社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

分散型モデル予測制御にもとづくスケーラビリティを有する仮想ネットワーク埋め込み手法

河島　滉太†　大歳　達也†　大下　裕一†　村田　正幸†

† 大阪大学 大学院情報科学研究科
〒 565–0871 大阪府吹田市山田丘 1–5
E-mail: †{k-kawashima,t-otoshi,y-ohsita,murata}@ist.osaka-u.ac.jp

あらまし　ネットワーク仮想化技術が進展し、サービスの要求に応じて仮想ネットワークの埋め込みを行う手法が議論されるようになった。仮想ネットワーク埋め込みでは、各サービスプロバイダーからの要求に応じて、当該サービスの提供に必要な仮想ネットワークを埋め込む。従来、仮想ネットワーク埋め込みの問題は、必要な計算機資源と通信帯域が既知で変化しないものとして、最適な埋め込み箇所を求める手法の検討が進められてきた。しかしながら、埋め込まれた仮想ネットワークにおいても、対応するサービスの需要の変動が生じ、必要な計算機資源や通信帯域が変化する。この変化に対応する手法として、モデル予測制御を適用し、各サービスの需要変動といった環境変動に追随する安定的な仮想ネットワーク埋め込みを行う手法が考えられる。しかしながら、この手法では、埋め込む仮想ネットワーク数が大きくなれば、短い周期での制御ができなくなり、環境変動へ追随するのに時間がかかる。そこで、本稿では、分散型モデル予測制御にもとづく仮想ネットワーク埋め込み手法を提案する。また、シミュレーションにより、提案手法が、環境変動に追随しつつも制御時間を短時間に抑制可能な手法であることを示す。
キーワード　ネットワーク仮想化、仮想ネットワーク埋め込み、モデル予測制御、分散型モデル予測制御

Scalable Virtual Network Embedding based on distributed model predictive control

Kohta KAWASHIMA†, Tatsuya OTOSHI†, Yuichi OHSITA†, and Masayuki MURATA†

† Graduate School of Information Science and Technology, Osaka University
Yamadaoka 1–5, Suita-shi, Osaka, 565–0871 Japan
E-mail: †{k-kawashima,t-otoshi,y-ohsita,murata}@ist.osaka-u.ac.jp

Abstract　Multiple virtual network can be accommodated on a single substrate network by using the network virtualization technologies. To accommodate the multiple virtual networks, where to embed the virtual networks is an important problem. This problem is called virtual network embedding (VNE). In VNE, the virtual machines are embedded depending on the request from each service provider. Most of the existing VNE methods assume that required resources are static. However, the fluctuations of service may occur, which leads to the change in the required resources. In this paper, we propose the flexible VNE method that follows the changes in the required resources by applying the model predictive control (MPC) to VNE. The VNE based on the MPC considers the future changes in required resources decreasing the impact of the prediction errors by correcting the prediction. However, the MPC takes a large time to calculate the optimal inputs as the number of the virtual networks becomes large. Therefore, we apply the distributed MPC (DMPC), where multiple agents are deployed, and each of them calculates the sub problem of the problem to be solved. We evaluated our method by simulation. The results indicate that the VNE based on DMPC can follow the fluctuations of service and reduce time to calculate.
Key words　Network Virtualization, Virtual Network Embedding, Model Predictive Control, Distributed Model Predictive Control

1. はじめに

ネットワーク上のサービスを柔軟に提供する技術として、ネットワーク仮想化技術 [1] の研究が進められている。ネットワーク仮想化では、ネットワークを管理する ISP の役割を、物理インフラを管理する InP (Infrastructure Provider) と仮想ネットワーク (VN;Virtual Network) を構築する SP (Service Provider) とに分離させ、異なる SP 同士が物理ネットワークを共有しながら仮想ノードと仮想リンクからなる VN をそれぞれ独立して構築することが可能となる。

ネットワーク仮想化において、各 VN の埋め込み位置は、各 VN の性能や、収容可能な VN 数に大きな影響を与える。各 VN が十分な性能を確保するためには、十分な資源割り当てが可能な箇所に仮想ノードや仮想リンクを配置することが必要となる。しかしながら、仮想ノードや仮想リンクに過剰な資源の割り当てを行うと、収容可能な VN 数が低下する。

そこで、VN の埋め込み位置を制御する仮想ネットワーク埋め込み (VNE；Virtual Network Embedding) と呼ばれる手法に関する研究が進められている [2]〜[4]。これらの手法では、現時点で要求される仮想資源のみに合わせて当該 VN の収容位置を決定する。しかしながら、実際のサービスでは、サービス需要は時々刻々と変動し、それに合わせて必要な仮想資源も時々刻々と変動する。この時々刻々変動する需要に合わせて VN の収容位置を変更することで、より多くのサービスを物理ネットワークで処理することが可能であると考えられるが、仮想マシンの移動には時間がかかり、必要な仮想資源が判明してから仮想マシンの移動を開始したのでは、十分な性能を達成できない。

