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.

パレート最適制御に基づく進化的ネットワーク省電力化手法

秋下　耀介†　　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: †{y-akishita,y-ohsita,murata}@ist.osaka-u.ac.jp

あらまし　近年，ネットワークにおける消費電力の増加は大きな課題となっており，不要なネットワークの機器やサーバをスリープさせ，ネットワークの消費電力を削減する手法の検討が進められている．しかしながら，従来のネットワーク低消費電力化手法の検討では，信頼性の確保については考慮しておらず，故障が発生した際には，ネットワークの性能が低下する．我々は，十分な通信性能，信頼性の確保と低消費電力化の 3 つの目的を達成するネットワーク制御手法を確立した [1]．しかしながら，トポロジの規模が大きくなった場合や総トラヒック量が大きく変化するような変動を考慮しておらず，省電力状態になるまで時間を要する．そこで，このような場合でもネットワーク内の環境変動に追随して省電力化を行うために，前の時刻のパレートフロントと，進化する能力の高い解を組み合わせた集合を初期解とし，進化計算により，現在の環境に合わせたパレートフロントを探索する．これにより，高速に適切なパレートフロントの取得が可能となる．本稿では，提案手法をシミュレーションにより評価を行い，提案手法により，環境変動に追随し，約 4 世代で性能・信頼性の要件を満たし，約 650 世代でネットワークの消費電力を低減することができることを示す．

キーワード　ネットワーク省電力化，多目的最適化，パレート最適解，パレートフロント，進化計算，進化性

Evolutionary Network Power-Saving Method based on Pareto Optimal Control

Yosuke AKISHITA†, 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: †{y-akishita,y-ohsita,murata}@ist.osaka-u.ac.jp

Abstract　The power consumption of networks has been increasing as the service over the Internet becomes popular, and has become a serious problem. Many methods to reduce the power consumption by shutting down unnecessary network devices have been proposed. However, these methods do not consider the fault tolerance, and shutting down some network devices may degrade the performance of the network in case of failures. We have established a network control method that saves power consumption securing adequate performance and reliability [1]. However, it does not take into consideration fluctuations in which the topology scale becomes large or the total traffic volume changes greatly. And it takes time to reach the power saving state. Therefore, in order to save power following the environmental fluctuation in the network, a set obtained by combining the previous Pareto front and solutions having a high evolvability is set as initial solutions. And by evolutionary computation, our method searches for the Pareto Front adapted to the current environment. According to the above, our method obtains an appropriate Pareto front immediately. In this paper, we evaluate the method by simulation. The results show that the method can satisfy performance and reliability constraints in about 4 generations, and reduce power consumption in about 650 generations.

Key words　Network power saving, Multi-objective optimization, Pareto optimal solutions, Pareto front, Evolutionary algorithms, Evolvability

1. はじめに

近年，ストリーミング配信や，クラウドストレージのようなクラウドサービス等のインターネットを介したサービスが普及するにつれて，ネットワーク内のトラヒックは増加し続けている [2]．トラヒック増加に伴い，ネットワークの消費電力は増加の一途にあり，大きな課題となっている [3]．

ネットワーク全体の制御における低消費電力化は，不要なネットワーク機器やサーバをスリープさせることで実現可能である．従来，このような低消費電力化は，１日周期の通信需要変動を想定した手法が検討されてきた [4], [5]．これらの手法では，通信量が多く，少数のネットワーク機器では十分な性能を確保することができない場合は，多数のネットワーク機器を動作させ，十分な処理性能を確保し，逆に，通信量が少ない時間帯には，多くの機器をスリープさせることによって低消費電力化が可能となる．

これらの従来のネットワーク省電力化手法では，信頼性の維持は考慮されていない．これらの手法では，省電力化のために，一部ノードやリンクへトラヒックを集約し，ノードやリンクをスリープさせることにより，冗長性が除去される．そのため機器の故障発生時にはネットワークの接続性が確保されず，サービスの停止を招く可能性がある．それに対して，現実のネットワークサービスでは，故障が発生した際にも，故障発生により性能低下する時間を一定以下とすることが求められる．

