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.

生物のモジュール構造に基づく進化適応能力のサービスチェイニング制御への応用

乙倉　麻里† 　ライプニッツ賢治††,† 　下川　哲也††,†††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
†† NICT/大阪大学 CiNet　〒 565–0871 大阪府吹田市山田丘 1–4
††† 大阪大学 大学院生命機能研究科　〒 565–0871 大阪府吹田市山田丘 1–3
E-mail: †{m-otokura,murata}@ist.osaka-u.ac.jp, ††leibnitz@nict.go.jp, †††simokawa@fbs.osaka-u.ac.jp

あらまし　ネットワーク仮想化技術の進展に伴い，サービスチェイニングが注目されている．サービスチェイニングにおいては，ネットワーク仮想化技術を利用することにより，サービスチェーンの構成・規模を動的に変更することが可能である．この時，ユーザごとの要求やトラフィック変動に対して迅速にサービスチェーンを変更することが求められる．しかしながら，サービスチェイニングにおける接続機器数は将来の IoT サービスの発展を考慮すると膨大になると想定されるため，適切な規模のサービスを迅速に提供するための計算処理を実時間で行うことは困難になると考えられる．我々は，ここに生物の進化適応能力の知見を応用する．生物は，進化の過程において，その生物学的なネットワークにモジュール性を自発的に生じさせることにより，環境変動に対する適応能力を上げてきた．本稿では，こうした生物の進化適応能力の知見を応用することにより，ネットワーク環境の変動に低コストで対応可能なサービスチェイニング制御手法を提案する．

キーワード　ネットワーク仮想化；NFV（Network Function Virtualization）；サービスチェイニング；遺伝的アルゴリズム；仮想マシン配置問題

Scalable control in service chaining through modularity from biological evolution

Mari OTOKURA†, Kenji LEIBNITZ††,†, Tetsuya SHIMOKAWA††,†††, and Masayuki MURATA†

† Graduate School of Information Science and Technology, Osaka University
1–5 Yamadaoka, Suita, Osaka 565–0871, Japan
†† Center for Information and Neural Networks (CiNet), NICT and Osaka University
1–4 Yamadaoka, Suita, Osaka 565–0871, Japan
††† Graduate School of Frontier Biosciences, Osaka University
1–3 Yamadaoka, Suita, Osaka 565–0871, Japan
E-mail: †{m-otokura,murata}@ist.osaka-u.ac.jp, ††leibnitz@nict.go.jp, †††simokawa@fbs.osaka-u.ac.jp

Abstract　Along with the development of network virtualization technology, service chaining is also gathering a lot of attention. In service chaining, the composition and scale of each service chain can be flexibly configured through network virtualization. In that case, it is necessary to change the service chain rapidly according to requests of users and temporal variations of traffic. However, due to the large number of connected devices expected for Internet of Things (IoT) services, it will be difficult to provide the appropriate sizes of network functions in real time. Here, we apply the knowledge of biological evolvability to this problem. It has been shown that the modularity of biological networks arises spontaneously in the process of evolution, which leads to a high ability to adapt to varying environments. In this work, we propose a technique inspired by such biological evolvability to rapidly adapt service chains to changes of the network condition.

Key words　Network Virtualization; NFV (Network Function Virtualization); Service Chaining; Genetic Algorithm; Virtual Machine Placement Problem

1. はじめに

近年，SDN（Software Defined Network）や NFV（Network Function Virtualization）等のネットワーク仮想化技術が急速に発達している．SDN は，従来は単一のネットワーク機器上に存在していたデータ転送部とコントロール部を分離することにより，ネットワークをコントロールソフトウェアによって集中的に制御することを可能とする技術である．NFV は，従来は専用ハードウェアによって実現されていたネットワーク機能をソフトウェア化し，汎用サーバ上の仮想マシン（VM）で実行可能とする技術である．これらの技術により，従来は困難であったネットワークの動的で柔軟な制御が可能となる．