このような問題を解消する方法として、VN 収容時に当該 VN が将来に要求する仮想資源を予測し、その予測も考慮したうえで収容位置を決定する手法が考えられる。予測と連携した制御としては、モデル予測制御 (MPC;Model Predictive Control) [5] と呼ばれる手法の検討が進められており、情報ネットワーク分野においてもトラヒックエンジニアリングに MPC を適用した手法 [6] が検討されている。しかしながら、VN の収容位置を集中制御によって決める問題は、収容する VN 数や、埋め込みが必要なノード数が増えれば、膨大な計算時間が必要となる。

そこで、本稿では、分散配置された複数のエージェント間の連携により行う分散型モデル予測制御 (分散型 MPC;Distributed Model Predictive Control) [7] を適用する。分散型 MPC では、制御対象であるシステムを複数のエージェントで管理し、各エージェントが次ステップの入力を決定する。また、エージェント間が連携をとることによって、制御系全体としての目的に沿う入力を取得する。本研究では、収容する VN ごとにエージェントを割り当て、各エージェントが当該 VN の収容位置を MPC によって決定する手法を検討する。分散型 MPC を適用することによって、「将来の予測をふまえた制御」と「短い時間で予測の補正と制御」を繰り返す、というモデル予測制御の利点を持ちつつも、最適化問題の対象を 1 つの VN とすることで最適化問題を解くための計算時間を抑えることが可能になる。

本稿の内容は以下の通りである。まず、2 章で本稿における仮想ネットワーク埋め込み問題の定式化を行う。次に、3 章では 2 章の定式化に対し、仮想ネットワーク埋め込み手法へ MPC を適用させたものについて述べ、4 章では分散型 MPC にもとづく仮想ネットワーク埋め込み手法について述べる。そして、5 章では、MPC を適用した際の評価と、分散型 MPC を適用した際の評価を行い、最後に、6 章でまとめと今後の課題について述べる。

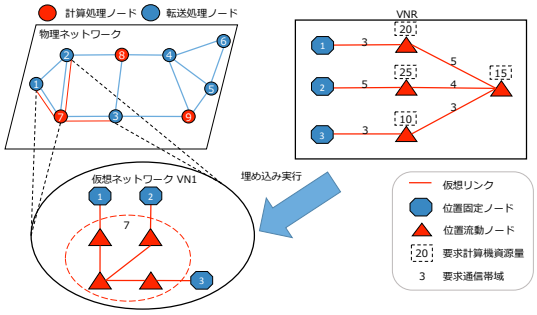

図 1　仮想ネットワーク埋め込みモデル

2. 仮想ネットワーク埋め込み問題の定式化

2.1 物理ネットワークのモデル化

本稿における仮想ネットワーク埋め込みモデルを図 1 に示す。VN を埋め込む対象は、転送処理ノード・計算処理ノード・リンクからなる「物理ネットワーク」である。ここで、転送処理ノードの集合を N_t^p、計算処理ノードの集合を N_f^p、リンクの集合を L^p とする。各転送処理ノード $n \in N_t^p$ は、トラヒックを他のノードへ転送する機能のみを持ったノードであり、計算機資源の提供はできない。各計算処理ノード $n \in N_f^p$ は計算機資源を提供することが可能で、仮想マシンの埋め込みが可能なノードである。ここで、転送処理ノードと計算処理ノードの和集合を物理ノードの集合 $N^p = N_t^p \cup N_f^p$ として定める。各物理ノード $n \in N^p$ にはそれぞれ番号 $i (1 \leqq i \leqq |N^p|)$ が与えられ、n_i は i 番目の物理ノードであることを表す。リンクは物理ノード間を結ぶものであり、仮想リンクの埋め込みが可能である。各計算処理ノード、各リンクには、提供可能な資源量が定まっており、計算処理ノード $n \in N_f^p$ で提供できる計算機資源総量は U_n、リンク $l \in L^p$ で提供できる帯域総量は B_l として与えられる。

2.2 VN の要求設定

物理ネットワークへ収容する VN の要求 (VNR;Virtual Network Request) について説明する。VNR は位置固定ノード・位置流動ノード・仮想リンクからなる「仮想ネットワーク」である。ここで、位置固定ノードの集合を N_k^v、位置流動ノードの集合を N_r^v、仮想リンクの集合を L^v、位置固定ノードと位置流動ノードの和集合を仮想ノードの集合 $N^v = N_k^v \cup N_r^v$ とする。位置固定ノードは、当該位置固定ノードが指定する物理ノード以外に収容することができない。それに対して、位置流動ノードは、仮想マシンで動作する当該 VN のサービスを提供するノードであり、計算処理ノードに収容することができる。ノード間のリンクは、仮想ノード間の連携の際に必要な通信を表す。各位置固定ノード $v \in N_k^v$ にはそれぞれ番号 $i_v (1 \leqq i_v \leqq |N^p|)$ が与えられ、番号 i_v の位置固定ノードは、番号 i で指定された物理ノード $n_i \in N^p$ に収容される。位置流動ノード $v \in N_r^v$ については、必要とする計算機資源量 u_v が与えられ、仮想リンク $l \in L^v$ については、必要とする通信帯域 b_l が与えられる。

