一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
CS2020-57(2020-11)

# NDNにおけるサービスファンクション配置アルゴリズム

徳永　嘉裕†　　中里　秀則††

† 早稲田大学大学院基幹理工学研究科情報理工・情報通信専攻
〒 169–8555　東京都新宿区大久保 3–4–1
E-mail: †kayuu.fai@toki.waseda.jp, ††@nakazato@waseda.jp

**あらまし**　効率的な IoT ネットワークの実現に向けて，IoT サービスの様々な処理 (ファンクション) をルータに配置し実行するシステムにおいて，ルータの次数を考慮したファンクション配置アルゴリズムを提案している．
**キーワード**　NDN，サービスファンクションチェイニング，ファンクション配置アルゴリズム

# Service Function Placement Algorithm in NDN

Kayu TOKUNAGA† and Hidenori NAKAZATO††

† Department of Computer Science and Communications Engineering,
School of Fundamental Science Engineering, Waseda University
Okubo 3–4–1, Shinjuku-ku, Tokyo, 169–8555 Japan
E-mail: †kayuu.fai@toki.waseda.jp, ††@nakazato@waseda.jp

**Abstract**　In order to realize an efficient IoT network, we have proposed a function placement algorithm that considers the degree of the router in a system that allocates and executes various processes (functions) of the IoT service to the router.
**Key words**　NDN，service function chaining，function placement algorithm

## 1. ま え が き

現在，IoT デバイスの急増により，通信トラフィックが増加して，ネットワークやクラウドに大きな負荷がかかることが予想されている．この負荷は低レイテンシを必要とする IoT サービスの運用に大きな問題となる，この問題の解決策として，IoT デバイス付近のルータに計算リソースを配置し，ネットワークエッジから様々な処理を行うエッジコンピューティングが提案されている．しかし，大規模な IoT ネットワークにおいてはこのエッジコンピューティングだけでは不十分である．そこで，我々は IoT サービスの様々な処理 (ファンクション) を連携させることでネットワークの負荷を図るサービスファンクションチェイニング (SFC) の活用を提案している．また，Named Data Networking (NDN) を通信プロトコルとして用い, 名前ベースでの宛先指定により，柔軟なファンクションの割り当てを行う．本研究の研究課題として，「ファンクション配置問題」と「ファンクション選択問題」が存在するが，本研究は前者の「ファンクション配置問題」を取り扱い，ルータの次数に着目したファンクション配置アルゴリズムを提案し，SFC の効率化を目指す．

## 2. サービスファンクションチェイニング

### 2.1 サービスファンクションチェイニング (SFC)

End-to-End なネットワークサービスは様々なソフトウェアを組み合わせることでコンテンツ毎のサービスを実現している．SFC はソフトウェアによってネットワークの様々な機能を実現させ，これらを組み合わせることで必要なネットワークのサービスを実現させる技術である．

これまではネットワーク機能を専用ハードウェアで実現させており，サービスの更新などに手間が生じていた．しかし，Network Function Virtualization (NFV) 技術の登場により，ネットワーク機能を汎用ハードウェアで実現できるようになった．この NFV 技術によって汎用サーバにソフトウェアとして実装されたネットワーク機能を Virtual Network Function (VNF) と呼ぶ．また，NFV 技術によって，サービス要求の状況やネットワーク状況に応じて柔軟に VNF の起動が可能となった．このように柔軟な VNF の起動制御がされている中で, 柔軟なパケット転送の制御が可能な SFC は効率的なネットワークサービスを実現させるにあたって非常に重要な技術である．現在，SFC 技術の標準化が IETF (Internet Engineering Task Force) によって進められており [2]，他にも SFC 技術に関連す