SDN/NFV の実用例として注目されているのが，サービスチェイニングである．サービスチェイニングとは，NFV で実現されるソフトウェア化されたネットワーク機能，すなわち仮想ネットワーク機能（VNF）を SDN により鎖状に繋いだものを 1 つのサービスとし，ユーザごとにカスタマイズされたサービスを提供することである．これにより，ユーザごとの要求の変化に応じたサービスの提供や，トラフィック増減に応じてネットワーク機能の規模を増大，縮小させるオートスケーリングによる資源の有効活用が可能となる．

サービスチェイニング制御においては，ユーザの要求に対して迅速にサービスを提供できることが求められる [1]．ここで，サービスチェイニングにおけるユーザ数は，今後の IoT サービスの実現等も考慮すると，膨大な数になることが想定されている．このため，変更要求発生時に最適化問題を解くことにより逐次資源配置と経路計算を行う，という単純な手法では，計算時間の面で対応が困難である．

本稿では，生物の進化適応能力の知見を活かしたサービスチェイニング制御手法を提案する．生物は環境変動に対する適応性を有するが，それは生物ネットワークのモジュール性に由来するものであると考えられている．実際に，多くの生物ネットワークでモジュール性が見られることが知られている [2]．生物ネットワークにモジュール性が生じる要因の 1 つとして，環境変動の中での進化が挙げられている．この環境変動の中での進化適応の概念を遺伝的アルゴリズムに組み込んだのが，MVG-GA（Modularly Varying Goals - Genetic Algorithm）[3] である．本稿では，MVG-GA を利用した，ネットワークの環境変動に低コストで対応可能なサービスチェイニング制御手法を提案し，その有効性を確認する．

本稿の構成は以下の通りである．2 章では，本稿で着目した生物の環境変動の中での進化に伴い生じる性質と，それを遺伝的アルゴリズムに組み込んだ MVG-GA について解説する．3 章では，MVG-GA のサービスチェイニング制御への応用手法について述べる．4 章では，提案手法の性能評価を行う．最後に 5 章で本稿のまとめと今後の課題について述べる．

2. 環境変動の中で生物に進化的に生まれる性質

タンパク質相互作用ネットワーク・遺伝子制御ネットワーク・脳機能ネットワークなどの生物ネットワークは多くの場合でモジュール構造を有することが知られている [2]．ネットワークのモジュール構造とは，ノードが密に連結されたモジュールと呼ばれる部分同士が，疎につながっている構造である．生物ネットワークにモジュール構造が生じる要因の 1 つとして，環境変動の中での進化が挙げられている [3]．生物が環境変動の中で進化すると，生物は各時点での環境に適応するために変化し，それに伴い生物ネットワークも変化していく．このとき，複数環境で共通して必要となる機能に対応する部分が，生物ネットワークの構造中の変化しない部分として現れるようになり，これが生物ネットワークのモジュールとなる．この結果，生物が環境変動に適応するために必要となる生物ネットワークの変化は，ネットワーク全体の変化ではなく，モジュール間のリンクの張り替えだけになる．

生物の進化を模した最適化アルゴリズムとして，遺伝的アルゴリズム（GA）がある．これは，ゲノムをある最適化目標に対して選択・突然変異・交叉等の遺伝的操作を用いて進化させるアルゴリズムである．文献 [3] では，GA に生物の環境変動の中での進化の概念を導入した，MVG-GA が提唱されている．通常の GA では 1 つの目標のみに対してゲノムを進化させるのに対し，MVG-GA では一定世代ごとに目標を切り替えて進化させる．ここで用いられる目標群は，いくつかの副目標を異なる方法で組み合わせたものとなっている．MVG-GA によって目標変動の中でネットワークを進化させた場合，最終的に，目標中の副目標に対応する部分がモジュールとなり，目標の切り替えにはモジュール間のリンクの張り替えだけで対応できるようになる．このような結果となる原因は，目標変動の中での進化を続けるうちに，ゲノム中に副目標に対応する部分が現れ，それがネットワークのモジュールとして表出するためである．目標が副目標の組み合わせになっていることより，目標切り替え後は，副目標すなわちモジュールに対応する部分のゲノムを変化させる必要はなく，モジュール間リンクに対応する部分のゲノムを突然変異で数回変化させるだけで新しい目標への対応が可能となる．すなわち，MVG-GA により，目標変動に少ない操作で迅速に適応できるゲノムが生成される．