2.3 仮想ネットワーク埋め込み手法

本稿では、制御を行う現在時刻を 0 とし、時刻 1 の仮想ノー

ド・仮想リンクの収容位置を最適化問題によって求める。本最適化問題では、入力として物理ネットワークのトポロジー情報 $SN=(N^p,L^p,P^p,G_{p,l}),U_n,B_l$ が与えられる。N^p は物理ネットワーク内のノードの集合、L^p は物理リンクの集合、P^p は物理ネットワーク上のパスの集合を表す。本最適化問題は、事前に仮想リンクの収容先候補となるパスを複数準備し、その候補の中から実際に仮想リンクの収容に用いるパスを選択する問題として定式化を行う。候補パスの集合を P^p とし、P^p の各要素 p は、当該パスが経由するリンクの集合として表される。また、パス p の始点・終点ノードはそれぞれ n_p^{start}、n_p^{end} と表す。$G_{p,l}$ はパスが利用するリンクを表す値であり、パス p がリンク l を経由する場合に 1、それ以外の場合は 0 となるように定義する。

収容する各 VN 情報として $VNR^j=(N_j^v,L_j^v)^j$ が与えられる。VNR^j は j 番目の VNR を表し、N_j^v、L_j^v はそれぞれ VNR^j における仮想ノード、仮想リンク集合を表す。ここで、収容する VN 数を J として表記する。各仮想リンク $l\in L_j^v$ については始点となる仮想ノード $v_{j,l}^{start}$、終点となる仮想ノード $v_{j,l}^{end}$ の情報と、時刻 0 で観測した、仮想リンク l を流れたトラヒック量 $b_{j,l}$ が与えられる。N_j^v は位置流動ノードの集合 $N_{j,r}^v$ と位置固定ノードの集合 $N_{j,k}^v$ の和集合であり、各位置固定ノード $v\in N_{j,k}^v$ には、$1\leqq i\leqq|N^p|$ の重複のないいずれかの番号が付与される。また、$v_i\in N_{j,k}^v$ は番号 i の位置固定ノードであることを表す。各位置流動ノード $v\in N_{j,r}^v$ について、当該位置流動ノードが時刻 0 で実際に要求した計算資源容量 $u_{j,v}$ が与えられる。

上記の入力を用い、時刻 1 に仮想ノードを収容する物理ノード、仮想リンクを収容するパスを決定する。ここで、$M_{j,v,n}^{Node}(1)$ を時刻 1 において VNj の仮想ノード v を物理ノード n に収容する場合には 1、それ以外は 0 となる変数として定義する。同様に、$M_{j,l,p}^{Link}(1)$ を時刻 1 において VNj の仮想リンク l をパス p に収容する割合を表す変数として定義し、$M_n^{Node}(1)$ を時刻 1 において計算処理ノード n に位置流動ノードが収容されている場合には 1、それ以外は 0 となる変数として定義する。また、$M_{j,v,n}^{Node}(0),M_{j,l,p}^{Link}(0)$ については、制御時に収容されている箇所に合わせて与えられるものとする。

これらの入力と変数を用い、各 VN の時刻 1 の収容位置を決定する最適化問題を以下に記す。

$$minimize\quad (1-w)\xi+w\sum_{1\leqq j\leqq J}\sum_{n\in N_f^p}\sum_{v\in N_{j,r}^v}R_{j,v,n}(1) \tag{1}$$

$$subject\ \ to$$

$$R_{j,v,n}(1)=|M_{j,v,n}^{Node}(1)-M_{j,v,n}^{Node}(0)| \tag{2}$$

$$\sum_{n\in N_f^p}M_n^{Node}(1)=\xi \tag{3}$$

$$\forall n\in N_f^p,\quad \frac{1}{\sum_{1\leqq j\leqq J}|N_{j,r}^v|}\sum_{1\leqq j\leqq J}\sum_{v\in N_{j,r}^v}M_{j,v,n}^{Node}(1)\leqq M_n^{Node}(1) \tag{4}$$

$$1\leqq j\leqq J,\forall v\in N_{j,r}^v,\ \sum_{n\in N_f^p}M_{j,v,n}^{Node}(1)=1 \tag{5}$$

$$1\leqq j\leqq J,1\leqq i\leqq|N^p|,\forall v_i\in N_{j,k}^v,M_{j,v_i,n_i}^{Node}(1)=1 \tag{6}$$

$$1\leqq j\leqq J,\forall l\in L_j^v,\sum_{p\in P^p}M_{j,l,p}^{Link}(1)=1 \tag{7}$$

$$1\leqq j\leqq J,\forall l\in L_j^v,\forall p\in P^p,\quad M_{j,l,p}^{Link}(1)\leqq M_{j,v_{j,l}^{start},n_p^{start}}^{Node}(1) \tag{8}$$