我々は，環境変動に追随し，性能・信頼性の両面から SLA（Service Level Agreement）を保証しながらも省電力化を実現するネットワーク制御手法を確立した [1]．我々の手法では，複数の指標を考慮して最適化を行うパレート最適制御をネットワークに適用した．パレート最適制御においては，多目的進化計算（Multi-Objective Evolutionary Algorithms：MO-EA）[6] によりパレート最適解の集合（パレートフロント）を求める．我々の手法では，消費電力，信頼性，性能の３つの指標についてのパレートフロントを進化計算により求める．そして，SLA を保証できる範囲内で最も低消費電力な解をネットワークに投入する．また，前の時刻のパレートフロントを初期解とし，進化計算により現在の環境に合わせたパレートフロントを探索することで，ネットワーク内の環境変動に追随する．しかしながらこれまでの手法では，急激なトラヒック変動やトポロジの規模が大きくなった場合に，前の時刻のパレートフロントでは対応できず，省電力状態に収束するまでに世代数を要するという問題点がある．そこで，本稿においては前の時刻のパレートフロントからでは世代数を要するような状況においても，少ない世代数で省電力状態に進化可能な手法を確立する．本稿では，Evolvability Search [7] により導出される解と，前の時刻のパレートフロントを組み合わせた集合を初期解として，進化計算により，現在の環境に合わせたパレートフロントを探索する．

本稿の構成は以下の通りである．まず第 2 章では，パレート最適制御に基づくネットワーク省電力化手法の概要を述べる．次に第 3 章で Evolvability Search の適用について述べ，第 4 章では本提案手法のシミュレーション評価の環境とシミュレーション結果を述べる．最後に，第 5 章で本稿のまとめと今後の課題について述べる．

2. パレート最適制御に基づくネットワーク省電力化手法

2.1 概　　要

本制御では，集中制御コントローラは，定期的にネットワーク内のトラヒック観測情報を収集，各時刻のトラヒック状況に応じて，制御対象のネットワークを経路，機器の電源の投入状況を制御することにより，ネットワークの低消費電力化を行う．本計算では，各制御間隔で毎回パレートフロントを導出し，ネットワーク管理者が定めた性能，信頼性の制約条件を満たすもののうち，最も消費電力が小さい解を投入する．これにより，ネットワークの性能，信頼性の条件を満たしつつ，消費電力を最小化することが可能となる．

2.2 ネットワーク構成の進化

本制御では，消費電力，性能，信頼性など複数の目標を考慮する．したがって，適切なネットワーク構成は，そのような複数目的のパレート最適解である．しかし，目的関数は現在の状況に依存する．たとえば，ネットワークの性能はトラヒックに依存する．すなわち，以前のタイムスロットにおけるパレート最適解は，現在のタイムスロットにおけるパレート最適解ではない可能性が高い．したがって，パレート最適解は，各タイムスロットについて計算されるべきである．

本稿では，以前のパレート最適解を進化させ，現在のタイムスロットに対するパレート最適解を求める．環境変化が小さい場合は，直前の解を進化させることによって，現在のパレート最適解がすぐに導出される．大きな変動が生じ，現在のパレート最適解が以前の解から大きく離れている場合，その後進化した解は現在のパレート最適から遠いままの可能性がある．しかし，このような場合であっても進化を続けることによって，現在のパレート最適の近傍の解を得る．

また本稿では，電源投入されるリンクとノード，およびノード間の経路を設定するという問題を解く．多くのノードを経由する経路が多くのリソースを消費するため，すべてのノードとリンクの電源がオンになっているネットワークトポロジ上で k-shortest path によって事前に計算された経路候補から，ノードペアの経路を選択する．k-shortest path に焦点を当てたネットワーク構成を進化させることにより，長い経路は回避され，進化したネットワーク構成は適切な経路に近づく．

各タイムスロットでは，MO-EA に基づいて候補ネットワーク構成を進化させる．各タイムスロットでは次のステップを複数回実行する．

（1） 評価：非優越ソートに基づいてネットワーク構成をランク付けし，各ランクの解の密度を計算する．

（2） 子孫の生成：ネットワーク構成である個体に対して遺伝子操作を行うことで子孫を生成する．

（3） 解の淘汰：ネットワーク構成のセットを更新し，ランク 1 の解をパレートアーカイブに保存する．

最後に，次のタイムスロットのために解をパレートアーカイブに保存する．

2.2.1 評　　価

候補となる各ネットワーク構成は，目的関数の値を計算することによって評価される． 次に，非優越ソートによって候補構成をランク付けする．ランク 1 の解は，パレートフロントに最も近い解となる．また，各解の混雑距離も計算する．