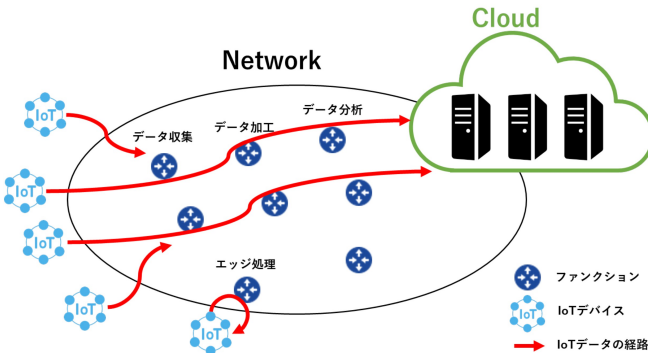


図 1　IoT データの SFC の例

る研究がされている．

### 2.2　IoT サービスにおける SFC

現在の SFC は基本的にミドルボックスのネットワーク機能 (ロードバランサ，Fire Wall 等) に着目している．本研究では SFC をより広い範囲なアプリケーション層に適応させ，その中でも IoT アプリケーションへの適応を考える．IoT サービスは様々なファンクションを組み合わせることで実現しており，基本的にクラウドでこれらのファンクションの処理を行っている．しかし，通信トラフィックが増加する中で全 IoT データをクラウドのみで処理することはクラウドとネットワークにとって大きな負荷となる．そこで，本研究はファンクションをネットワーク内のリソース (ルータ等) にも配置することを考える．このファンクションはサービス要求の状況やネットワーク状況に応じて柔軟に起動が可能であり，SFC を用いて適切なファンクションを経由するように IoT データを処理させる．このように IoT デバイスからクラウドまでのネットワーク内での処理を可能にすることでクラウドにかかる負荷や遅延を軽減させる．例として，センサのなどの IoT デバイスから取得するデータを分析する IoT サービスを考えると，ファンクションとして，「データ収集」，「データ分析」，「データ加工」と分けることができる．これらのファンクションをルータなどに配置し，必要な IoT データを SFC 技術で適切なファンクションを経由するように処理させることで IoT デバイスからクラウドまでのネットワーク内で必要な処理を完了させる．さらに，イベント検知やエッジ処理などのファンクションは IoT データを処理した結果に応じて，クラウドまで IoT データを送ることなくアクチュエータに何か指示することもできる (図 1)．本研究では，NDN と呼ばれるコンテンツ指向な通信プロトコルを用いて，IoT データの SFC を実現させる．

### 2.3　関 連 研 究

NDN における SFC の関連研究について，柔軟なファンクションチェイニングが可能な ICN ベースのフレームワークである ICN-FC が提案されている [3]．この ICN-FC のファンクションは基本的に映像処理や医療シミュレーションなどのアプリケーション層における機能を指す．ユースケース例として，あるスタジアムを複数のカメラで撮影して，ユーザが撮影された複数カメラを視聴するというケースがある．最初に複数の映像を一つの映像するために複数の映像を組み合わせるファンクションの処理を行い，映像圧縮の処理を行うファンクションにより，ユーザに適したデータを届けることができる．ICN-FC はファンクション名とデータ名との区別をするために，ICN のネーミングに「←(矢印)」を用いている．例として，あるデータ A をファンクション B とファンクション C で処理するリクエストを考えると，ICN-FC では ICN の interest パケットに対して，/C←/B←/A と名前付けを行う，このリクエスト名はデータ A をファンクション B で処理した後にファンクション C で処理することを示す．ICN-FC はこれまでのミドルボックスなネットワーク機能ではなく，アプリケーション層の機能を ICN で SFC 技術を扱う点において本研究に似ている．本研究と異なる点は元のデータと対するファンクション名を一元化せずにデータ名とファンクション名を二分化して考える点である．このデータ名とファンクション名を二分化して考える詳細については 3.4 章で述べる．

## 3.　Named Date Networking

### 3.1　Named Date Networking(NDN)

現在のインターネットの通信はデバイスなどの位置情報を IP アドレスとして表すロケーション指向な通信で設計されている．しかし，現在のインターネットの使用目的の多くはコンテンツの取得であり，現在のインターネットアーキテクチャと使用目的の間に大きなギャップが存在する．そこで，このギャップをなくすために新たなインターネットアーキテクチャが考案されており，それをコンテンツ指向型ネットワーク (ICN) と呼ぶ．ICN に関する研究は多く存在し，NDN はその中でも最も研究されている通信プロトコルである [4], [7]．

