し，的確な割り当てを行う必要がある．

第三に，実際のノード，レイヤ 2 スイッチ設定を行う．各ノードに対してネットワークインタフェイス，経路制御，名前解決の設定を行い，レイヤ 2 スイッチに対して VLAN の設定を行う．このことにより，実験用ネットワークを構築する．また，実験に必要なプログラムを各ノードに配布し，設定を行う．そして，各ノードでプログラムを動作させ，実験を行う．この際，実験の種類によっては，各ノードでのプログラム実行の時間，順序を考慮する必要がある．プログラムの実行終了後，実験結果の収集を行う．実験結果とは，実験に用いるプログラムの出力，実行結果である．実験結果は，各ノードのハードディスクに保存されるか，syslog によりリモートホストに転送される．最後に実験の後処理を行う．各ノードやレイヤ 2 スイッチに対して行った設定を初期状態に戻す．また，各ノードのハードディスクに書き込んだプログラム，データの消去を行う．

上述した手順で実験を行う際に実験者は実験ネットワークを構築するためにクラスタの各ノードとレイヤ 2 イーサネットスイッチに対して設定を行う必要がある．しかし，ノード数が増えるにしたがって，この設定に要する作業量が増加し，実験者の負荷となる．例えば，我々が過去に大規模ネットワークエミュレーションテストベッドの StarBED[1] において行った IP Traceback 実験 [2] では，64 台の PC クラスタを用いて実インターネットの BGP トポロジを再現するために，延べ 4 日を費やした．

また，特定の PC クラスタ環境において作成した設定ファイルはその PC クラスタ環境のハードウェア構成に依存している．このため，ハードウェア構成の異なった PC クラスタ環境において，同一トポロジで実験を行う際に過去に使用した設定ファイルの再利用が困難である．大規模なネットワークエミュレーションテストベッドにおいては，多人数で同時に PC クラスタを共有して使用する．このため，実験日により利用できる PC クラスタが変化し，ハードウェア構成が異なる PC クラスタを利用しなければならない場合がある．このような場合を考えると，過去に使用した設定ファイルを再利用出来ることが必要である．

本研究では，これらの問題点を解決するためにクラスタ環境での実験ネットワーク構築支援機構 AnyBed を提案する．

AnyBed は第一に資源割り当てを自動化により実験者の作業量と人的ミスを減らし，短時間での実験ネットワーク構築を目的とする．第二に実験に利用する PC クラスタのオペレーティングシステム，ハードウェア構成が異なる場合でも，実験ネットワークトポロジの再利用性を確保することを目的とする．

本論文においては，まず 2 章で本研究が提案する実験ネットワーク構築機構である AnyBed の設計について述べ，3 章で今回実装を行った部分である論理トポロジへの資源割り当て機構の詳細について述べる．その後，4 章において実装した機構の評価を行い，その結果と考察を述べる．5 章においては関連研究について述べる．6 章ではまとめと今後の課題について述べる．

2 AnyBed の設計

AnyBed 全体の設計を図 1 に示す．AnyBed は三つの層から構成される．一つめは実験ネットワーク構築のための情報を集める層であるデータ収集層 (designing layer)，二つめは資源割り当てを行う資源割り当て層 (assigning layer)，三つ目は設定ファイルを配布し，実際の実験ネットワークを構築する設定反映層 (reflecting layer) である．各層を構成するコンポーネントは容易に交換可能である．データ収集層と資源割り当て層のデータ交換には XML で記述されたファイルを用いる．

AnyBed では実験ネットワークのトポロジを論理トポロジと物理トポロジに分離する．論理トポロジは実験ネットワークのレイヤ 3 トポロジを記述する．このため，論理トポロジは特定のクラスタ環境には依存しない．一方，物理トポロジにはクラスタ固有の情報である物理ノード，配線，ネットワークインタフェイスの帯域などの情報を記述する．

実験を行う際には，物理トポロジを元に論理トポロジに対して資源割り当てを行い，その割り当ての結果生成された設定ファイルをクラスタ環境の各ノード・レイヤ 2 スイッチに配布して設定を行い，実験ネットワークが構築される．