次に，ランク 1 の解から，他の制約を満たす解の中で消費電力が最も小さい 1 つの最適解を選択する．選択された解を**優先解**と呼ぶ．優先解は，現在の候補構成の中で最適なネットワーク構成となる．

a） 目 的 関 数

我々の手法ではあらゆる目的関数を適用できるが，以下の 3 つの目的関数に基づいて議論する．

消 費 電 力

本稿では，ネットワークの消費電力は，電源投入されたリンクとノードの消費電力の合計として定義される．

$$E^{\mathrm{net}}(x) = \sum_{(i,j)\in L} E^{\mathrm{Link}} p_{i,j}(x) + \sum_{k\in V} E^{\mathrm{Node}} p_k(x) \qquad (1)$$

ここで L はリンクの集合，V はノードの集合，E^{Link} は各リンクの消費電力，E^{Node} はそれぞれのリンクの消費電力，$p_{i,j}(x)$ は，遺伝子 x が i と j の間のリンクを使用する場合は 1，その他の場合は 0，$p_k(x)$ は x がノード k を使用する場合は 1，それ以外の場合は 0 になる変数である．

信 頼 性

障害発生時に接続を維持するための 1 つの方法は，電源投入されたリンク上の独立なバックアップ経路を準備することである．障害が発生した場合，障害の発生したリンクを通過するフローの経路は，独立なバックアップ経路に変更される． 新たにノードの電源を投入するには時間を要するが，これにより，障害が検出された直後に新たにノードに電源を投入することなく接続を維持することが可能である．

より多くのバックアップ経路を用意することで，大きな障害に対しても接続性を保つことが可能となる．したがって，電源が投入されたリンク上の独立な経路の数に基づいて信頼性の指標を定義する．信頼性は，以下のように定義される．

$$R(x) = \frac{1}{\min_{i,j} r_{i,j}(x) + \alpha \sum_{i,j} r_{i,j}(x)}$$

$r_{i,j}$ は i と j の間の独立な経路の数であり，α はパラメータである．この評価では，α を $\max_x \sum_{i,j} r_{i,j}(x)$ に設定する．

故障が発生しない場合には k-shortest path から経路を選択するが，k-shortest path に含まれない経路の独立な経路の保持を許容する．このようなバックアップ経路を許容することによって，必要な独立なバックアップ経路の数を確保しながら，より多くのノードとリンクをオフにすることが可能となる．

性 能

本稿では，性能の指標としてデバイス間のリンク利用率を使用する．つまり，性能 $P(x)$ の目的関数は，次のように定義される．

$$P(x) = \max_{(i,j)\in L} \rho_{i,j}(x)$$

ただし，$\rho_{i,j}(x)$ はリンク i–j の利用率である．

2.2.2 子孫の生成

本制御では，MO-EA によって適切なパレートフロントを導出する．MO-EA では，以下のように選択，交叉および突然変異によって新しい子孫を生成する．

選択と交叉：トーナメント戦略に基づいてネットワーク構成を選択する．トーナメント戦略では，k 個のネットワーク構成をランダムに選択し，それらの内から最適な個体を選択する．最適な個体は以下のように選択される．選択された k 個のネットワーク構成に優先解が含まれている場合は，優先解が選択される．それ以外の場合は，解のランクを比較し，ランクが最も小さい解を選択する．ランクが最も小さい解が複数存在する場合は，混雑距離を比較し，ランクが最も小さい解の中で最大の混雑距離を持つ解を選択する．

上記の手順を 2 回実行することにより，2 つのネットワーク構成を選択する．ネットワーク構成には，ノード間のパスと電源投入されたリンクに関する情報が含まれる．次に，選択した 2 つのネットワーク構成において，ランダムに選択された経路を交換して新しい子孫を生成する交叉を実行する．選択した経路を交換した後，オン状態のリンクの情報を，構成に含まれるリンクの電源がオンになるように変更する．