### 3.2　NDN の特徴や概念

インターネットサービスではネットワークレイヤのプロトコルとして Internet Protocol (IP) を使い，IP を軸にした砂時計型のモデルとなっている．砂時計型のモデルとなっていることから，上位層から下位層まで独立して発展することができ，インターネットが飛躍的に成長した．NDN はこの IP のスリムな形を応用させ，通信単位を位置情報のみではなく，オブジェクト名も用いる (図 2)．つまり，NDN は「アドレスに届ける」のではなく「名前ベースでデータを取り出す」通信である．さらに，NDN は IoT における SFC にも適した通信プロトコルである．IoT デバイスを IP で管理するより名前ベースで管理した方が簡素化した管理ができる．また，ファンクションへのリクエスト要求もファンクション名を用いるため，ファンクションの配置が動的に変化しても柔軟に対応ができる．

### 3.3　NDN のアーキテクチャ

#### 3.3.1　NDN のパケット

NDN の通信では Interest パケットと Data パケットと呼ばれる 2 種類のパケットを用い，コンテンツ名を元にしたパケットのやり取りによって通信が実現されている．Interest パケットとはユーザから要求されたデータを識別する名前を含んだコンテンツ要求のパケットであり，Data パケットは目的のコンテンツがあるプロデューサーからコンテンツ名によって要求されたデータをユーザまで送り返すパケットのことである (図 3)．

### 3.3.2 NDN のフォワーディングプロセス

NDN では各ルータに Content Store (CS) と Forwarding Information Base (FIB) と Pending Interest Table(PIT) の三つのデータ構造を持たすことで Internet パケットと Data パケットのフォワーディング機能を実行させている (図 4).

CS は Data パケットをキャッシュするためのデータストレージであり，PIT は Internet パケットの送信元ユーザに Data パケットを返すためのルートを記録するデータ構造であり，FIB は Interest パケットの転送先を決めるためのデータ構造である．以上三つのデータ構造を踏まえて，Interest パケットと Data パケットのフォワーディングは以下の様に行われる．

- Interest パケットのフォワーディング動作

NDN のルータに Interest パケットが到着すると，まず，CS に目的のコンテンツのデータがあるかを調べ，該当データがある場合はこの時点で Data パケットを返す．該当データがない場合は PIT を参照し，目的のコンテンツ名に該当する PIT エントリを調べる．該当エントリがある場合は，すでに同じ目的のコンテンツ名を含んだ Interest パケットが上流ルータに送信されているため，この Interest パケットを上流のルータに送信する必要はなく，該当したエントリにこの Interest パケットを受信したインターフェイスの情報を追加する．該当エントリがない場合， Interest パケットのエントリを PIT に作成し，Interest パケットに記されたコンテンツ名と受信インターフェイスを記録し，FIB で目的のコンテンツ名のプレフィックスに該当するエントリがあるかどうかを調べる．該当エントリがある場合は Interest パケットをエントリに記されたインターフェイスに送信する．該当エントリがない場合は，この Interest パケットを破棄，または送信元に NACK を返す．以上の流れが Internet パケットのフォワーディング動作であり，図 5 に示す．