$$1\leqq j\leqq J,\forall l\in L_j^v,\forall p\in P^p,\quad M_{j,l,p}^{Link}(1)\leqq M_{j,v_{j,l}^{end},n_p^{end}}^{Node}(1) \tag{9}$$

$$1\leqq j\leqq J,\forall n\in N_f^p,\ \sum_{v\in N_{j,r}^v}M_{j,v,n}^{Node}(1)u_{j,v}\leqq U_n \tag{10}$$

$$1\leqq j\leqq J,\forall l^p\in L^p,\quad \sum_{l^v\in L_j^v}\sum_{p\in P^p}G_{p,l^p}M_{j,l^v,p}^{Link}(1)b_{j,l}\leqq B_{l^p} \tag{11}$$

ただし、本目的関数では、第一項は位置流動ノードが収容されている計算処理ノード数であり、第二項は仮想ノードの移動にかかるコスト、w は重みパラメーターである。本目的関数に合わせて制御をすることにより、時刻 1 における仮想ノード埋め込みに要する計算処理ノード数を小さくしながらも、仮想ノードの移動が必要になることを避けることが出来る。

3. モデル予測制御の VNE への適用

3.1 モデル予測制御の概要

予測と連携した制御として、制御理論分野ではモデル予測制御 (MPC) [5] と呼ばれる手法の検討が進められている。MPC は、システムの入力を決定する際に、現状態のみを考慮するのでなく有限ステップ先の将来を予測し、その予測値と目標値とのずれを最小にする入力を決定する制御方式である。ある時刻 k のときの入力を $u(k)$、出力を $y(k)$ とし、現時刻 t の状況を考える。このとき、MPC による制御方式では、現時刻 t の様々な状態から、$[t+1,...,t+h]$ 間のシステムの動作を予測する。この予測にもとづき、$[t+1,...,t+h]$ 間の予測値と目標値とのずれが最小となるような入力 $u(t+1)\sim u(t+h)$ を決定する。h ステップ先までの入力が決定されると、次はシステムへ入力を実際に投入することになるが、システムへは $u(t+1)$ のみを投入する。システムへ入力を投入することによって、システムからは新たに出力 $y(t+1)$ が得られ、この $y(t+1)$ の値はフィードバックとして以降の予測に用いる。そして、フィードバックから予測を修正し、再度適切な入力の計算を行う。この手順を短時間で繰り返すことによって、予測をふまえた制御を行いながら、予測誤差の補正を行うことが可能になり、予測した需要変動に追随しつつも、想定外の需要変動に対してもロバストな制御手法を確立することが可能になる。

3.2 MPC にもとづく仮想ネットワーク埋め込み手法

本稿における MPC を適用した仮想ネットワーク埋め込みを VNE–MPC と呼ぶ。VNE–MPC では、次の時刻だけでなく、H ステップ先までの各時刻の仮想ノード・仮想リンクの収容先を最適化問題によって求める。本最適化問題では、入力として 2.3 と同様に、$SN=(N^p,L^p,P^p,G_{p,l}),U_n,B_l,VNR^j=(N_j^v,L_j^v)$ が与えられる。また、VNR^j が時刻 t で要求する仮想資源について、各位置流動ノード $v\in N_{j,r}^v$ が要求する計算機資源容量の予測値として $\hat{u}_{j,v}(t)$、各仮想リンク $l\in L_j^v$ が要求する通信帯域の予測値として $\hat{b}_{j,l}(t)$ が与えられる。

上記の入力を用い、各時刻 t における仮想資源の収容箇所を決定する。ここで、2.3 と同様に、時刻 t における仮想資源の収容先を表す変数として、$M_{j,v,n}^{Node}(t),M_{j,l,p}^{Link}(t),M_n^{Node}(t)$ を定義する。

これらの入力と変数をもとに、各 VN の各時刻における収容

位置を計算する。本最適化問題では、以下の目的関数を与える。

$$minimize \ (1-w)\sum_{0<t\leqq H}\xi(t)+w\sum_{0<t\leqq H}\sum_{1\leqq j\leqq J}\sum_{n\in N_f^p}\sum_{v\in N_{j,r}^v}R_{j,v,n}(t) \quad (12)$$

ただし、本最適化問題では、2.3 で定めた各制約条件について、時刻 1 のみでなく各時刻 $t(1 \leqq t \leqq H)$ における制約条件を満たす範囲内での解の探索を行う。また、式 (10)、(11) の $u_{j,v}, b_{j,l}$ については、各時刻 t における予測値 $\hat{u}_{j,v}(t), \hat{b}_{j,l}(t)$ を用いる。

4. 分散型モデル予測制御の VNE への適用

4.1 分散型モデル予測制御の概要

MPC は、管理対象の規模が大きくなるにつれて、制御にかかる計算時間が増大する問題がある。この問題を解消する方法として分散型モデル予測制御 (分散型 MPC) [7] と呼ばれる、対象とする制御系全体に対してエージェントを複数配置し、そのエージェントそれぞれが MPC を実行する手法が検討されている。分散型 MPC では、各エージェントがシステムへの入力を決定するが、エージェント間で情報交換をすることで、制御システムが分散化された状況においても、制御系全体としての目的に沿う入力を各エージェントが決定することが可能になる。