突然変異：突然変異は，前世代の候補とは異なるネットワーク構成の候補を生成する．本稿では，突然変異は 2 つの役割をもつ．以下のようにノードをシャットダウンするほか，負荷を分散することによってより多くのトラヒックを収容するネットワーク構成を生成することによって，省電力のネットワーク構成を生成する．

- ノードのシャットダウン：オン状態のノードを 1 つ選択する．次に，すべての経路に選択したノードが含まれないように経路を更新する．フローの経路に選択されたノードが含まれている場合，電源投入されたノードのみを含む新しい経路が，経路候補からランダムに選択される．オン状態のノードのみで構成される経路の候補が存在しない場合は，生成された構成を破棄する．
- 負荷分散：オン状態のリンクの中で最も混雑しているリンクを選択する．ネットワークの性能を評価するために，各リンクでリンク利用率が計算されているため，この情報を利用する．そして，対象リンクを含む経路が破棄され，更新が必要な各フローの新しい経路は，フローの経路候補からランダムに選択される．

2.3 ネットワーク構成の選択

本稿では，他の目標の制約下で消費電力を最小化することを目指す．そこで，制約条件を満たす候補の中で消費電力が最小となる構成を選択する．選択されたネットワーク構成は，消費電力のみを最小化する解であるが，候補となるネットワーク構成がパレートフロントに近づくように更新されるため，他の目的に対してもほぼ最適な解である．

3. Evolvability Search の適用

3.1 Evolvability Search [7]

本稿では，パレートアーカイブに加えて，Evolvability の高い解をアーカイブすることを考える．Evolvability は生物の進化における重要な要素の一つであり，Evolvability が進化過程を全体的に加速することに加えて，絶滅を回避または新しいニッチを確立することによって生物種に利点を与えるものであるされている [8]．一般的に適応度関数が 1 つの目的のみに基づく静的な計算の場合，Evolvability が増大する必要はないとされている．むしろ進化的アルゴリズムにおけるこのような Evolvability の再現は，アルゴリズムの性能を低下させるとされてきた．しかしながら，複数の目的や動的な計算を行う必要がある場合には，Evolvability が必要とされる．

文献 [7] では，進化的アルゴリズムにおいても Evolvability を再現するために，*Evolvability Search* という方法によって，

Evolvability の高い解を生み出す．具体的な方法は，適応度としてより Evolvability の高い解を直接選択するというものである．このとき Evolvability は，ある個体から生成された子孫（後に廃棄される）のサンプル間の表現型の変動性を計算することにより導出される．変動性はユニークなふるまいの数として定量化する．各子孫は逐次的に考慮され，その挙動が既にリストにある個体のふるまいと著しく異なる場合にのみ，ユニークなふるまいのリストに追加される．すなわち，以下の手順で個体 i の Evolvability を決定する．

（1） 個体 j をランダムに選択する．

（2） 個体 i と個体 j で遺伝子操作を行い，個体 k, l を生成する．

（3） 個体 k, l がユニーク個体であるかを指標 $B(x)$ で判定する（$x \in \{k, l\}$）．ここでユニーク個体であればリスト U_p に追加し，ユニーク個体でなければ破棄する．

（4） 生成した i の子孫が一定数に達していた場合は手順 5，それ以外は手順 1 へ．

（5） 個体 i の Evolvability を U_p の個体数として終了する．

文献 [7] では，ロボット領域での適用を行い，$B(x)$ はロボットの特定領域の行動距離によって計算され，$B(x)$ が予め定義された閾値を超える場合，2 つのふるまいは異なるとみなされる．文献中では，Evolvability Search を用いることで，従来の目的ベースの探索アルゴリズムで生成されたものよりも高い Evolvability を有する解が生成されることを示している．また実験の中では，生み出された Evolvability が一般的であるかを検証するため，Evolvability Search を実行した環境から新しい環境に移入し，移入後も Evolvable であることを実証している．したがって，Evolvability Search は解の Evolvability を著しく増大させ，一般化可能な Evolvability を生じさせることが可能である．本稿では Evolvability Search によって導出される解を Evolvable Solution（ES）と呼ぶ．

