修士論文

ノード間の近接性を考慮した
定数次数構造化オーバレイネットワーク

白石 裕輝

2019年3月15日

奈良先端科学技術大学院大学
情報科学研究科

本論文は奈良先端科学技術大学院大学情報科学研究科に
修士 (工学) 授与の要件として提出した修士論文である。

白石 裕輝

審査委員：

門林 雄基 教授	(主指導教員)
安本 慶一 教授	(副指導教員)
笠原 正治 教授	(副指導教員)
妙中 雄三 准教授	(副指導教員)

ノード間の近接性を考慮した
定数次数構造化オーバレイネットワーク*

白石 裕輝

内容梗概

分散ライングラフ (DL グラフ) は，正則グラフとそのライングラフに基づく有向グラフで，定数次数構造化オーバレイネットワークを構築することを目的として提案された．DL グラフは，出次数が定数である，耐障害性やスケーラビリティに優れるといった性質を持つ．特に，de Bruijn グラフや Kautz グラフを初期グラフとする DL グラフは最適な直径を持つため，オーバレイネットワークのトポロジとして適切である．しかしながら，DL グラフに基づくオーバレイネットワークは，ノード間の近接性を考慮せず構築されるため，オーバレイネットワーク上の探索遅延が大きくなるという問題点がある．

そこで本研究では，DL グラフに基づく構造化オーバレイネットワークにおいてノード間の近接性を考慮する手法: 近接性を考慮した分散ライングラフ (PDL グラフ) を提案する．PDL グラフは，DL グラフにおけるノード分割とノード融合に着目し，オーバレイネットワークのトポロジの拡大や縮小の際にノード間の通信遅延を考慮して局所的にトポロジの最適化を行う．シミュレーションにより，PDL グラフに基づくオーバレイネットワークは，DL グラフに基づくものと比較して探索遅延を 10% 低減することを示した．

キーワード

近接性を考慮したルーティング，分散ハッシュ表，構造化オーバレイネットワーク，Peer-to-Peer コンピューティング

*奈良先端科学技術大学院大学 情報科学研究科 修士論文, 2019 年 3 月 15 日.

A Proximity-Aware Constant Degree Structured Overlay Network*

Youki Shiraishi

Abstract

A distributed line (DL) graph is a directed graph based on a regular graph and its line graphs. It enables to construct a constant degree structured overlay networks from an initial regular graph. DL graphs have some excellent properties, such as constant out-degree, fault tolerance, and scalability. Furthermore, they are known to have an optimal diameter, therefore, they indeed possess desirable characteristics for structured overlay networks. On the other hand, since they are constructed without considering proximity between nodes in the underlying network, their lookup latency turns out to be serious.

In this paper, I propose proximity-aware distributed line (PDL) graphs, a method for designing proximity-aware regular graph-based structured overlay networks. More precisely, PDL graphs focus on communication delays between nodes when splitting or merging nodes on DL graphs, with an aim to reduce the lookup latency. Simulation results show that the overlay networks based on our proposed scheme reduce lookup latency by 10% compared with the ones based on original DL graphs.

Keywords:

proximity-aware routing, distributed hash tables, structured overlay networks, peer-to-peer computing

*Master's Thesis, Graduate School of Information Science, Nara Institute of Science and Technology, March 15, 2019.

目 次

図 目 次

表 目 次

アルゴリズム目次

1. 序論

オーバレイネットワークは，IP ネットワークなどの上位に構築される論理ネットワークである．オーバレイネットワークは，大規模な分散データベースや，ファイル共有システム，コンテンツ配信ネットワーク，暗号通貨のブロック配信ネットワーク，データセンターなどを支える重要な技術となっている．これらの分散アプリケーションを構成するノードは，自律分散的にオーバレイネットワークを構築することで，アプリケーション全体の耐障害性やスケーラビリティを向上させている．例えば，BitTorrent や Apache Cassandra，Google Bigtable は，分散ハッシュ表 (Distributed Hash Table, DHT) と呼ばれる数学的に構造化されたオーバレイネットワークの上に構築されている．