4.2 VNE–MPC の分散化

4.2.1 概　　要

本稿では、仮想ネットワーク埋め込みに分散型 MPC を適用した手法 (VNE–DMPC) を提案する。本手法では、各 VN に対応する仮想ネットワーク制御エージェント (VNA;Virtual Network Agent) を配置する。VNA 間の情報交換は、利用可能な物理ネットワーク資源を管理している物理ネットワーク管理エージェント (PNA;Physical Network Agent) を通して行う。

VNA は、PNA から他の VN が将来の各時刻に利用する資源の情報を取得し、当該 VN の収容位置を計算し、計算結果を PNA に通知する。そして、通知された情報は、PNA を介して、別の VNA が取得する。VNE–DMPC では、VNj に対応する VNA は最適化問題を解くことにより、VNj 内の仮想資源の収容位置を求め、PNA に通知する。

4.2.2 VNE–DMPC における制御目標

VNE-DMPC についても、各時刻 t で位置流動ノードが収容されている計算処理ノード数 $\xi(t)$ を最小化することを制御目標とする。しかしながら、分散型制御を適用している本手法では、各 VNA の制御対象は対応する一つの VN であり、他の VN の位置流動ノードの収容位置は制御することができない。そのため、$\xi(t)$ を最小化する最適化問題により仮想ノードの収容先を決定すると、他の VN の仮想ノードの移動を行うことができないため、制御対象の仮想ノードの移動のみでは計算処理ノード数を削減できない可能性がある。

そこで、本手法では、各仮想資源の収容にかかるコストを、収容先の物理ノードで提供可能な資源量に対して単調増加な関数として定義し、VNA が当該 VN を収容するのにかかるコストの最小化を目的関数とする最適化問題を作成する。これにより、空き資源が多い物理ノードの利用は避けた埋め込みを実現でき、埋め込みに必要な物理機器数を削減する挙動を示す。

4.2.3 VNE–DMPC の最適化問題

本最適化問題では、入力として、$SN=(N^p, L^p, P^p, G_{p,l})$ が与えられる。また、3.2 の U_n, B_l のかわりに、各計算処理ノード $n \in N_f^p$ について、他の VN から予約された仮想資源量を除いた、当該 VN に対して各時刻 t において提供可能な計算機資源容量として $C_n^{Node}(t)$ が与えられ、各リンク $l \in L^p$ についても各時刻 t において提供可能な通信帯域として $C_l^{Link}(t)$ が与えられる。また、VNj の情報について、$VNR=(N_j^v, L_j^v), \hat{u}_{j,v}(t), \hat{b}_{j,l}(t)$ が入力として与えられる。

上記の入力を用い、$M_{j,v,n}^{Node}(t), M_{j,l,p}^{Link}(t), M_n^{Node}(t)$ を定める。時刻 $t(0<t\leqq H)$ において計算処理ノード n へ位置流動ノードを収容するのにかかるコストを $C_n(t)$ として定義したときの、各時刻 t における仮想資源の収容先を求める最適化問題を以下に示す。

$$minimize$$

$$(1-w)\sum_{0<t\leqq H}\sum_{n\in N_f^p}X_n(t)+w\sum_{0<t\leqq H}\sum_{n\in N_f^p}\sum_{v\in N_{j,r}^v}R_{j,v,n}(t) \quad (13)$$

$$subject \ \ to$$

$$X_n(t)=M_n^{Node}(t)C_n(t) \quad (14)$$

$$R_{j,v,n}(t)=|M_{j,v,n}^{Node}(t)-M_{j,v,n}^{Node}(t-1)| \quad (15)$$

$$0<t\leqq H, \forall n\in N_f^p, \frac{1}{|N_{j,r}^v|}\sum_{v\in N_{j,r}^v}M_{j,v,n}^{Node}(t)\leqq M_n^{Node}(t) \quad (16)$$

$$0<t\leqq H, \forall v\in N_{j,r}^v, \sum_{n\in N_f^p}M_{j,v,n}^{Node}(t)=1 \quad (17)$$

$$0<t\leqq H, 1\leqq i\leqq |N^p|, \forall v_i\in N_{j,k}^v, M_{j,v_i,n_i}^{Node}(t)=1 \quad (18)$$

$$0<t\leqq H, \forall l\in L^{j,v}, \sum_{p\in P^p}M_{j,l,p}^{Link}(t)=1 \quad (19)$$

$$0<t\leqq H, \forall l\in L^v, \forall p\in P^p,$$
$$M_{j,l,p}^{Link}(t)\leqq M_{j,v_{j,l}^{start},n_p^{start}}^{Node}(t) \quad (20)$$