3.2 提案手法の概要

3.2.1 制御方法

ES を保持してパレートフロントと共に用いることで，進化が促進され急激な変動後にも追随して収束が可能になることが期待される．しかしながら文献 [7] 中で述べられているように，Evolvability Search は，ある個体の Evolvability を推定するために十分な子孫サンプルを評価しなければならないという点で，計算に時間を要する手法である．そのため本制御を現実的な時間間隔で行う場合，この方法を進化的アルゴリズム自体に組み込むことは避ける必要がある．そこで，図 1 に示すような制御方法を提案する．

本提案手法では，制御開始以降 Evolvability Search を並行して実行し，ある間隔で ES のアーカイブを更新する．ただし，Evolvability を適応度として進化させることで，独立な経路などを考慮せず遺伝子操作が進むため，ES のアーカイブとパレートアーカイブの解を合わせた集合においては，パレートフロント計算時に ES が淘汰されてしまい進化が促進されないといった状況を引き起こす．そのため，効果的に ES を導入するためにはパレートフロントから離れすぎてはならない．そこで，Evolvability Search を実行する際には，毎回パレートフロントを初期解として ES を計算する．パレートフロントを初期解として計算を開始することで，Evolvability Search はパレートフロントから離れない範囲で Evolvability の高い解を探索することが可能になる．またある間隔で毎回初期解をパレートフロントとすることで，ノードの故障や追加といったトポロジの更新情報を取り込む．

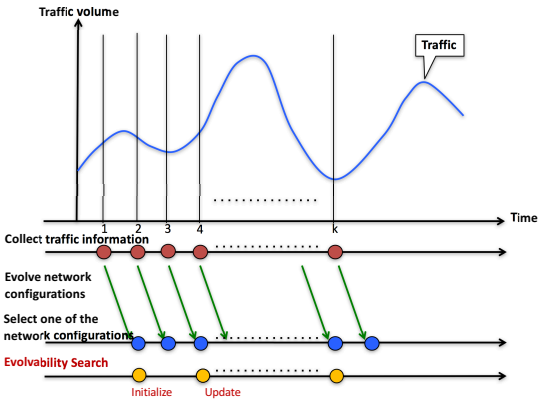


図 1 提案手法の概要

3.2.2 ES の計算

Evolvability の計算は問題依存であるため，本稿においてはネットワークに関する指標で決定する．3.1 節に示すように，Evolvability は親個体から生まれるユニーク個体の数で定量化されるが，ここではユニーク個体の判断，すなわち 1 世代進化させた個体が，どれだけ異なる状況の個体になるかをネットワークの指標で定める．ES に期待する性能は，環境変動により前のパレートフロントからでは制約を満たした状態および省電力状態への収束に時間がかかるような状況において，進化を促進することである．ボトルネックとなっているリンクを流れるフローを他のリンクを経由するように変更できれば，制約を満たす状態に落ち着くことができる．また，フローの流れないリンクを作り出す一方で，混雑していないリンクにフローが集約できれば，省電力化が促進される．このような性能を個体に求める場合，進化後の各個体のボトルネックリンクがどのように変化するかを確認しなければならない．進化により生成されるそれぞれの個体が様々なリンクでボトルネックを形成するのであれば，親個体はフローを柔軟に集約・分散できるものとみなせるため，あらゆるトラヒックに対応して進化適応する可能性が高い．ただし，ボトルネックとなりうるリンクが変化したとしても，別のリンクを同じフローの組み合わせが経由する場合は，当該リンクで新たな輻輳が発生してしまうため，輻輳を回避できない．したがって，ボトルネックリンクとなりやすいリンクがどのリンクかではなく，どのフローがボトルネックとなるリンクを経由するかまで考慮して，個体のユニークさを評価する必要がある．

本稿では，ユニークとみなした個体をリスト U_p に追加していきながら，新たな個体 x がユニークであるかを以下の式により評価する．

$$B(x) = \min_{g \in U_p} b(x, g)$$

ただし，$b(x, g)$ は個体 x と個体 g のふるまいの差を示し，以下の式で定義される．

$$b(x, g) = \|\boldsymbol{b_x} - \boldsymbol{b_g}\|_2$$

ただし，フローの集合 F に対して $\{1, ..., n\} \in F$ であるとき，$\boldsymbol{b_x} = (b_x(1), b_x(2), ..., b_x(f), ..., b_x(n))$ であり，$b_x(f)$ は x におけるフロー f がどれだけ混雑したリンクを経由しているのかを