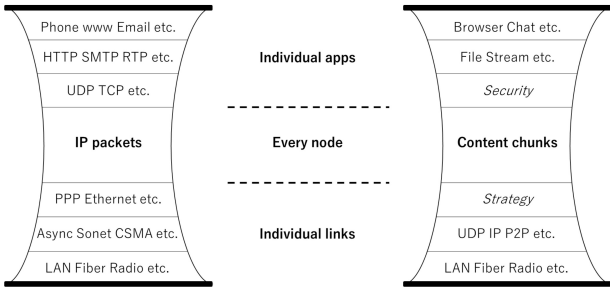


図 2　IP ネットワークと NDN のアーキテクチャ

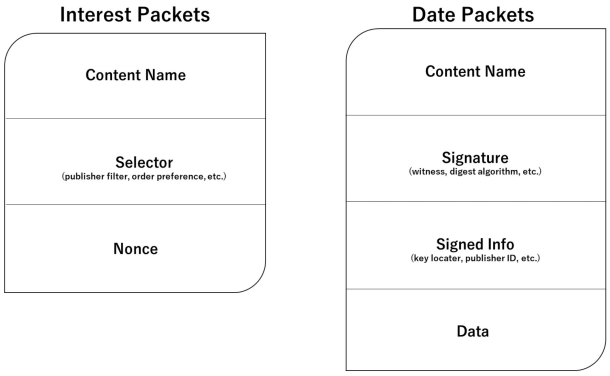


図 3　Interest パケットと Data パケットの構造

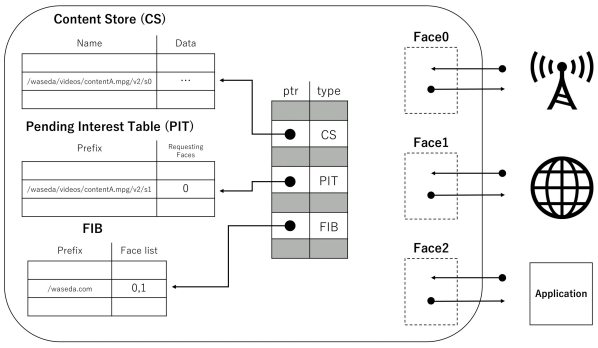


図 4　NDN におけるルータ

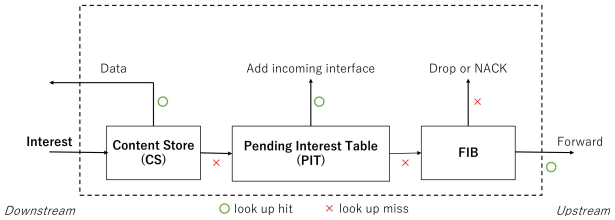


図 5　Interest パケットのフォワーディング動作

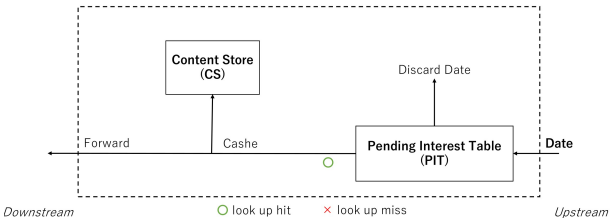


図 6　Data パケットのフォワーディングプロセス

- Data パケットのフォワーディング動作

NDN のルータが Data パケットを受信すると，まず，PIT に Data パケットに記されたコンテンツ名に該当するエントリがあるかどうかを調べる．該当エントリがない場合はこの Data パケットを必要としていたユーザは他の Data パケットにより満たされたと判断し，この Data パケットは破棄される．該当エントリがある場合はエントリ記された全インターフェイスにこの Data パケットを送信する．このとき，Data パケットを CS にキャッシュするかどうかをキャッシュポリシーに従って決める．キャッシュポリシーに該当した場合は Data パケットをキャッシュする．以上の流れが Data パケットのフォワーディング動作であり，図 6 に示す．

### 3.4 NDN における SFC

SFC はユーザから要求に応じて IoT データを必要なファンクションを経由させる技術である．つまり，SFC はファンクションで処理するパケットを適切な順番でファンクションを経由させなければならない．NDN では，送信先を決める元は Internet パケットに記されたコンテンツ名である．そこで，NDN に SFC を組み合わせるためには，パケットが特定のファンクションを適切な順番通りに経由できる仕組みが必要である． NDN では Internet パケットが通るルートを PIT で記録することで Data パケットの通るルートを確保している．つまり，Data パ