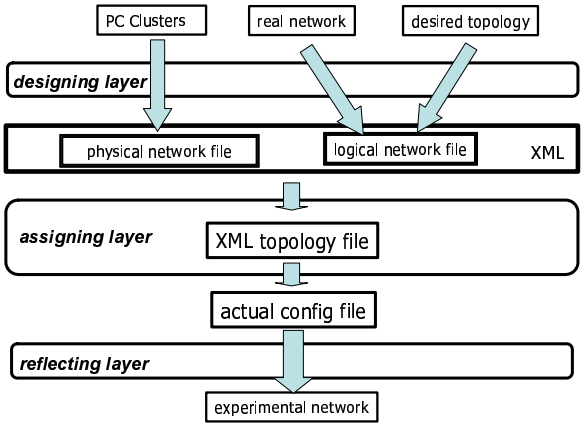

図 1: AnyBed 概念図

2.1 実験ネットワーク構築の流れ

実験者が AnyBed を用いて実験ネットワークを構築する際の流れを図 1 を参照して示す.

1. 実験者は論理トポロジ (logical network file) を作成する.
2. 作成した論理トポロジとあらかじめクラスタ毎に準備されている物理トポロジ (physical network file) から資源割り当て機構を用いて XML で記述された XML トポロジ (XML topology file) を生成する.
3. XML トポロジを元にノードの種類により異なる実設定ファイル (actual config file) を生成する.
4. 実設定ファイルを各ノードに配布し，設定を反映する.

このように，実験者は論理トポロジファイルを作成し AnyBed に与えるだけで，実験ネットワーク構築のための大半の作業は AnyBed 自動的にが行ってくれる.

2.2 物理トポロジと論理トポロジの記述

物理トポロジには機器の物理的接続状況と可能な能力を記述する．現在のところ，ノードの能力として記述できるのはそのノードが持つインタフェイスの 802.1q 対応の有無である．物理トポロジの構造は次の通りである．まずノードの集合を表現する <nodes> 要素がある．<nodes> 要素はその子要素としてノードを表現する <node> 要素を持つ．<node> 要素は子要素としてノードが持つネットワークインタフェイスを表現する <interface> 要素を持つ．<interface> 要素はその子要素として，物理的リンクを表現する <link> 要素を持つ.

```
<nodes>
    <node name="mc12" os="FreeBSD">
        <interface name="bge0" bandwidth="1000"
                dot1q="yes" purpose="management"
                managementip="172.16.1.12">
        </interface>
        <interface name="bge1" bandwidth="1000"
                dot1q="yes" purpose="experiment">
           <link tonode="mc1-sw2" toint="ethernet 1/2"/>
        </interface>
        <interface name="bge2" bandwidth="1000"
                dot1q="yes" purpose="experiment">
           <link tonode="mc1-sw2" toint="ethernet 1/7"/>
        </interface>
    </node>
</nodes>
```

図 2: 物理トポロジの例

```
<nodes>
    <node name="NodeA">
        <interface name="NodeA-Int1">
            <network name="Net1"/>
        </interface>
        <interface name="NodeA-Int2">
            <network name="Net2"/>
        </interface>
    </node>
</nodes>
```

図 3: 論理トポロジの例

物理トポロジの例を図 2 に示す．この例では mc12 という名前のノードが bge0 という名前の管理用ネットワークインタフェイスと bge1，bge2 という名前の 2 つの実験用ネットワークインタフェイスを持ち，実験用ネットワークインタフェイスはそれぞれ mc1-sw2 というスイッチの ethernet 1/2，ethernet 1/7 という名前のポートに接続されている.

論理トポロジは実験者が構築しようとする実験ネットワークの論理的トポロジを記述する．論理トポロジの構造は次の通りである．まずノードの集合を表現する <nodes> 要素がある．<nodes> 要素はその子要素としてノードを表現する <node> 要素を持つ．<node> 要素は子要素としてノードが持つネットワークインタフェイスを表現する <interface> 要素を持つ．<interface> 要素はその子要素として，ネットワークインタフェイスが属するネットワークを表現する <network> 要素を持つ.

論理トポロジの例を図 3 に示す．この例では NodeA が NodeA-Int1, NodeA-Int2 というインタフェイスを持ち Net1, Net2 というネットワークに属している.