3. MVG-GA のサービスチェイニング制御への応用

本稿では，ネットワーク環境変動としてトラフィック変動を考える．また，サービスチェイニング制御の中でも特に，物理ネットワーク上のサーバへの VNF 配置問題を考える．以下では，トラフィック変動下における物理ネットワーク上のサーバへの VNF 配置手法について解説する．アルゴリズムとして，2 章で説明した MVG-GA を用いることで，トラフィック変動に少ない操作で迅速に適応できる VNF 配置を生成する．

3.1 問題のモデル化

ネットワーク機能の構造は，パケット受信部・フロー状態検索部・パケット処理部の 3 つに分けられ，ネットワーク機能での処理はそれらを順に経由して行われる [4]．すなわち，パケットがパケット受信部に到着し，それに対するフロー状態をフロー状態検索部で検索して処理を決定し，パケット処理部で処

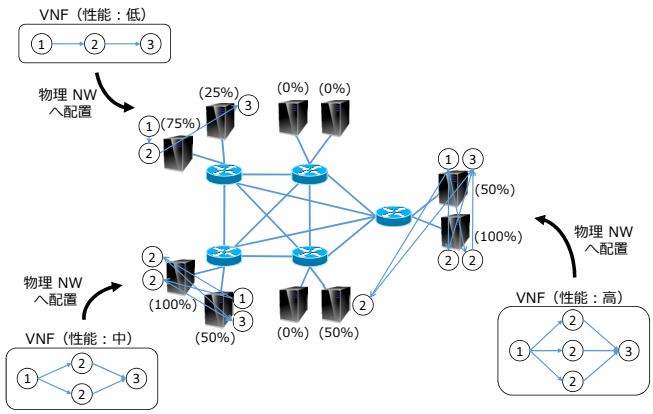

図 1　モデル化の概念図．コンポーネント 1，2，3 の配置にそれぞれ 4，8，4 コアが必要であり，各サーバは 16 コアを持つとした場合の配置の一例を示す．括弧内のパーセンテージは各サーバのコア使用率である

理が行われて出力される．本稿では，以上のネットワーク機能の特性に基づき，1 つの VNF は 3 つの要素で構成されるとし，各要素を順にコンポーネント 1，2，3 と呼ぶ．

フロー数が多くなった場合にネットワーク機能中でボトルネックとなるのはフロー状態検索部である [5]．よって，フロー数が多くなった時には特にフロー状態検索部の性能を向上させる必要がある．NFV においては，VM の個数増大によって性能の向上が可能となる．これを考慮し，本稿では，各コンポーネントは VM で実行されるとして，トラフィック変動を，各 VNF のコンポーネント 2 に対する VM 要求数（以降では要求コンポーネント数と表記）の増減で表現する．

サービスチェイニングの VNF は通信事業者の保有するクラウド上のサーバに配置される．通信事業者のクラウドは，将来的には 1 つのデータセンターのみでなく複数のデータセンターで構成されるようになると予想されている [6]．この場合，データセンター間の通信では遅延が発生するため，1 つの VNF をできるだけ同一データセンター上に配置することが重要となる．これを考慮し，本稿では，物理ネットワークとして，複数サーバが接続したルータからなるネットワークを考える．このとき，ルータを 1 つのデータセンター，ルータ間ネットワークをデータセンター間ネットワークとみなすことができる．

各サーバに配置できるコンポーネント数は有限である．文献 [7] では，VNF をサーバに配置する問題を扱う際に，コア数の概念を導入することで各サーバに配置できる VNF の数を制限している．すなわち，各 VNF を実行する VM に必要なコア数と各サーバが持つコア数を予め指定しておき，各サーバが持つコア数を超過しないように VNF を配置している．本稿ではこれに倣い，各コンポーネントを実行する VM に必要なコア数（以降ではコンポーネントの必要コア数と表記）と各サーバが持つコア数を予め指定しておき，各サーバが持つコア数を超過しないようにコンポーネントを配置することで，各サーバに配置できるコンポーネント数を制限する．