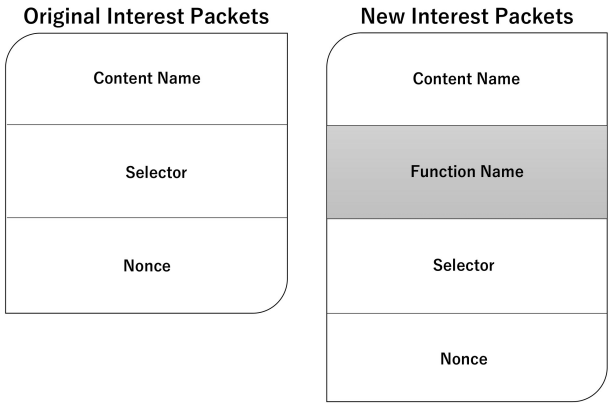


図 7　NDN-FC の Interest パケット

ケットが必要なファンクションを順番通りに経由するためには，Internet パケットが通るルートで必要なファンクションを経由しておく必要があり， Interest パケットは SFC リクエストの逆の順番で経由させる必要がある．本研究では，[8] が提案した NDN による SFC の仕組みである NDN-FC を仮定し，NDN の Interest パケットに新しいフィールドとして Function Name を追加する (図 7)．

Function Name にはユーザの要求である SFC リクエストで必要なファンクションを URL のような階層構造で記述する．NDN は pull 型通信であるため，ファンクション名を記述する際は SFC で実行するファンクションの逆順で記述する．例として，ユーザがあるサービスを利用していて，ユーザの要求に対して，ファンクション f1，f2，f3 を f1→f2→f3 の順で SFC を実行すると考える．このとき「/f3/f2/f1」と Function Name に記述する．Interest パケットがルータに送信されると，まず，Function Name を参照し，ファンクション名が記される場合はコンテンツ名ではなくファンクション名でフォワーディングを行う．FIB にはコンテンツ名とインタフェースの対応関係だけでなくファンクション名とインタフェースの対応関係の記録もあり，コンテンツ名でのフォワーディング動作と同じように FIB はファンクション名と最長一致な FIB エントリで送信先のインタフェースを決める．この例の場合，f3 はファンクション名の先頭プレフィックスであるため，Interest パケットを f3 が存在する場所に送る．Interest パケットがファンクションの場所に到着した際は先頭プレフィックスを削除する．この場合，Interest パケットが f3 の存在するルータに到着した際はファンクション名から f3 を削除し，ファンクション名を「f2/f1」とする．このように，次々とファンクション名がなくなるまで必要なファンクションを通過させていき，NDN のフォワーディングに従い，Interest パケットを目的のデータがあるプロデューサーまで送信する．プロデューサーまで Interest パケットが送られたら，プロデューサーは必要なコンテンツのデータを含む Data パケットをユーザに送り返す．このときのデータパケットの通るルートは Interest パケットが通ったルートの逆となるため，適切なファンクションの順番での処理が行われる．こうして適切な順番で必要なファンクションの処理をされた Data

**Algorithm 1:** NDN における SFC のフォワーディングプロセス

```
Data: interest
function = interest.getFunction();
fibEntry = FIB.findLongestPrefixMatch(function);
nextHops = fibEntry.getNextHops(); for nextHop : nextHops do
    outFace = nextHop.getFace();
    sendInterest(outFace, interest);
end
```

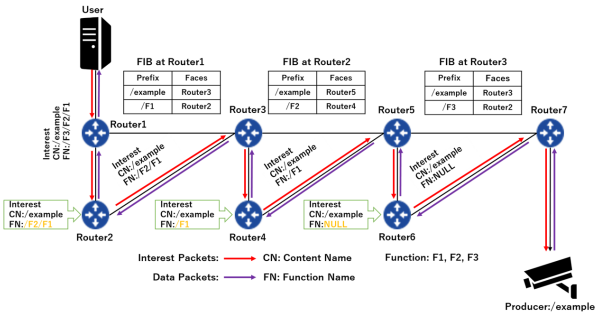


図 8　NDN における SFC のフォワーディング

パケットがユーザに届けられ，SFC が実行される．以上の一連の動作をアルゴリズムと図 8 に示す．

本研究では「ファンクション配置問題」を取り扱っているが，SFC の効率化を図るために「ファンクション選択問題」も無視することはできない．そこで，本研究で用いている「ファンクション選択手法」を紹介する．本研究では [6] で提案された手法を用いている．ファンクション呼び出し回数に着目し，ファンクション呼び出し回数が少ないファンクションも選択することで使われるファンクションの分散を図り，加えて，SFC を実行する上でパケットが経由するホップ数に着目し，ホップ数を最小限にすることで遅延を抑え，SFC サービスの実行時間の減少を図る手法である．