2.3 データ収集層

データ収集層はハードウェアに関連する情報や，実際に運用されているネットワークからトポロジ情報を収集する層である．ハードウェアに関する情報とは，具体的には PC クラスタノードの MAC アドレス，ネットワークインタフェイスの情報である．トポロジ情報とはネットワークインタフェイスとレイヤ 2 スイッチとの接続情報，実運用されているレイヤ 3 ネットワークの情報である．これらの情報を収集し，収集した情報を元に物理トポロジと論理トポロジを作成する．

2.4 資源割り当て層

資源割り当て層ではデータ収集層により生成された 3 つのファイルを元に，実験者の意図する論理トポロジへ資源割り当てを行い，実際の実験ネットワークを構築する機構について説明を行う．資源とは，物理ノード，物理ネットワークインタフェイス，IP アドレス，VLAN，帯域である．資源割り当て層は以下で述べる 3 つの機構からなる．

第一の機構は dispatcher である．本機構は，物理トポロジ，論理トポロジを入力として取り，XML トポロジを出力する．

本機構の入力ファイルである物理トポロジは node - interface - link という node を根とした木構造を持ち，論理トポロジは node - interface - network という node を根とした木構造を持つ．このため，dispatcher により link と network を対応付けることにより，適切に論理トポロジにおけるノードを物理的なノードに対応付けることができる．

第二の機構は実設定ファイル生成機構である．本機構は，XML トポロジを入力として取り，実設定ファイルを生成する．

2.5 設定反映層

設定反映層では資源割り当て層の機構により生成された実設定ファイルをクラスタの各ノード，レイヤ 2 スイッチに反映させる．

設定反映層唯一の機構は実設定ファイル反映機構である．本機構は実設定ファイルを入力として取り，各ノード，レイヤ 2 スイッチに設定を反映させる．

3 AnyBed の実装

本論文では，AnyBed のうち，核となる資源割り当て層について実装を行った．資源割り当て機構の実装を行った理由は，実験手順のうち，資源割り当てに関する作業量が最も多く，この部分を自動化することにより飛躍的に実験ネットワーク構築の作業効率が向上するためである．

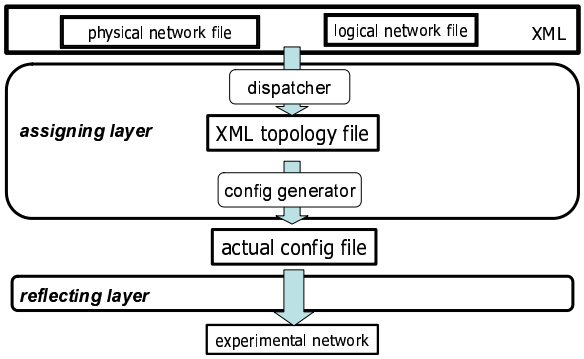

図 4: AnyBed 機構図

AnyBed のうち資源割り当て層に存在する機構は図 4 の通り資源割り当て機構 (dispatcher) と実設定ファイル生成機構 (config generator) である．

3.1 資源割り当て機構の実装

本節では，資源割り当て機構の実装について述べる．

実装のための言語には，Ruby を使用した．この理由は，XML やノードの設定ファイルといったテキストファイルの処理を多く行うため，テキストファイルの処理が容易に実装できるスクリプト言語を用いたほうが効率よく実装できると考えたためである．

実装を行った資源割り当て機構は，物理トポロジと論理トポロジを読み込み，XML トポロジを出力せずに直接実設定ファイルを出力する．

資源割り当て機構の現在のバージョンが動作するためには以下に示す条件が必要である．

- 各ノードは必ず 1 つ以上の LAN スイッチに繋がっている．
- 各レイヤ 2 スイッチは VLAN 機能を備える．
- 各レイヤ 2 スイッチは必ず 1 つ以上の 802.1q 対応のリンクで繋がっている．
- 各レイヤ 2 スイッチには共通の VLAN ID が設定できる．