DHT をはじめとする構造化オーバレイネットワークは，2000 年ごろから盛んに研究されており，Chord [1] や Kademlia [2] など，これまでに多くの手法が提案されてきた [3]．構造化オーバレイネットワークは，ノードに対して付与される識別子に基づいてネットワークトポロジを構築する．ノード数を n とすると，従来の構造化オーバレイネットワークでは，各ノードが $O(\log n)$ の隣接情報を持つことでネットワーク上の探索を $O(\log n)$ のクエリ転送回数で実現することができる．近年では，ネットワークトポロジのメンテナンスコストを低減するために，正則グラフに基づく定数次数構造化オーバレイネットワークが盛んに研究されている．定数次数構造化オーバレイネットワークとして代表的なものに，de Bruijn グラフに基づく Koorde [4] や D2B [5]，Kautz グラフに基づく FissionE [6] や Moore [7]，バタフライグラフに基づく Viceroy [8] や Ulysses [9] が提案されている．これらのオーバレイネットワークでは，各ノードがある一定数の隣接情報を持つことでネットワーク上の探索を $O(\log n)$ のクエリ転送で行うことができる．他にも，耐障害性や探索効率に優れることが知られている [10]．

正則グラフに基づくオーバレイネットワークの実用上の問題は，隣接ノードと探索経路が厳密に決定される点にある．オーバレイネットワーク上のノードは，世界中のさまざまな場所に存在し，それらの間のネットワーク近接性は多様である．通常のオーバレイネットワークは，下位ネットワークとは無関係に決められた識別子に基づいて隣接ノードや探索経路を決定するため，下位ネットワークにおける

ノード間の近接性を考慮しない．その結果，オーバレイネットワーク上の探索において通信遅延の大きいノードと通信を行うことが頻繁に生じ，探索遅延の増大につながる．したがって，オーバレイネットワークを実用する上で近接性の考慮は必要不可欠である．近接性に基づいてオーバレイネットワークを構築する手法として，近接隣接ノード選択 (Proximity Neighbor Selection, PNS) [11, 12, 13, 14], 近接経路選択 (Proximity Route Selection, PRS) [12, 15], 近接識別子選択 (Proximity Identifier Selection) [16] が知られている [12]．これらの手法は，オーバレイネットワークにおける最短経路だけでなく，ノード間の通信遅延や通信帯域も考慮したよりよい経路を見つけることを目的としている．PNS では，近接性に基づいて隣接ノードを，PRS では，近接性に基づいて次ホップを，PIS では，近接性に基づいてノードの識別子をそれぞれ決定する．このように，PNS や PRS，PIS は，各ノードが隣接ノードや次ホップを近接性の観点から局所的に最適化することによって，オーバレイネットワーク全体の性能を改善する．一方で，これらの手法は，隣接ノードや次ホップに選択の余地があるという仮定のもとで適用可能なものであり，正則グラフに基づくオーバレイネットワークのように隣接ノードや次ホップが厳密に決定されるオーバレイネットワークに対して適用することは困難である．

1.1 研究の目的

本研究は，任意の正則グラフに基づく構造化オーバレイネットワークにおいてノード間の近接性を考慮し，探索遅延を低減することを目的としている．これを実現することにより，正則グラフの構造による高い探索効率や優れた耐障害性といった性質と，近接性の考慮による高速な探索を両立することができる．

本研究の成果は，低遅延が求められるサービスを展開するための基盤ネットワークとしての応用が期待できる．

1.2 研究のアプローチ

本研究では，任意の正則グラフに基づくオーバレイネットワークにおいて探索遅延の低減を実現する手法として，近接性を考慮した分散ライングラフ (Proximity-Aware Distributed Line Graph, PDL グラフ) を提案する．PDL グラフは，正則グラフに基づくオーバレイネットワークを構築する手法である分散ライングラフ (Distributed Line Graph, DL グラフ) [17] に着目し，ノードの参加や離脱の際にノード間の近接性を考慮して局所的にトポロジを最適化する．具体的には，オーバレイネットワークのトポロジに対応する DL グラフを考え，ノードの参加や離脱の際に引き起こされる頂点分割と頂点融合の際に，可能なすべてのグラフ候補を列挙する．そして，その候補の中から隣接ノードの通信遅延を最小化するようなグラフを選択し，オーバレイネットワークのトポロジとする．