## 4. 提 案 手 法

### 4.1 概　　要

本提案手法はネットワーク内の個々のルータの次数（インタフェース数）と SFC リクエストにおけるファンクションの出現数に着目した手法である．使用頻度が高いファンクションを次数が大きいルータに配置する．また，本提案手法は配置するファンクションの種類とその複製数があらかじめ決められていることを前提としている．

### 4.2 ファンクション配置方法

ルータ数が $n$ 個のネットワークを考え，ネットワーク内のルータの集合を

$$R = \{r_1, r_2, ..., r_n\} \tag{1}$$

と定義する．このネットワークに $m(m \leqq n)$ 種類のファンクションを配置するものとし，ファンクションの集合を

$$F = \{f_1, f_2, ..., f_m\} \tag{2}$$

と定義する．なお，本提案では，各ルータには高々一つのファンクションが割り当てられるものとし，複数のファンクションを同一のルータに割り当てることはないものとしている．

本提案手法ではファンクションを配置する際に，$R$ における個々のルータがもつ次数の降順でファンクションをルータに配

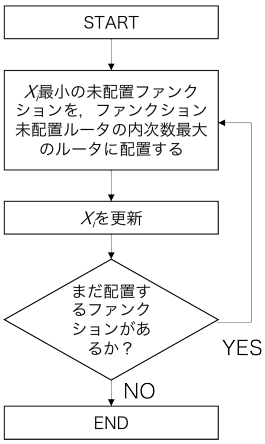


図 9　提案手法のフローチャート

置していく．すなわち，ルータ $r_j$ の次数を $|r_j|$ とすると，まず，$|r_i|$ が最大のルータにファンクションを割り当て，次に既に割り当てたルータを除いた中で，$|r_i|$ 最大のルータにファンクション割り当てるという割り当てを順次行う．

どのファンクションを割り当てるかについては以下のように決定する．$m$ 種類のファンクションがある中で SFC リクエストにおける各ファンクションの出現数を $|f_i|$ $(1 \leqq i \leqq m)$ とし，既にファンクション $i$ を配置したルータの次数の総数を $D_i$ とする．すなわち，

$$D_i = \sum_{f_i \text{を配置した } r_j} |r_j|$$

である．

ファンクション $f_i$ を配置するたびに

$$X_i = \frac{D_i}{|f_i|} \tag{3}$$

を更新する．この $X_i$ の値が小さいファンクションをルータに配置していく．ただし，まだルータに配置されていないファンクションは式 (3) で他のファンクションと比較ができるように $D_i$ の値は 1 とする．また，あるファンクションを配置する際に次数が同数のルータが複数あればランダムでどれかに配置を行う．さらに，一つのファンクションがより他の種類のファンクションとの繋がりを増やすために隣接するルータに同一種類のファンクションが配置しないように配置を行う．

この配置をすべてのファンクションが配置されるまで繰り返す．これら一連の流れのフローチャートを図 9 に示す．

## 5. 評　　価

### 5.1 シミュレーション環境

提案手法の評価は ndnSIM [5] と呼ばれる NDN のシミュレータを用いて評価する．提案手法の比較対象としてランダム配置の場合と比較し，サービス実行時間を評価する．シミュレーションでは [1] で比較例として使用された U.S-24-ノードトポロジーをネットワークトポロジーとして用いた (図 10)．さらに，U.S-24-ノードトポロジーにデータをもつプロデューサーを 2 つ，SFC リクエストを発生するコンシューマーを