以上をまとめたモデル化の概念図を図 1 に示す．

3.2　MVG-GA の応用方法

3.1 節でモデル化した問題に MVG-GA を応用する方法を解説する．物理ネットワーク上への複数 VNF の配置状態を生成する GA を継続的に実行し，要求コンポーネント数が変動した場合は目標及び適応度関数を変更する．これにより，ゲノムは目標及び適応度関数の変動（以降では目標変動と表記）の下で進化することになる．すなわちこの GA は MVG-GA となる．以下，その詳細について述べる．

本稿の MVG-GA は以下の手順で実行される．ii）～vii）を 1 世代とする．

i)　N 個のゲノムをランダム生成する．

ii)　（1 世代目以外）前世代の実行中に要求コンポーネント数が変動した場合は，適応度関数の変更を行う．

iii)　適応度関数により，各個体の適応度を計算する．

iv)　適応度上位 α 個のゲノムをエリートとし，保存する．

v)　適応度下位 α 個以外のゲノムに対して，交叉率 p_c で交叉を行い，その後突然変異率 p_m で突然変異を行う．

vi)　保存しておいたエリートと，交叉と突然変異を行ったゲノムを，次世代のゲノムとする．

vii)　ii）に戻る．

ただし，交叉率・突然変異率は，全ゲノムのうち交叉・突然変異を起こすゲノムの割合を示すものである．ii）において適応度関数を変更している点が，通常の GA と大きく異なる点であり，本稿の GA が MVG-GA となる要因である．

GA のゲノムは物理ネットワーク上へのコンポーネントの配置状態を表し，適応度関数はその配置の良さを計算する関数となる．適応度関数 F として式 (1) を用いる．

$$F = \kappa - \frac{P_U + P_C}{\kappa} \tag{1}$$

ここで，κ は要求コンポーネント数と現在の配置のコンポーネント数との一致度，P_U は物理ネットワーク資源使用率に応じて課せられるペナルティ，P_C はコンポーネントの必要コア数合計がサーバのコア数を超過した場合に課せられるペナルティである．κ は以下のように求められる．

$$\kappa = 1 - \frac{\sum_{j=1}^{3}\sum_{k=1}^{V} D_{jk}}{3 \cdot V}$$

ただし，

$$D_{jk} = \begin{cases} 0 & \text{if } R_{jk} = \sum_{i=1}^{S} x_{ijk} \\ 1 & \text{otherwise} \end{cases}$$

ここで，D_{jk} は要求コンポーネント数と配置数が一致していなければ 1，していれば 0 となる変数，V は配置する VNF 数，R_{jk} は VNFk に対するコンポーネント j の要求数，x_{ijk} はサーバ i への VNFk のコンポーネント j の配置数を表す変数である．D_{jk} は x_{ijk} より求まり，x_{ijk} はゲノムより決定される．また，P_U は以下のように求められる．

$$P_U = \begin{cases} U - U_{th} & \text{if } U \geqq U_{th} \\ 0 & \text{otherwise} \end{cases}$$

ただし，

$$U = \left(\frac{\hat{C}}{C} + \frac{\hat{L}}{L}\right) \times \frac{1}{2}$$

$$\hat{C} = \sum_{i=1}^{S}\sum_{j=1}^{3}\sum_{k=1}^{V}(C_j \times x_{ijk})$$

$$C = \sum_{i=1}^{S} S_i \qquad \hat{L} = \sum_{k=1}^{L} y_k$$

ここで，U は物理ネットワーク資源の利用率（以降では資源利用率と表記）を表す変数，$\hat{C}$ は使用されるコンポーネントの必要コア数の合計を表す変数，C は全サーバのコア数の合計，$\hat{L}$ は使用されるルータ間リンク数を表す変数，S は物理ネットワーク上のサーバ数，C_j はコンポーネント j が必要とするコア数，S_i はサーバ i が持つコア数，L は物理ネットワークのルータ間リンク数，y_k はリンク k が使われているならば 1，使われていなければ 0 となる変数である．U，$\hat{C}$，$\hat{L}$，y_k は x_{ijk} より求まる．また，P_C は以下のように求められる．

$$P_C = P \cdot \sum_{i=1}^{S} E_i$$

ただし，

$$E_i = \begin{cases} 1 & \text{if} \quad \sum_{j=1}^{3}\sum_{k=1}^{V}(x_{ijk} \times C_j) > S_i \\ 0 & \text{otherwise} \end{cases}$$