表 1　評 価 環 境

ノードの消費電力	0.7 [kw]
リンク 1 本あたりの消費電力	0.07 [kw]

表 2　進化計算におけるパラメータ

母集団の個体数	交叉率	突然変異率	ES および RS の個体数
50	0.5	0.5	50

示す値であり以下の式で表される．

$$b_x(f) = \min_{k \in L_f} \frac{C(k)}{N(k)}$$

ただし，L_f はフロー f において経由するリンクの集合，$N(k)$ はリンク k を経由するフロー数，$C(k)$ はリンク k の Capacity である．

4. 評　　価

4.1 評 価 環 境

4.1.1 ネットワークトポロジ

Waxman モデルによって生成された PoP（Point Of Presence）レベルのトポロジをもとに，各 PoP 内に複数のコアルータを配置，各アクセスルータはいずれのコアルータとも接続するようなトポロジを構成し，評価に用いた．このトポロジでは，コアルータの電源を落としても接続性の確保ができるため，トラヒック量に応じて省電力化が可能である．本評価では，フロー数 90，380 のトポロジを評価に用いた．

また，ノードの消費電力はリンクの 10 倍程度であることにもとづき [4]，各機器の消費電力は，表 1 に示される値を用いた．

a） トラヒック

本稿では 2 パターンのトラヒック変動を用意した．

- 定常変動：総トラヒック量はほぼ変化しないが，各アクセスルータ間のトラヒックが全てランダムに変化
- 突発変動：総トラヒック量が 2 倍に変化し，各アクセスルータ間のトラヒックも全てランダムに変化

4.1.2 比 較 手 法

本評価では，以下の 3 つの場合を比較する．

ES あり　（w/ ES：with Evolvable Solution） 本研究で提案する，前の時刻のパレートアーカイブと ES のアーカイブをもとに初期解を生成し，進化計算を行うことにより，パレート最適制御を行う手法．

ランダム解あり　（w/ RS：Random Solution） 前の時刻のパレートアーカイブとランダムに生成した解を初期解とし，進化計算を行うことにより，パレート最適制御を行う手法．

ES と RS なし　（w/o ESRS：without Evolvable Solution nor Random Solution） 前の時刻のパレートアーカイブを初期解とし，進化計算を行うことにより，パレート最適制御を行う手法．

本評価では，これらの手法を評価に用いることにより，環境変動に追随して消費電力を抑えることができることを明らかにする．いずれの手法においても，表 2 のパラメータを用い，進化計算を行う．

4.1.3 SLA

制御の際に達成が必要な性能条件として，最大リンク利用率を 0.4 以下にするという条件を与えた．また，耐故障性に関する制約条件として，全機器間に 2 本以上の独立な経路を確保するという条件も満たすという条件も与えた．

4.1.4 収束の定義

本評価では，適切な解を達成するために必要な世代数を評価するために，2 つの適切な解を定義した．

- 制約を満たす解：全ての制約を満たした解
- 省電力解：全ての制約を満たしかつ十分に省電力な状態の解

消費電力はトポロジとトラヒックに依存して抑えることができる．そこで，提案手法で十分な世代数計算して導出された解の消費電力によって省電力解を決定する．本稿では，500 世代消費電力が改善されなかったときの消費電力を最小電力 ($P^{\min}$) として定義する．このとき，$(P^{\max} - P^{\min})(1-\epsilon)$ の消費電力を削減する解を省電力解として定義する．ただし $P^{\max}$ は最大消費電力，ϵ はパラメータである．本評価では ϵ を 0.1 とする．

4.2 評 価 結 果

本評価では，トラヒック変動後の収束までの世代数を評価するために，変動直前のパレートフロントを統一する必要がある．そこで，フロー数 90，380 のトポロジにおいて変動直前の環境で十分に進化させたパレートフロントをそれぞれ 3 つずつ用意した．