$$0<t\leqq H, \forall l\in L^v, \forall p\in P^p,$$
$$M_{j,l,p}^{Link}(t)\leqq M_{j,v_{j,l}^{end},n_p^{end}}^{Node}(t) \quad (21)$$

$$0<t\leqq H, \forall n\in N_f^p,$$
$$\sum_{v\in N_{j,r}^v}M_{j,v,n}^{Node}(t)\hat{u}_{j,v}(t)\leqq C_n^{Node}(t) \quad (22)$$

$$0<t\leqq H, \forall l^p\in L^p,$$
$$\sum_{l^v\in L_j^v}\sum_{p\in P^p}G_{p,l^p}M_{j,l^v,p}^{Link}(t)\hat{b}_{j,l}(t)\leqq C_{l^p}^{Link}(t) \quad (23)$$

ただし、本目的関数では、第一項は全時刻におけるコストの総和であり、第二項は仮想ノードの移動にかかるコスト、w は重みパラメーターである。$C_n(t)$ は、計算処理ノード n で当該 VN へ提供可能な資源に対して単調増加する関数であれば、任意の関数を用いることができる。5. 章の評価では、$C_n(t)$ は、各時刻 t で計算処理ノード n が当該 VNA へ提供可能な計算機資源容量 $C_n^{Node}(t)$ をコストとして定義した。

この制御では、各 VNA は、対応する VN の収容箇所のみ計算するため、計算時間を短く抑えることが可能であり、短い時間周期で、仮想資源の収容位置の計算を繰り返すことができる。また、この制御を短い周期で繰り返すことにより、将来の需要変動を考慮し、新たな VN の埋め込み要求の発生や、予想外の需要変動が発生した場合にも対応が可能となる。

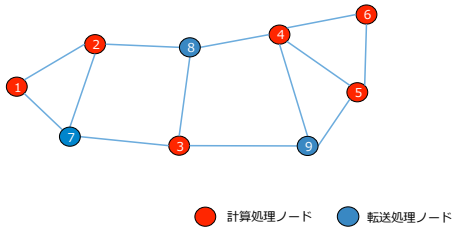

図 2　Internet2 トポロジー

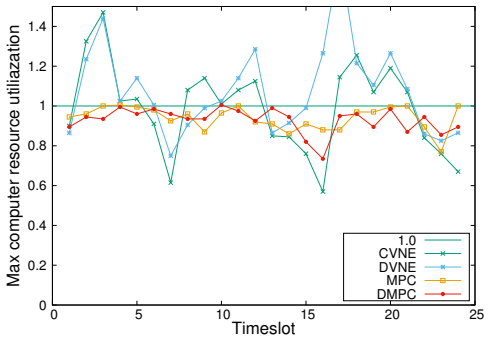

図 3　最大計算機資源利用率

5. 評　　価

5.1 比 較 手 法

CVNE 手法　2.3 に記した最適化問題を解くことにより、埋め込み位置の決定を行う手法である。

DVNE 手法　4.2.3 に記した最適化問題について、予測値を用いることなく、現時刻 t の需要値 $b(t), u(t)$ を用いて、時刻 $t+1$ のみの収容位置を計算する最適化問題を解くことによって仮想ネットワークの収容位置を決定する。DVNE 手法では、各 VNA は独立したタイミングで動作をすることができる。本評価では、VNA が同時に動作することはない仮定での評価を行う。

VNE–MPC 手法　3.2 に記した最適化問題によって仮想ネットワークの収容位置を決定する制御手法である。

VNE–DMPC 手法　4.2.3 に記した最適化問題を用いて収容位置を決める制御手法である。VNE–DMPC では、各 VNA は独立したタイミングで制御を行う。DVNE 手法と同様、本評価では、VNA が同時に仮想ネットワークの再構成を試みることはない場合を仮定して評価を行う。

5.2 評 価 環 境

5.2.1 物理ネットワーク環境

本評価では、対象とするネットワークとして、Internet2 のバックボーントポロジー (Internet2 トポロジー) を用いる。Internet2 トポロジーを図 2 に示す。Internet2 トポロジーは 9 ノードのうち、ノード番号 1~6 のノードが計算処理能力を有する計算処理ノードである。また、各計算処理ノードの計算機資源総量については 200 の均一な値としている。リンクについても同様に各リンクの通信帯域総量は 400 の均一な値として設定する。

また、本評価では、各物理ノードの組について、ホップ数が小さいものから順に 3 つのパスを候補パスとしており、Internet2 トポロジーにおける候補パス数は 216 である。

5.2.2 仮想ネットワーク要求

収容する VN のトポロジーと、仮想資源の需要変動の与え方について説明する。VN のトポロジーとして、ランダムに選択した位置固定ノードと、各位置固定ノードと接続した、位置固定ノードと同数の位置流動ノード、すべての位置流動ノードと接続している一つの位置流動ノードを含む VN を生成する。