1.3 論文の構成

本論文の構成は以下のとおりである．2 節では，本論文で用いる基礎理論や表記について述べる．3 節では，関連研究として正則グラフに基づく構造化オーバレイネットワークとオーバレイネットワークにおける近接性の考慮手法について説明し，正則グラフに基づく構造化オーバレイネットワークにおいて近接性の考慮が困難であることを説明する．4 節では，提案手法のベースである分散ライングラフについて述べ，オーバレイネットワークの構築手法について詳細な説明を与える．5 節では，提案手法である近接性を考慮した分散ライングラフについて述べる．6 節では，提案手法の評価を行い，提案手法が探索遅延を削減することを示す．7 節では，提案手法の応用について議論を行う．8 節では，評価と応用についての議論を踏まえて，提案手法の有用性について考察する．最後に，9 節で本論文の結論を述べる．

2. 準備

本節では，論文内で用いられる用語や表記，および基礎知識について述べる．

2.1 形式言語

アルファベット Σ は，文字の有限集合である．アルファベットに含まれる文字数は，アルファベットのサイズと呼ばれ，$|\Sigma|$ と表記される．ここでは簡単のために，d 個の整数 $0, 1, \ldots, d-1$ を文字として構成されるアルファベットを $\Sigma_d = \{0, 1, \ldots, d-1\}$ と定義する．

文字列は，アルファベット上の文字の有限列である．文字列 $\boldsymbol{s} = \sigma_1\sigma_2 \ldots \sigma_\ell$ の長さは $|\boldsymbol{s}|$ と表記する．文字列 $\boldsymbol{u} = u_1u_2 \ldots u_m$ について，長さ $n \leq m$ の文字列 $\boldsymbol{v} = v_1v_2 \ldots v_n$ がある整数 i $(1 \leq i \leq m-n+1)$ に対して $\boldsymbol{v} = u_iu_{i+1} \ldots u_{i+n-1}$ を満たすとき，$\boldsymbol{v}$ は $\boldsymbol{u}$ の部分文字列という．文字列 $\boldsymbol{u} = u_1u_2 \ldots u_m$ の接頭辞とは，ある整数 k $(1 \leq k \leq m)$ に対して $u_1u_2 \ldots u_k$ となる文字列で，$\boldsymbol{u} \restriction k$ と表記する．また，$\boldsymbol{u}$ の接尾辞とは，整数 k $(1 \leq k \leq m)$ に対して $u_{m-k+1}u_{m-k+2} \ldots u_m$ となる文字列で，$\boldsymbol{u} \upharpoonright k$ と表記する．2 つの文字列 $\boldsymbol{u} = u_1u_2 \ldots u_m$, $\boldsymbol{v} = v_1v_2 \ldots v_n$ の連接は，文字列 $u_1u_2 \ldots u_mv_1v_2 \ldots v_n$ であり，$\boldsymbol{uv}$ と表記する．

長さ k のすべての文字列の集合を Σ^k，すべての文字列の集合を Σ^* とする．言語 L は Σ^* の部分集合，すなわち $L \subseteq \Sigma^*$ と定義される．

2.2 グラフ

2.2.1 有向グラフ

有向グラフ $G = (V, E)$ は，頂点集合 V と $V \times V$ の部分集合である辺集合 E から構成される．G の頂点集合を $V(G)$，辺集合を $E(G)$ と表記する．本論文では，単射性を持つ写像 $f : V \to \Sigma^*$ を暗黙的に用いる．具体的には，頂点 $v \in V$ はある文字列 $f(v) = \boldsymbol{v} \in \Sigma^*$ に一対一で対応し，v と $\boldsymbol{v}$ を同じものとみなす．