この条件を置くことにより，実装を簡略化することにした．

資源割り当て機構の処理内容について次に述べる．資源割り当て機構は，論理トポロジの内容を読み込み，データを格納する．次に，物理トポロジの内容

```
# 実験ネットワークトポロジにおけるノードの配列
BedNodes nodes
# 物理ノードの配列
BedPhysicalNodes pnodes
# 最も残り帯域の多い物理インタフェイス
BedPhysicalInterface maxleftbwpint

# 論理ノードをネットワークインタフェイスの数が多い順に並べる
sort(nodes)
# 物理ノードをネットワークインタフェイスの帯域の合計が多い
順に並べる
sort(pnodes)
# 論理ネットワークインタフェイスを帯域が多い順に並べる
sort(nodes.int)
# 物理ネットワークインタフェイスを帯域が多い順に並べる
sort(pnodes.int)

nodes.each {|n|
  p = pnodes.shift
  n.int.each {
    average_bandwidth = p.int.bandwidth_total /
        n.int.bandwidth_total
    p.int.bandwidth_left = p.int.bandwidth_total

    maxleftbwpint = search_max_max_bandwidth_left(p.int)
    assign_vlan(maxleftbwpint, n.int)
    maxleftbwpint.bandwidth_left -= average_bandwidth
  }
}
```

図 5: ノード・インタフェイス割り当てアルゴリズム

を読み込み，同様にデータを格納する．次に，実験ネットワークトポロジ構築に用いるネットワークに対して VLAN の ID とサブネットアドレスを割り当てる．このサブネットアドレスは 10.0.0.0/8 のプライベートアドレス空間から割り当てる．その後，そのサブネットに属するネットワークインタフェイスに対してアドレスを割り当てる．そして，物理ノード，実験インタフェイスと物理インタフェイスの対応付けを行う．その後，レイヤ 2 イーサネットスイッチ間の接続に用いられているポート検出し，必要な VLAN 設定を計算する．

最後に，対応づけられた結果を元に，実設定ファイルを出力する．実設定ファイルの内容は，FreeBSD 上の rc.conf, hosts, zebra.conf, ospfd.conf とレイヤ 2 イーサネットスイッチの設定を記したファイルである．本実装で対応したレイヤ 2 イーサネットスイッチは，DELL 社製 PowerEdge 1655MC Integrated Switch, ExtremeNetworks 社製 Summit48, Summit5i である．

実験ネットワークトポロジのノードと物理ノードの対応付け，ノードのインタフェイスと物理ノードの物理インタフェイスの対応付けに関しては，本プロトタイプ実装では，簡単のために制約条件をインタフェイスの帯域のみに絞り，実験ノードのインタフェイスに対して，物理ノードのインタフェイスが持つ帯域を少ない計算量にで公平に割り当てるアルゴリズムとした．このアルゴリズムを図 5 に示す．

表 1: 均質な実験用 PC クラスタの構成

CPU	Intel Pentium3 1.4GHz
メモリ	1024MB
NIC	Broadcom BCM5703X 2 台
レイヤ 2 スイッチ	DELL PowerEdge1655MC Switch

4 AnyBed の検証・評価

4.1 再利用性・作業量低減に対する検証

最初に，均質な PC クラスタ環境と異なったハードウェアによる不均質な PC クラスタ環境において動作するか検証を行った．つまり，一つの実験ネットワークトポロジファイルをハードウェア構成の異なる PC クラスタ環境に再利用できるということを確認した．同時に，資源割り当て機構の実行，実設定ファイル反映機構による実設定ファイルの配布，反映にかかる時間を計測した．この時間計測により短時間で実験ネットワークトポロジが構築できるかどうかを検証した．また，構築された実験ネットワークトポロジの正当性を確認することにより，実装した機構が設計通りに動作しているかを検証した．そして、実験者が記述したファイルのサイズと資源割り当て機構により生成されたファイルの合計サイズを比較することにより，実験者の設定ファイル記述に関する作業量が軽減されていることを確認した．

4.1.1 検証環境

異なるクラスタ環境において AnyBed が正常に動作するかを検証するため，構成するハードウェアが均質なクラスタ環境と不均質なクラスタ環境の二種類を準備した．

均質なクラスタ環境のハードウェア構成を表 1 に示す．

均質なクラスタによる検証実験は，表 1 のハードウェア構成を持つ DELL 社製ブレードサーバを 17 台用い，1 台を実験用サーバとし，残りの 16 台を実験ネットワーク構築用ノードとして行った．実験用サーバ，実験ネットワーク構築用ノードは，6 台のレイヤ 2 スイッチにより相互接続されていた．