VN が各タイムスロット $t(t \geqq 1)$ で要求するリンク帯域 $b(t)$、計算機資源量 $u(t)$ は、周期関数を用いて生成する。制御開始時のタイムスロットを 0 としたとき、タイムスロット 0 における当該 VN 中の仮想リンク、位置流動ノードが要求するリンク帯域 b_0、計算機資源量 u_0 は、[30,50] の範囲の整数を擬似乱数により取得し、各タイムスロット $t(t \geqq 1)$ の需要値は、式 (24)、式 (25) の周期関数から取得し、仮想資源要求の需要変動を与える。ここでは、δ、θ は各 VN ごとに割り当てられた値とし、δ は当該 VN が埋め込まれた順番を 3 で割った余りにより、余りが 0 の場合は 10、1 の場合は 12、2 の場合は 14 に設定される。また、θ は、VN が収容された順番と等しい値とする。

$$b(t) = \begin{cases} b_0 & (t=0) \\ b_0 + \delta(\sin\frac{2}{3}\pi(t-\theta) + \sin\frac{1}{4}\pi(t-\theta)) & (1 \leqq t) \end{cases} \quad (24)$$

$$u(t) = \begin{cases} u_0 & (t=0) \\ u_0 + \delta(\sin\frac{2}{3}\pi(t-\theta) + \sin\frac{1}{4}\pi(t-\theta)) & (1 \leqq t) \end{cases} \quad (25)$$

5.2.3 仮想資源要求における予測誤差生成方法

分散型モデル予測制御を適用するにあたり、制御開始タイムスロットを t とした場合、$[t+1, t+H]$ で仮想リンク、位置流動ノードが要求する仮想資源を予測する必要があるが、その予測結果には誤差が含まれることが想定される。そこで、本評価では、各タイムスロット t の仮想リンクの予測値 $\hat{b}(t)$、位置流動ノードの予測値 $\hat{u}(t)$ について、各タイムスロット t における仮想資源の需要の真の値 $b(t)$、$u(t)$ に正規乱数で生成した誤差 z の整数部分を足し合わせた値を予測値とし、$[t+1, t+H]$ の需要予測を取得する。各タイムスロットでの予測誤差を与える際の、正規分布における平均 μ と標準偏差 σ の指定方法についてであるが、本評価では、$1 \leqq i \leqq H$ なる整数 i について、タイムスロット $t+i$ での予測誤差を取得する際には、$\mu=0$、$\sigma=i$ とすることで、予測ステップが先になるにつれて予測誤差が比例的に増加することを実現する。

5.2.4 パラメーター設定

VNE–MPC、VNE–DMPC では、制御パラメーターとして、式 (1)、(12)、(13) 中の重み w と、予測対象区間を表す予測ホライズン H をパラメーターとして持つ。本報告では、$w = 0.9$ の定数として設定する。また、$H = 3$ を予測ホライズンとして設定する。

5.2.5 最適化問題の計算

本評価では、4 つの CPU(Xeon E7–4879) を搭載した計算機上で、CPLEX [8] を用いて最適化問題を解く。

5.3 評 価 結 果

5.3.1 モデル予測制御適用における資源利用率評価

本評価では、仮想ネットワーク埋め込み手法にモデル予測制御を適用したことによる効果として、需要変動が大きい環境下においても、VN が要求する計算機資源が物理ネットワーク上に確保されていることをシミューレション結果によって得られた結果から確認する。なお、以降では、VNE–DMPC を DMPC、VNE–MPC を MPC として表記する。

図 3 は、3 つの VN を収容した際に、位置流動ノードが収容

されている計算処理ノードのうち、各タイムスロットにおける最大計算機資源利用率を表したものである。図 3 からは、予測を行わない仮想ネットワーク埋め込み手法による制御を行った場合、当該計算処理ノードの処理能力を超える過剰な計算機資源容量が計算処理ノードに割り当てられているタイムスロット数が 14 であり、全タイムスロットの約 58%が計算処理ノードの処理能力の閾値を超える埋め込みが行われていたのに対し、予測をふまえた仮想ネットワーク埋め込み手法による制御を行った場合、閾値を超える埋め込みが行われていたタイムスロット数は 1 であり、収容能力を超えた埋め込みが行われる割合を約 4%に抑えられていることがわかる。これは、CVNE や DVNE が観測された過去の需要にもとづいて埋め込みを行っているのに対して、MPC や DMPC は予測された需要にもとづいて埋め込みを行っていることが原因である。需要変動が激しい場合は、CVNE や DVNE が埋め込みの入力として用いた情報と実際の需要が大きく異なる。その結果、需要が大きい仮想ノードを把握することができず、適切な配置ができない。それに対して、MPC や DMPC では、予測の誤差があるものの、将来の需要変動の傾向を把握した上で、それに合わせて仮想ノードの埋め込みを行う。予測誤差は遠い将来になれば大きくなるものの、MPC や DMPC では繰り返し新たな観測結果をもとにフィードバックを行うため、将来の予測誤差の影響を緩和できる。その結果、MPC や DMPC では、必要な計算機資源を割り当てるように仮想ノードの配置を決めることができ、最も計算機資源使用率が高い時間帯でも、計算機資源使用率を 1.005 に抑えることができている。