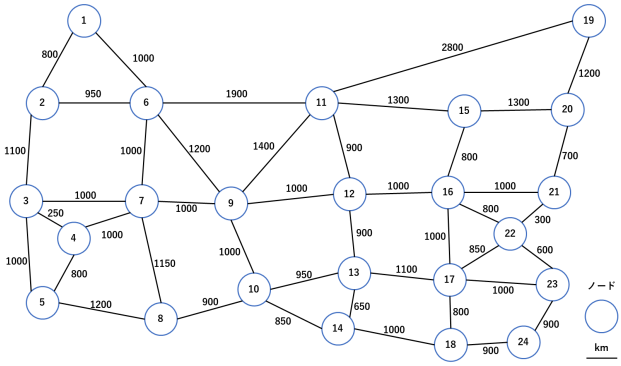


図 10　U.S-24-ノードトポロジー

表 1　SFC リクエストの種類とファンクションの出現数

| Type | Request |
|---|---|
| 1 | F1→F3→F6 |
| 2 | F1→F3→F7 |
| 3 | F1→F3→F8 |
| 4 | F2→F4→F6 |
| 5 | F2→F4→F7 |
| 6 | F2→F4→F8 |
| 7 | F3→F5→F6 |
| 8 | F3→F5→F7 |
| 9 | F3→F5→F8 |
| 10 | F1→F6→F7 |
| 11 | F1→F6→F8 |
| 12 | F2→F6→F7 |
| 13 | F2→F6→F8 |
| 14 | F3→F6→F7 |
| 15 | F3→F6→F8 |

| Function | Frequency |
|---|---|
| F1 | 5 |
| F2 | 5 |
| F3 | 8 |
| F4 | 3 |
| F5 | 3 |
| F6 | 9 |
| F7 | 6 |
| F8 | 6 |

4 つ配置した．SFC リクエストで使うファンクションとして F1,F2,F3,F4,F5,F6,F7,F8 の 8 種類を用意し，各ファンクションの複製（インスタンス）をそれぞれ 3 つずつノードで実行するものとする．これら 24 個のファンクションインスタンスを U.S-24-ノードトポロジーに配置した．また，15 種類の SFC リクエスト (表 1) を用意し，コンシューマーにランダムで発生させ．1 回のシミュレーションでの SFC リクエストの発生回数を 300 回とした．このときの SFC リクエストにおけるファンクション毎の出現数を 15 種類の SFC リクエストと一緒に表 1 に示す．ファンクション選択手法は [6] を用いおり，ファンクションの呼び出し回数と SFC を実行する際のホップ数に着目することでファンクションの負荷を分散しつつ，サービス実行時間を最小限に抑える手法である．あるルータで過去 $x$ 回のファンクション呼び出し回数のうち，あるファンクションインスタンス $j$ が呼び出される回数を $C_j$，ルータ $r_k$ からルータ $r_l$ の経路のポップ数を $H_{rl}$ で表すとする．このファンクション選択手法では，ある SFC リクエストを実行するファンクションインスタンスを，SFC 実行経路上の $C_j$ の合計と SFC 実行経路のホップ数に，それぞれ重み $\alpha$ と $\beta$ を掛けて加えた値を最小にする SFC 実行経路を選択する．本シミュレーションでは上記で説明した $x$ の値を 30，重み $\alpha$ と $\beta$ の値をそれぞれ 1 としている．

### 5.2 サービス平均実行時間

評価を行うために，提案手法での配置を 10 パターンとランダム配置を 30 パターンを用意した．さらに，各パターンにつきシミュレーションを 50 回行い，サービス平均実行時間の評

表 2　ランダム配置と比べた際の提案手法の向上率

| – | 提案手法 | ランダム | 向上率 |
|---|---|---|---|
| 10/s | 807.56 [ms] | 855.94 [ms] | 5.99 [%] |
| 20/s | 1159.33 [ms] | 1254.03 [ms] | 8.17 [%] |
| 30/s | 1382.70 [ms] | 1517.25 [ms] | 9.73 [%] |