均質なクラスタでは config reflector の実装として PXE boot[3] による OS 起動・設定ファイル配布機構を元に改良した機構を利用する．この配布機構は PXE により FreeBSD の TFTP サーバ上に存在する kernel image と disk image を読み込んで各ノードを起動させ，起動時に各ノードが FTP サーバ上に存在する設定ファイルを取得するという仕組みである．この機構を用いることにより各ノードを

表 2: 不均質な実験用 PC クラスタの構成

1 ～ 3	CPU	Intel Pentium3 450MHz
	メモリ	256MB
	NIC	Intel Pro 10/100B/100+
		3Com 3c905B-TX
4 ～ 7	CPU	Intel Pentium3 900MHz
	メモリ	256MB
	NIC	3Com 3c905B-TX
		Netgear GA620
レイヤ 2 スイッチ		FoundryNetworks EdgeIron4802F
		ExtremeNetworks Summit48
		ExtremeNetworks Summit5i

起動させ，各ノードに設定ファイルを配布し設定を行うことにより，実験ネットワークトポロジを構築する．

不均質クラスタ環境の各ノードのハードウェア構成を表 2 に示す．

不均質なクラスタ環境のノードの内，1 から 3 台目のノード と 4 から 7 台目のノードではネットワークインタフェイスカードの種類が異なるため，オペレーティングシステムでのネットワークインタフェイス名と利用できる帯域が異なる．

不均質なクラスタ環境においては，ノードのネットワークインタフェイスの種類によっては PXE Boot が不可能な場合がある．このため，資源割り当て機構を動作させ，実設定ファイルを NFS により公開する管理サーバを準備し，各実験ノードが NFS により取得した個々の設定ファイルを個々のハードディスクに複製し，設定の反映を行う．

4.1.2 検証内容

これらの二種類のクラスタ環境で同一の論理トポロジを用いた場合，同様の実験ネットワークが構成されるかを検証した．検証に用いる論理トポロジは 7 個のルータ間でフルメッシュリンクを構成するトポロジを用いた．二種類のクラスタ環境で同様の実験ネットワークトポロジが構築できているかどうかを検証するために，双方の環境で各ノードの OSPF ルーティングテーブル内容を比較し，同様のネットワーク経路が存在するかどうかを調べた．比較するルーティングテーブルの内容は，Zebra OSPF デーモンに telnet でログインし，show ip ospf route コマンドにより取得した．

この検証を行う際，実験ネットワーク構築を開始してから完了するまでの時間を計測する．計測は 10 回行い，その平均値を得た．更に、両クラスタ環境において，実験者が記述した論理トポロジ, 物理トポロジのファイル数・サイズと資源割り当て機構により生成された 実設定ファイルの合計ファイル数・サイズを比較した．

表 3: 検証結果

	均質環境	不均質環境
AnyBed 設定ファイル数	2	2
AnyBed 設定ファイルサイズ	10671 バイト	6549 バイト
設定ファイル数	31	30
設定ファイルサイズ	33957 バイト	39191 バイト
平均実験ネットワーク構築時間	137 秒	135 秒

4.1.3 検証結果

検証結果を表 3 にまとめる．均質・不均質どちらの環境においても AnyBed を利用せず手作業で実験ネットワークを構築する場合は表 3 に示されている設定ファイル数・サイズのファイルを手作業やスクリプトを用いて作成せねばならない．

ファイルサイズでは，どちらの環境においても論理トポロジのサイズが 3377 バイト，物理トポロジのサイズが均質環境では 7294 バイト，不均質環境では 3172 バイトであった．これに対し，実設定ファイルの合計サイズは均質環境では 33957 バイト，不均質環境では 39191 バイト であった．このことから，AnyBed を用いることにより実験者が実験ネットワーク構築するために必要とされる記述量が減少していることがわかる．

ファイル数については論理トポロジは 1 ファイルからなり，実設定ファイルは各ノード毎に rc.conf, zebra.conf, ospfd.conf, hosts の 4 ファイルと各スイッチ毎に 1 ファイルからなる．本検証でのノード数は均質，不均質環境ともに 7 ノード，スイッチ数は均質環境では 3 台，不均質環境では 2 台であった．このため，均質環境でのファイル数は $4\times7+3=31$ ファイル，不均質環境でのファイル数は $4\times7+2=30$ ファイルである．このことから AnyBed を用いることにより実験者が記述するファイル数が減少していることがわかる．