5.3.2 分散型制御手法が有するスケーラビリティ評価

分散型制御を導入することにより削減できる計算時間を明らかにする。本評価では、収容する VN 数を 10 から 100 まで変化させ、VNA が 1 回の埋め込み制御に要する計算時間を評価する。本評価においては、100 個の仮想ネットワークを収容するのに十分な資源総量を確保できるように、計算機資源総量を 4000、通信帯域総量を 5000 とし、シミュレーションを行った。

表 1 は、DMPC 手法、MPC 手法において、VNA が 1 回の埋め込み制御で要する計算時間 (秒) を表したものである。なお、この結果は、各 VNA が 10 回の埋め込み制御を行い、それらで要した計算時間の平均をまとめたものである。

表 1 より、DMPC 手法が各時刻において VN の収容先を計算するのにかかる時間は、VN 数が 100 であっても 0.1 秒以下となっている。それに対して、MPC は VN 数が 30 の場合でも、300 秒以上の計算時間が必要となる。これは、MPC では単一の VNA がすべての VN の埋め込み位置を計算しているのに対して、DMPC の各 VNA は対応する VN の埋め込み位置のみを計算しているためである。そのため、最適化問題における変数の数を小さく抑えることができ、計算時間を抑えることができる。

分散型仮想ネットワーク埋め込み手法において、各 VNA が同時に埋め込み制御を行わない、逐次処理が行われている状況を仮定したとしても、VN 数が 100 の場合であっても、0.06059×100＝6.059 秒しかかからず、VNE–DMPC は、VNE-MPC とくらべ、計算時間を大幅に削減することができる。

表 1　仮想ネットワーク収容位置決定の平均計算時間

収容 VN 数	DMPC 手法	MPC 手法
10	0.0187	0.139
20	0.01825	0.264
30	0.0982	353.383
40	0.088125	-
60	0.0899667	-
80	0.0611	-
100	0.06059	-

6. ま と め

本稿では、仮想ネットワーク埋め込みにおける、時々刻々と需要変動する環境へ追随する手法として、モデル予測制御を取り入れた仮想ネットワーク埋め込み手法を提案した。さらに、モデル予測制御適用における問題点である、収容 VN 数が増大するにつれて計算時間が増加し、制御周期が長くなり、環境変動へ追随できない可能性に焦点を当て、各 VN に対応する VNA を配置し、当該エージェントが対応する VN に関する制御を行う、分散型仮想ネットワーク埋め込み手法 VNE–DMPC を提案した。シミュレーション評価により、VNE–DMPC は、各時刻における計算時間を 0.1 秒以下と低く抑えつつ、需要の変動に追随して、仮想ノード、仮想リンクに十分な資源を割り当てるような VN の埋め込みを達成できることを示した。

本稿における評価では、需要の予測誤差は正規乱数で与えられるものとして評価を行った。今後は、実際の需要の予測手法と組み合わせて、提案手法の評価を行うことを予定している。

謝辞　本研究を遂行するにあたり、ご議論いただいた NTT ネットワーク基盤技術研究所の高橋洋介氏、上山憲昭氏、石橋圭介氏、塩本公平氏に感謝します。

文　献

[1] N.M.K. Chowdhury and R. Boutaba, “A survey of network virtualization,” Computer Networks, vol.54, no.5, pp.862–876, April 2010.

[2] A. Fischer, J.F. Botero, M.T. Beck, H. deMeer, and X. Hesselbach, “Virtual Network Embedding: A Survey,” IEEE Communications Surveys & Tutorials, vol.15, no.4, pp.1888–1906, Fourth Quarter 2013.

[3] J.F. Botero, X. Hesselbach, M. Duelli, D. Schlosser, A. Fischer, and H. deMeer, “Energy efficient virtual network embedding,” IEEE Communications Letters, vol.16, no.5, pp.756–759, May 2012.

[4] W. Liu, Y. Xiang, S. Ma, and X. Tang, “Completing virtual network embedding all in one mathematical programming,” Electronics, Communications and Control (ICECC), 2011 International Conference on, pp.183–185, Sept. 2011.

[5] S.J. Qin and T.A. Badgwell, “A survey of industrial model predictive control technology,” Control Engineering Practice, vol.11, no.7, pp.733–764, July 2003.

[6] T. Otoshi, “Prediction-based Control Theoretic Approach for Robust Traffic Engineering,” Master’s thesis, Graduate School of Information Science and Technology, Osaka University, Feb. 2014.

[7] E. Camponogara, D. Jia, B.H. Krogh, and S. Talukdar, “Distributed Model Predictive Control,” IEEE Control Systems Magazine, vol.22, no.1, pp.44–52, Feb. 2002.

[8] “IBM ILOG CPLEX Optimizer”. optimization software : http://www-01.ibm.com/software/integration/optimization/cplex-optimizer.