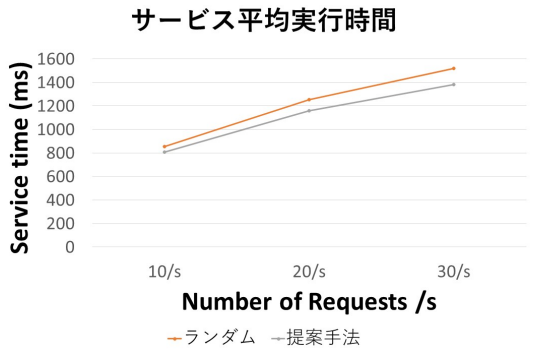


図 11　サービス平均実行時間

価を行った．サービス実行時間とはコンシューマーが SFC リクエストの Internet パケットを送り出してから，Data パケットとして返ってくるまでの時間を示す．Data パケットをファンクションが処理する時間を 40ms と設定し，SFC リクエスト頻度を 10/s，20/s，30/s の 3 パターンを評価し，提案手法での配置の 10 パターンのサービス平均実行時間を平均したものとランダム配置の 30 パターンのサービス平均実行時間を平均したものを比較し，評価を行った．その評価結果を表 2 と図 11 に示す．SFC リクエスト頻度の 3 つのパターンにおいて，ランダム配置より提案手法の方が良い結果となった．表 2 から SFC リクエスト頻度を大きくするのにつれて向上率が上昇しているため，提案手法での配置は SFC リクエスト頻度が上がるにつれてさらに効果を発揮すると考えられる．以上から提案手法での配置はランダム配置より効率的に配置できていると言える．

## 6. ま と め

本研究では IoT デバイスの増加に伴う通信トラフィックの増加によるクラウドやネットワークへの負荷の軽減のため，IoT データを NDN における SFC によりネットワーク内で処理させる際に，SFC を効率化するためにルータの次数に着目したファンクション配置アルゴリズムを提案した．本研究では使用頻度が高いファンクションを次数が大きいルータに配置するような手法を提案することで，使用頻度の高いファンクションがより多くのファンクションとの繋がりをもてるようにし，サービス実行時間を減少させることで SFC の効率化を目指した．サービス実行時間を評価した結果，提案手法はランダム配置と比べて良い結果を示した．しかし，本研究の提案手法は乱数を含むことがあるため，配置結果が複数出てしまうため，「ファンクション配置問題」の最適解に到達できていない．そのため，ルータの次数に着目するだけでなく，次数以外の着目点も絡めたファンクション配置アルゴリズムが必要なのではないかと考えている．また，ファンクションの配置と選択の 2 つの面で最適化することで，さらに柔軟で効率的な SFC の実現が期待できる．

## 文　献


[1] A. Banerjee, W. Feng, D. Ghosal, and B. Mukherjee. Algorithms for integrated routing and scheduling for aggregating data from distributed resources on a lambda grid. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 19, No. 1, pp. 24–34, 2008.

[2] Joel Halpern, Carlos Pignataro, et al. Service function chaining (sfc) architecture. In *RFC 7665*. 2015.

[3] L. Liu, Y. Peng, M. Bahrami, L. Xie, A. Ito, S. Mnatsakanyan, G. Qu, Z. Ye, and H. Guo. ICN-FC: An information-centric networking based framework for efficient functional chaining. In *2017 IEEE International Conference on Communications (ICC)*, pp. 1–7, May 2017.

[4] Named Data Networking Project. Named data networking project website. [Online]. Available: `https://named-data.net`, Last accessed on Nov. 2, 2019.

[5] ns-3 base ndn simulator . ns-3 base ndn simulator website. [Online]. Available: `http://ndnsim.net/current/`.

[6] Y. Shiraiwa and H. Nakazato. Function selection algorithm for service function chaining in ndn. In *2019 IEEE ComSoc International Communications Quality and Reliability Workshop (CQR)*, pp. 1–5, April 2019.

[7] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, KC Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. Named data networking. *ACM SIGCOMM Computer Communication Review*, Vol. 44, No. 3, pp. 66–73, 2014.

[8] 吉井宏希, 白岩善昭, 中里秀則. NDN における IoT データのファンクション・チェイニング. 2017 年電子情報通信学会通信ソサイエティ大会講演論文集, pp. B–8–1, 9 月 2017.