ここで，E_i はコア数超過サーバの数を表す変数，P はコア数超過サーバが 1 つ生じる度に適応度に課せられるペナルティである．E_i は x_{ijk} より求まる．

式 (1) を適応度関数とすることで，コンポーネント要求数を過不足なく満たす，遅延・資源使用率が閾値以下となる，各サーバで使用可能なコア数を超過しない，の 3 点を満たすコンポーネント配置の適応度が 1 となる．

4. シミュレーション評価

本稿では，コンピュータシミュレーションにより，3.2 節で解説した MVG-GA に基づくサービスチェイニング制御手法（以降では MVG-GA 手法と表記）の，要求コンポーネント数変動に対する適応性を評価する．

4.1 評価環境

シミュレーションで用いた物理ネットワークの構造とトラフィック変動（コンポーネント数の変動）について解説する．

4.1.1 物理ネットワークの構造

物理ネットワークとして，図 1 に示す物理ネットワークを用いる．物理ネットワークは 5 個のルータをフルメッシュ状に接続したものであり，各ルータには 2 つのサーバが接続している．サーバ・ルータには番号が付与される．サーバ番号は 1〜10，ルータ番号は 1〜5 となる．各サーバの持つコア数は 16 とする．各コンポーネントの必要コア数は，コンポーネント 1，2，3 に対しそれぞれ 4，8，4 コアとする．

4.1.2 トラフィック変動

3.1 節で解説したように，本稿ではトラフィック変動をコンポーネント 2 の要求数の増減で表現する．本稿では，物理ネットワークに同種の VNF5 個を配置する問題を考える．このとき，r_k を VNFk に対するコンポーネント 2 の要求数とすると，コンポーネント 2 に注目した要求コンポーネント数 $R'_k(t)$ は $R'_k(t) = (r_1, r_2, r_3, r_4, r_5)$ なるベクトルとして表される．$R'(t)$ は間隔 T ごとに，$|R'(t-T) - R'(t)| = 1$ となるように変動する．

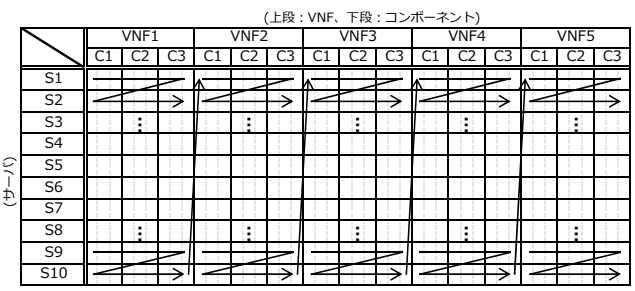

図 2 ゲノム構造

表 1 ゲノム列の数値への変換方法

ゲノム列	コンポーネント 1，3 の数	コンポーネント 2 の数
000	0	0
001	0	0
011	0	1
010	0	2
110	0	3
111	1	0
101	0	0
100	0	0

4.2 提案手法

3.2 節で解説した MVG-GA を利用する．以下では，今回のシミュレーションでの設計・設定について解説する．

4.2.1 ゲノムの設計

本稿で用いた GA のゲノムの構造を図 2 に，ゲノム列の数値への変換方法を表 1 に示す．図 2 は，実線部がサーバへのコンポーネントの割り当て個数の表となっている．ここで，サーバへ割り当てられるコンポーネント数が最大となるのは，コンポーネント 1 または 3 を 4 個割り当てた場合となる．よって，割り当て個数は 0〜4 となる．このため，表の各要素は図 2 の破線で示されるように 3 ビットで表される．ゲノムは，図 2 を矢印で示されるようにベクトル化したものである．

4.2.2 パラメータ設定

本稿で用いた GA のパラメータ設定を以下に示す．

- 集団数 N：1000
- エリート数 α：100
- 交叉率 p_c：0.9
- 突然変異率 p_m：0.9
- シミュレーション終了世代数 G_{max}：50000 世代
- 目標変動の間隔 T：20 世代
- 資源使用率の閾値 U_{th}：0.5
- 超過サーバ 1 つあたりに課せられるペナルティ P：0.1