実験ネットワークの構築時間は，どちらの環境においても 140 秒以下であった．この結果は，7 台のノードにより実験ネットワークを構築する時間としては，十分短い時間であると言える．

これらの結果から明らかに実験者の作業量は AnyBed により削減されていると言える．

OSPF ルーティングテーブルの内容はどちらの環境においても同質であったことから，同質の実験ネットワークが構築できたと言える．このことから，同じ実験ネットワークトポロジを異なる PC クラスタ環境において再利用可能であると言える．

表 4: スケーラビリティ評価環境

ハードウェア	
CPU	UltraSPARCIII 900MHz × 24
メモリ	64GB
ソフトウェア	
オペレーティングシステム	Solaris 8
言語	Ruby 1.8.1
XML パーサ	REXML 2.4.8

4.2 実験ノード数に対するスケーラビリティ評価

次に，提案システムが大規模のネットワーク数，ネットワークインタフェイス数において実用に耐えうるかどうかの検証を行った．

実装を行った資源割り当て機構が，多数のノードを有する PC クラスタ環境で動作するかどうかを検証を行った．資源割り当て機構に入力するファイルの内，物理トポロジを固定し，論理トポロジ中に記述されているノード数を変化させ，資源割り当て機構の各部分における実行時間と資源割り当て機構全体におけるメモリ使用量を測定した．最大ノード数は StarBED での PC クラスタ環境を想定し，512 ノードとした．

評価用の実験ネットワークトポロジは出来る限り各ノードがフルメッシュリンクを持つ構造とした．これは，ノード数が同じ場合，フルメッシュ構造とするのが最もネットワーク数が多くなり，ネットワーク数と関連して設定量が多くなるためである．しかし，実際の実験においては VLAN の最大数制限のため，512 台の各ノードがフルメッシュのリンクを持つことはできない．よって本実験では，IEEE802.1Q での最大 VLAN 数は 4096 から管理用 VLAN を除いて，実験に利用できる VLAN 数を 4000 と想定し，その VLAN 数において出来る限り複雑なネットワークトポロジを考えた結果，ノード数が 88 台まではフルメッシュ構成とし，それ以上のノード数ではフルメッシュを構成する各ノードに対してぶら下がる形でリンクを持つノードからなる構成とした．この実験ネットワークトポロジを表 4 の示す環境においてノード数を 2 から 512 まで変化させ，実験を行った．

実験では論理トポロジの読み込みに要する実時間，物理トポロジの読み込みに要する時間，資源割り当てに要する時間，実設定ファイルの出力に要する時間を測定した．また，資源割り当て機構が使用するメモリ量を測定した．計測した結果を図 6，図 7，図 8 に示す．

512 ノードでの実行時間は 579 秒であり，そのうちアルゴリズム処理部は 2 秒であった．また，512 ノードでのメモリ使用量は 211 メガバイト であった．

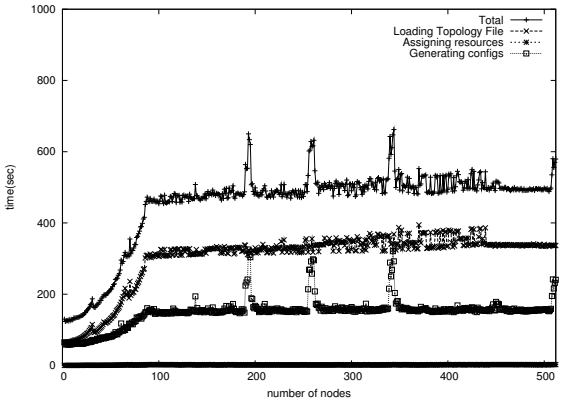

図 6: ノード数に対する資源割り当て機構の処理時間

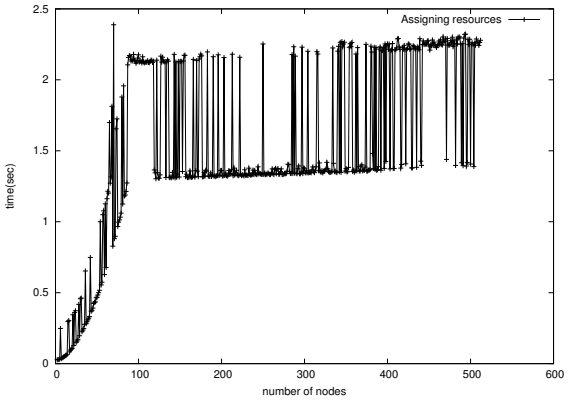

図 7: ノード数に対する資源割り当て機構の処理時間 (アルゴリズム処理部)