有向グラフの辺 $e = (u, v)$ について，e は頂点 u から出て頂点 v へ入るという．このとき，u を e の始点，v を e の終点と呼ぶ，また，u を v の入隣接頂点

(あるいはプレデセッサ) といい，v を u の出隣接頂点 (あるいはサクセッサ) という．始点と終点が等しくなるような辺をループと呼ぶ．

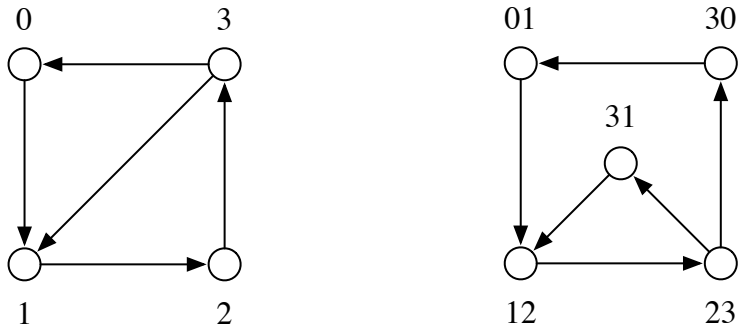
有向グラフ G の頂点 v に対して集合 $N_G^-(v) = \{u \mid (u,v) \in E(G)\}$ を v の入隣接頂点集合，集合 $N_G^+(v) = \{w \mid (v,w) \in E(G)\}$ を v の出隣接頂点集合，そして，$N_G(v) = N_G^+(v) \cup N_G^-(v)$ を v の隣接頂点集合と呼ぶ．また，$\delta_G^-(v) = \left|N_G^-(v)\right|$ を入次数，$\delta_G^+(v) = \left|N_G^+(v)\right|$ を出次数という．ある整数 $d \geq 1$ に対して，G 上のすべての頂点 v が $\delta_G^-(v) = d$ を満たすとき，G は d-入正則であるという．また，G 上のすべての頂点 v が $\delta_G^+(v) = d$ を満たすとき，G は d-出正則であるという．G が d-入正則かつ d-出正則であるとき，G は d-正則であるという．

正則有向グラフの典型的な例として完全有向グラフとループ付き完全有向グラフがある．完全有向グラフとは，G 上の各頂点 u が u 以外のすべての頂点 $v \in V(G) \setminus u$ への辺 (u,v) を持つような有向グラフである．ループ付き完全有向グラフとは，完全有向グラフのすべての頂点にループを加えたものである．頂点数が n の完全有向グラフを K_n，ループ付き完全有向グラフを K_n^+ と表記する．K_n は $(n-1)$-正則で，K_n^+ は n-正則である．

簡単のために，$(u,v) \in E(G)$ を $u \to v$ と表記する．また，ℓ を 1 以上の整数とする．$W : u = w_1 \to w_2 \to \cdots \to w_{\ell+1} = v$ を u から v への経路という．W に含まれる辺の数をその長さといい，$|W|$ と表記する．u から v への経路のうち経路長が最小となるものを u から v への最短経路といい，その長さを $\mu_G(u,v)$ と表記する．G の直径 $D(G)$ は，すべての異なる頂点間の最短経路長の最大値で次のように定義される．

$$D(G) = \max_{u,v \in V(G)} \mu_G(u,v). \tag{1}$$

G のすべての異なる 2 つの頂点 u, v に対して，u から v への経路と v から u への経路が存在するとき，G は強連結であるという．強連結有向グラフ $G = (V,E)$ について，ある整数 k が $|V| \geq k+1$ を満たすと仮定する．このとき，任意の整数 $k' < k$ に対して，V から k' 個頂点を除去しても強連結性を失わないとき，G は k-強連結であるという．G が k-強連結性を維持する最大の k を点連結度といい，$\kappa(G)$ で表す．