4.2.3 MVG-GA 手法によって得られる配置列

各目標に対する解は，目標が別のものへ変動する直前の世代で得られたゲノムのうち，適応度が最大のものとする．すなわち，$T = 20$ 世代ごとに，目標に対する解を取得し次の世代で目標を変動させる．よって，$G_{max} = 50000$ 世代の MVG-GA シミュレーションで，$\frac{Gmax}{T} = 2500$ 個の配置からなる配置列が得られる．また，2500 個の R_{jk} が，MVG-GA に入力された R_{jk} の列として得られる．

4.3 比較対象

比較対象として，ILP (Interger Linear Problem：線形計画問題）により定式化された問題を解く（以降ではこれを ILP 手法と表記）ことで得られる配置を用いる．ILP 手法のシミュレーションでは，MVG-GA 手法のシミュレーションにおいて得られた 2500 個の R の列を，MVG-GA 手法のシミュレーションにおいて入力された順番と同一の順番で ILP に入力する．これにより，MVG-GA 手法シミュレーションと同一の要求コンポーネント数変動に対して，ILP 手法を用いて配置を行った場合の配置 2500 個の列が得られる．これを MVG-GA 手法で得られた配置 2500 個の列と比較する．

4.3.1 ILP による定式化

ILP により，コンポーネント要求数を過不足なく満たす，各サーバで使用可能なコア数を超過しない，の 2 点の制約を満たした，遅延・資源使用率が最小となる配置を求める．

以下，問題の ILP による定式化について解説する．以下では，3.2 節と同一の変数・定数を用いる．決定変数は x_{ijk} である．最小化関数は U である．制約条件は以下の 6 個である．

- $\kappa = 1$
- $\forall i \quad \sum_j^3 \sum_k^V (x_{ijk} \times C_j) \leqq S_i$
- $\forall i, j, k \quad x_{ijk}$ は整数であり $0 \leqq x_{ijk} \leqq 4$
- $\forall l \quad y_l$ はバイナリ値
- $\forall r, r' \quad (\exists k \quad (\tilde{x}_{r1k} = 1 \wedge \tilde{x}_{r'2k} \geqq 1) \rightarrow y_l = 1)$
- $\forall r, r' \quad (\exists k \quad (\tilde{x}_{r3k} = 1 \wedge \tilde{x}_{r'2k} \geqq 1) \rightarrow y_l = 1)$

ここで，$\tilde{x}_{rjk}$ は，ルータ r に接続するサーバに配置されている，VNFk のコンポーネント j の数を表す変数であり，$\tilde{x}_{rjk} = x_{(2r)jk} + x_{(2r+1)jk}$ である．また，r,r' は，リンク l の両端のルータ番号とする．

4.4 評価指標

評価指標として，目標変更に伴う配置変更に必要な操作数（以降では配置変更操作数と表記），資源利用率 U を用いる．以下では，配置変更操作数について解説する．

4.4.1 配置変更操作数

各コンポーネントが VM で実行されることを想定しているため，配置変更は VM の migration・replication・merge によって行われる．ここで，migration は VM をあるサーバから別のサーバへ移動させること，replication は VM を複製すること，merge は複数 VM を併合して 1 つにすることを指す．本稿では，各配置変更に必要な migration・replication・merge の回数を評価指標として用いる．これらの操作ではほとんどの時間がネットワークを介したディスクやメモリの移動に費やされる [8] ため，本稿ではルータ間リンクを介した操作のみを数えるとする．すなわち，migration では，あるコンポーネントが，それが配置されているサーバと接続するルータとは別のルータのサーバへ移動した時を migration 1 回として数える．replication・merge では，あるコンポーネントの数が，それが配置されているサーバと接続するルータとは別のルータのサーバにおいて増加・減少するときに，元の配置における個数からの増加数・減少数をそれぞれ replication・merge の回数とする．