図 2 は，フロー数 90 において 3 つのパレートフロントからそれぞれ Evolvability Search を実行した際の Evolvability の遷移を示したものである．各世代の計算には時間を要する一方で，Evolvability Search によって Evolvability の高い解が直接選択されることで Evolvability が急増することから，ES の計算は少ない世代数で計算可能であると言える．したがって，数日間隔で ES を更新することを考えた場合には，十分に制御可能である．

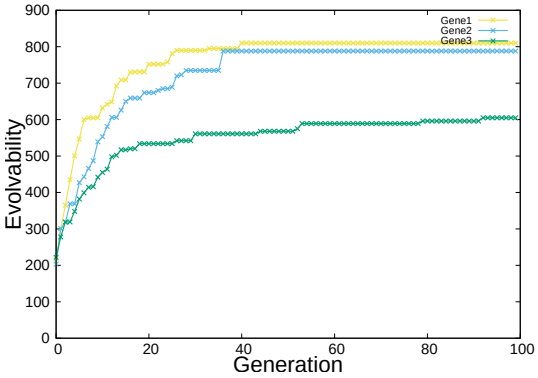
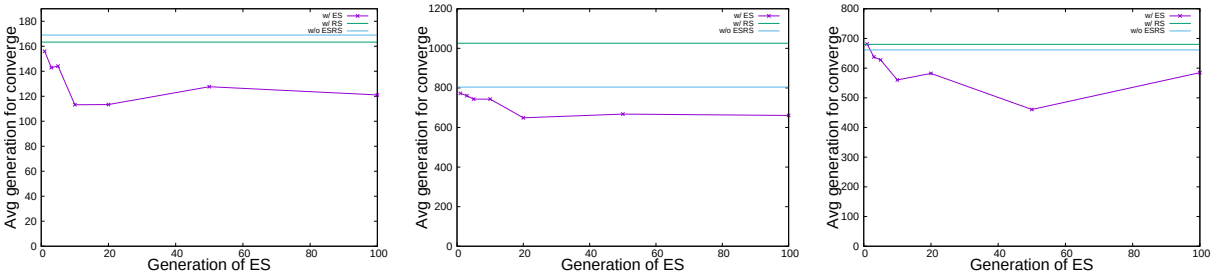


図 2　Evolvability の遷移

図 3 は，横軸が w/ ES における ES の世代数を示しており，縦軸が省電力解に至るまで要した世代数の平均値を示したものである．図より，w/ ES は ES の世代が増加するにつれて w/ RS，w/o ESRS よりも省電力解に至るまでの世代数は少なくなる傾向にあるとわかる．ただし，ES の世代数が大きすぎると，独立な経路などを考慮しない変異が積み重なった結果の解が多く存在し，パレートフロントを生成する際に淘汰されてしまうことが多くなるため，選択される機会が減り 100 世代の ES などは効力が低下していると考えられる．また，図 3(a) は 10 世代の ES が，図 3(b) では 20 世代の ES が最も効果的であるというように，ES もパレートフロントも同じであるにも関わらず，変動の大きさによって効果的に作用する ES に差異が



(a) 定常変動（Flow=90）　(b) 突発変動（Flow=90）　(c) 定常変動（Flow=380）

図 3　収束までの平均世代数

表 3　制約を満たした解に収束するまでの世代数

(a) 定常変動（Flow=90）

	w/ ES	w/ RS	w/o ESRS
Avg.	1	1	1
Max	1	1	1
Min	1	1	1

(b) 突発変動（Flow=90）

	w/ ES	w/ RS	w/o ESRS
Avg.	3.93	2.93	3.96
Max	14	8	29
Min	1	1	1

(c) 定常変動（Flow=380）

	w/ ES	w/ RS	w/o ESRS
Avg.	1	1	1
Max	1	1	1
Min	1	1	1

見られることがわかる．これは，変動後の適切なパレートフロントと前の時刻のパレートフロントがどれだけ離れているかに起因する．図 3(b) の結果から，トラヒック変動が大きい場合には適切なパレートフロントに対して前の時刻のパレートフロントはかけ離れていることがわかる．したがって，前の時刻のパレートフロントは省電力解への収束には効果的に作用せず，ES においては前の時刻のパレートフロントへの近さよりも，Evolvability の高さが進化の促進に作用していると考えられる．また図 3(c) のようにトポロジの規模が大きくなった場合にも，図 3(a) の場合と同様の傾向が確認された．ただし，トポロジの規模が大きくなると，解空間が広くなる一方で省電力解は少ないため，探索には時間を要する．