図 6，図 7，図 8 のグラフでは 88 ノードまでは値が急激に増加している．これは評価用の実験ネットワークトポロジが 88 ノードまではフルメッシュ構成であるためネットワーク数が急激に増加するが，それ以上のノード数では各ノードに対してぶら下がる形のリンクになるため，88 ノード以上ではネットワーク数が急激に増加しないためであると考えられる．

5 関連研究

ネットワークエミュレーションテストベッドにおける資源割り当てに関する関連研究にはユタ大の NetBed[4] で行われている研究 [5] やデューク大の ModelNet[6] がある．これらの研究は特定のクラスタ環境に対して実現したい実験ネットワークトポロジを与え，適切な資源割り当てを行い，実験ネット

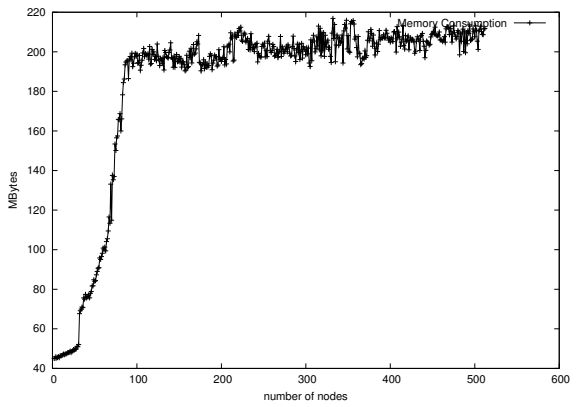

図 8: ノード数に対する資源割り当て機構のメモリ使用量

ワークを構築するという手法である．それに対して，本研究では様々な構成のクラスタ環境での利用を前提とし，そのクラスタ環境上で同一の実験ネットワークを構築することを目的の一つとしている．このことにより関連研究とは研究の方向性が異なる．

6 おわりに

本論文では，ネットワークエミュレーションテストベッドでの実験ネットワーク環境構築に関する問題点を指摘し，その問題点を解決する実験ネットワーク構築支援機構である AnyBed を提案した．そして AnyBed のうち，資源割り当て層とを実装し評価することにより，実験ネットワークトポロジの再利用性を確保し，資源割り当てを自動化することで短時間で実験ネットワークを構築可能なことを示した．

よって，本提案によりネットワークエミュレーションテストベッドでの実験ネットワーク構築の省力化が可能となったと言える．

今後の課題として，未実装となっている各機構の実装，資源割り当てアルゴリズムの改良，アプリケーション設定ファイルへの対応，クラスタが物理的に分散している場合の実験ネットワークの構成方法などについて考えていく．

参考文献

[1] 通信・放送機構 北陸 IT 研究開発支援センター. http://http://www.hokuriku-it.tao.go.jp/.

[2] 大江将史, 櫨山寛章, 門林雄基. 階層型 IP トレースバックの評価. 電子情報通信学会 技術研究報告 インターネットアーキテクチャ，第 103 巻, 2003 年 7 月.

[3] Intel Corporation, Preboot Execution Environment (PXE) Specifiction Version 2.1, September 1990.

[4] Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar. An integrated experimental environment for distributed systems and networks. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation*, December 2002.

[5] Robert Ricci, Chris Alfeld, and Jay Lepreau. A solver for the network testbed mapping problem. In *ACM SIGCOMM Computer Communications Review 33(2)*, 2003.

[6] Amin Vahdat, Ken Yocum, Kevin Walsh, Priya Mahadevan, Dejan Kostic, Jeff Chase, and David Becker. Scalability and accuracy in a large-scale network emulator. In *Proceedings of 5th Symposium on Operating Systems Design and Implementation (OSDI)*, Dec. 2002.