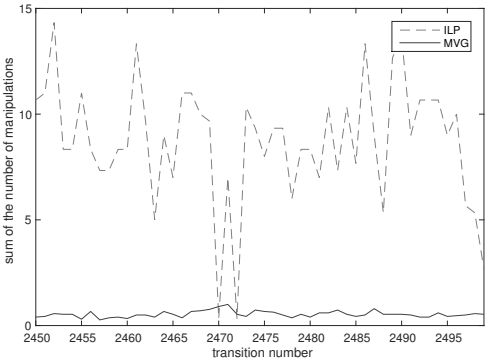

図 3　2450〜2499 回目の配置変更における配置変更操作数の合計．横軸は何番目の配置変更かを表す

4.5 評価結果

MVG-GA 手法の結果としては，同じ R の列を使い，乱数のシードを変更した 10 回のシミュレーションの平均値を用いている．ILP 手法の結果としては，MVG-GA 手法と同じ R の列を使ったシミュレーション 1 回の結果を用いている．

4.5.1 配置変更操作数

図 3 に 2450〜2499 回目の配置変更における配置変更操作数の合計（replication, merge, migration 数の合計），図 4 にシミュレーション全体での配置変更操作数の分布，表 2 にシミュレーション全体での配置変更操作数の平均値と標準偏差を示す．これらの図表より，MVG-GA 手法における配置変更操作数が，ILP 手法と比べ少なくなっていることが分かる．また，図 4 からは，MVG-GA 手法では，シミュレーション中の全ての配置変更において，migration・replication・merge の各操作数が 3 を超えることはほとんどなかったことが分かる．このように MVG-GA 手法において少ない操作数で配置変更が行われているのは，MVG-GA 手法では，目標変動の中でゲノムが進化するにつれて，全目標に共通な部分がゲノム中に出現するためだと考えられる．また，図 3 からは，最終 50 回の配置変更においては，MVG-GA では各配置変更に必要な操作数の合計が 1 程度と非常に小さくなっていることが分かる．これは，目標変動に関する学習がほとんど完了したシミュレーション終盤においては，数回の突然変異によって目標変動に対応できるようなゲノムが生成されているためだと考えられる．

4.5.2 資源利用率

図 5 に 2450〜2500 番目の配置における資源使用率 U，図 6 にシミュレーション全体での資源使用率 U の分布を示す．ILP 手法では U を最小化する配置が生成される一方で，MVG-GA 手法は U が閾値 $U_{th} = 0.5$ を超えた場合にのみ適応度にペナルティが加えられるため，シミュレーション全体で ILP 手法

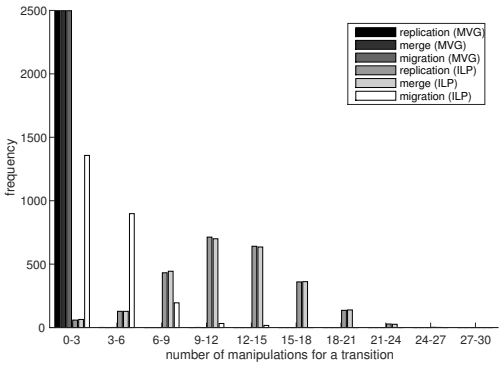

図 4　シミュレーション全体での配置変更操作数の分布

表 2　シミュレーション全体での配置変更操作数の平均値と標準偏差

	MVG		ILP	
	平均値	標準偏差	平均値	標準偏差
replication	0.58	0.42	1.6	4.2
merge	0.58	0.50	1.6	4.2
migration	0.57	0.32	13.0	2.4