表 3 は，制約を満たした解に到達するまでの世代数を示している．w/ ES に関しては，図 3 において最も性能が高かった世代の ES を用いた際の結果である．総トラヒック量が変化しない時には，全ての手法において 1 世代で制約を満たした状態に遷移することがわかる．一方で，総トラヒック量が 2 倍になるような大きな環境変動が起きた際には，w/ ES で最大 14 世代を進化に要する．制約を満たした状態への遷移に世代数を必要とするのは，前の時刻のパレートフロントが適切なパレートフロントに対して大きくかけ離れており，トラヒックを収容するために経路の構成を大きく変更する必要があるためである．しかしながら平均世代数より，どの手法においても約 4 世代で制約を満たした状態に収束可能であることがわかる．

5. おわりに

本稿では，突発的な環境変動に追随して，十分な通信性能，信頼性の確保と低消費電力化の 3 つの目的を達成するネットワーク制御手法を確立した．

これらの指標をすべて考慮した制御を実現する手法として，パレート最適解の集合（パレートフロント）を求め，そのうち，必要な性能・信頼性の制約を満たす解をネットワークに投入することにより，性能・信頼性の要件を満たす範囲内で，消費電力を最小化する．この制御をネットワーク内の環境変動に追随して行うために，前の時刻のパレートフロントと，Evolvability の高い解を組み合わせた集合を初期解とし，進化計算により，現在の環境に合わせたパレートフロントを探索する．これにより，少ないステップで適切なパレートフロントにたどり着くことができ，高速に省電力状態へと遷移可能である．

本稿では，提案手法をシミュレーションにより評価を行い，提案手法を用いることにより，提案手法により，突発的な環境変動に追随し，約 4 世代で性能・信頼性の要件を満たし，約 650 世代でネットワークの消費電力を低減することができることを示した．

謝辞　本研究の一部は，文部科学省科学究費補助金基盤研究 (C) 16K00125 によっている．ここに記して謝意を表す．

文　献

[1] 秋下耀介, 大下裕一, and 村田正幸, “パレート最適制御によるネットワーク省電力化手法,” in **信学技報**, vol. 116 of *IN2016-35*, (北海道), pp. 73–78, 7 月 2016. 2016 年 7 月 15 日 (金) － 7 月 16 日 (土)　松前町町民総合センター (IN, NV).

[2] Cisco, “Cisco global cloud index:forecast and methodology, 2014-2019,” tech. rep., Cisco Systems Inc., Oct. 2015.

[3] Van Heddeghem, Ward and Lambert, Sofie and Lannoo, Bart and Colle, Didier and Pickavet, Mario and Demeester, Piet, “Trends in worldwide ICT electricity consumption from 2007 to 2012,” *Computer Communications*, vol. 50, pp. 64–76, Sep. 2014.

[4] Amaldi, E. and Capone, A. and Gianoli, L. G., “Energy-aware IP traffic engineering with shortest path routing,” *Comput. Netw.*, vol. 57, pp. 1503–1517, Apr. 2013.

[5] Giroire, Frédéric and Moulierac, Joanna and Phan, Truong Khoa and Roudaut, Frédéric, “Minimization of network power consumption with redundancy elimination,” *Computer communications*, vol. 59, pp. 98–105, Mar. 2015.

[6] Kessaci, Yacine and Melab, Nouredine and Talbi, El-Ghazali, “A pareto-based metaheuristic for scheduling HPC applications on a geographically distributed cloud federation,” *Cluster Computing*, vol. 16, pp. 451–468, Sep. 2013.

[7] H. Mengistu, J. Lehman, and J. Clune, “Evolvability search: directly selecting for evolvability in order to study and produce it,” in *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference*, pp. 141–148, ACM, 2016.

[8] J. Brookfield, “Evolution: the evolvability enigma,” *Current Biology*, vol. 11, no. 3, pp. R106–R108, 2001.