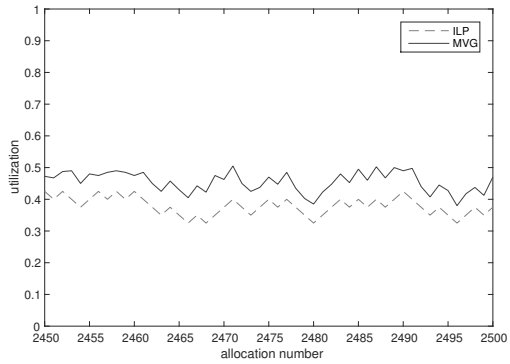

図 5　2450〜2500 番目の配置における資源使用率 U

の U は MVG-GA 手法の U 以下となる．シミュレーション全体での ILP 手法と MVG-GA 手法の U の差の平均値は 0.0825 となっている．本稿ではリンク 10 本，サーバ 10 個というネットワークを用いたことと U の定義を考慮すると，0.0825 という値は，MVG-GA 手法でリンクまたはサーバが 1〜2 個余分に使われているということを意味する．このような結果になった要因の 1 つとして，MVG-GA 手法によって生成された変動に強い配置は，最適配置と比べていくらかの冗長性を必要とするため，資源が余分に必要であることが考えられる．しかし，物理ネットワークの規模を大きくする，適応度に与えるペナルティを決定する値である U_{th} を適切に設定するなどの変更により，この差をより小さくすることが可能であると考えられる．

5. まとめと今後の課題

本稿では，環境変動の中での生物の進化を模した遺伝的アルゴリズムである MVG-GA を利用した，トラフィック変動下での物理ネットワーク上のサーバへの VNF 配置手法を提案した．提案手法では，トラフィック変動に伴う要求コンポーネント数変動の度に GA の目標を切り替える MVG-GA により，コンポーネント配置を表すゲノムを変動に強い構造へ進化させる．

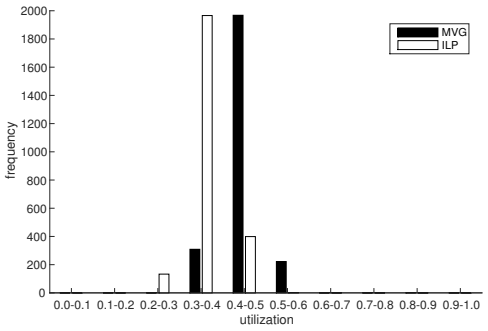

図 6　シミュレーション全体での資源使用率 U の分布

提案手法により，ある要求コンポーネント数に対する配置から次の要求コンポーネント数に対する配置への変更に必要な操作数が少なくなり，変動に対してより適応的な制御を行えることが明らかとなった．本稿では 10 サーバ・10 リンクの物理ネットワークを想定して評価を行ったが，より大規模で複雑なネットワークを想定した評価も必要である．今後は，コンポーネントの配置だけではなく，VNF 複数で構成されるサービスチェーンの配置を行えるように提案手法を拡張し評価を行う予定である．

謝辞　本研究の一部は，文部科学省博士課程教育リーディングプログラムの補助によるものである．ここに記して謝意を表す．

文　献

[1] W. John, K. Pentikousis, G. Agapiou, E. Jacob, M. Kind, A. Manzalini, F. Risso, D. Staessens, R. Steinert, and C. Meirosu, “Research directions in network service chaining,” in *Proceedings of 2013 IEEE SDN for Future Networks and Services*, pp. 1–7, Nov. 2013.

[2] G. P. Wagner, M. Pavlicev, and J. M. Cheverud, “The road to modularity,” *Nature Reviews Genetics*, vol. 8, pp. 921–931, Dec. 2007.

[3] N. Kashtan and U. Alon, “Spontaneous evolution of modularity and network motifs,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, pp. 13773–13778, Sept. 2005.

[4] S. Rajagopalan, D. Williams, H. Jamjoom, and A. Warfield, “Split/merge: System support for elastic execution in virtual middleboxes,” in *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, pp. 227–240, Apr. 2013.

[5] Y.-K. Chang, C.-I. Lee, and C.-C. Su, “Multi-field range encoding for packet classification in tcam,” in *Proceedings of the 30th IEEE international conference on Computer Communications*, pp. 196–200, Apr. 2011.

[6] J. Soares, C. Goncalves, B. Parreira, P. Tavares, J. Carapinha, J. Barraca, R. Aguiar, and S. Sargento, “Toward a telco cloud environment for service functions,” *Communications Magazine, IEEE*, vol. 53, pp. 98–106, Feb. 2015.

[7] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, “On orchestrating virtual network functions in NFV,” *eprint arXiv:1503.06377*, Mar. 2015.

[8] U. Sharma, P. Shenoy, S. Sahu, and A. Shaikh, “A cost-aware elasticity provisioning system for the cloud,” in *Proceedings of the 31st International Conference on Distributed Computing Systems*, pp. 559–570, June 2011.