(a) G (b) $L(G)$

図 1 有向グラフ G とそのラングラフ $L(G)$

2.2.2 ラングラフ

グラフ G のラングラフ $L(G)$ とは，G のすべての辺を頂点とし，G における辺の隣接関係に基づいて辺を張って構築されるグラフである．ラングラフは，元のグラフにおける辺の隣接関係を表現する．

定義 1 (ラングラフ). 有向グラフ G の各頂点がアルファベット Σ 上の文字列を識別子として持つとする．そして，識別子 $\boldsymbol{v} = v_1 v_2 \dots v_n$ を持つ頂点 v を $v = \boldsymbol{v} = v_1 v_2 \dots v_n$ と考える．このとき，有向グラフ G のラングラフ $L(G)$ は次の頂点集合と辺集合を持つグラフである．

$$V(L(G)) = \{uv \mid (u, v) \in E(G)\}, \tag{2}$$

$$E(L(G)) = \{(uv, vw) \mid (u, v), (v, w) \in E(G)\}. \tag{3}$$

上式は，ライン反復と呼ばれる．

図 1 にラングラフの例を示す.

反復ラングラフは，整数 $k \geq 1$ に対して次のように再帰的に定義される.

$$L^1(G) = L(G),\ L^2(G) = L\left(L^1(G)\right), \dots,\ L^{k+1}(G) = L\left(L^k(G)\right). \tag{4}$$

2.2.3 De Bruijn グラフと Kautz グラフ

一般に，高性能なネットワークとは，任意の 2 頂点間の経路長が短く，かつ耐障害性の高いものを指す．高性能なネットワークを設計するためには，頂点数 n，最大出次数 d のグラフ G について，直径 $D(G)$ の最小化と点連結度 $\kappa(G)$ の最大化を両立するグラフの構成手法が重要となる．しかしながら，この 2 つの目的関数を最適化するグラフは知られていない．

上述の問題に深く関連するものとして，次数直径問題 [18] がある．次数直径問題とは，最大出次数が d，直径が k となる有向グラフのうち，頂点数が最大となるものを求める問題である．このとき，頂点数の上界は，次の Moore バウンドと呼ばれる不等式によって与えられる．

定理 1 (有向グラフの Moore バウンド)**.** 最大出次数 d，直径 k の有向グラフについて，その頂点数 n は次式を満たす．

$$n \leq 1 + d + d^2 + \cdots + d^k = \frac{d^{k+1} - 1}{d - 1}. \tag{5}$$

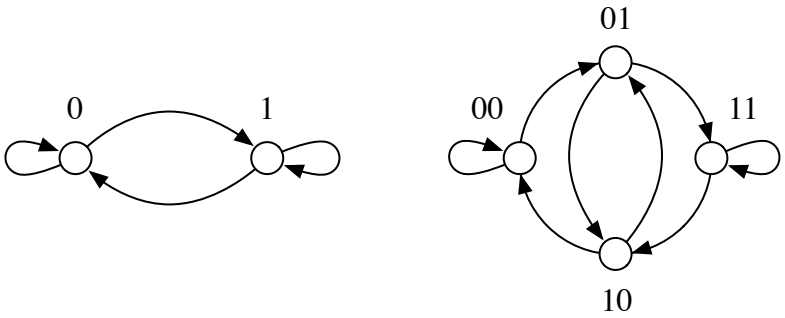
また，頂点数 n，最大出次数 d の有向グラフ G の直径 $D(G)$ の下界は，(5) から導くことができる [19]．

系 1 (有向グラフの直径の下界)**.** 頂点数 n，最大出次数 d の有向グラフ G について，その直径 $D(G)$ は次式を満たす．

$$D(G) \geq \lceil \log_d(n(d-1)+1) \rceil - 1. \tag{6}$$

任意の最大次数と直径について，頂点数が Moore バウンドの上界に一致する有向グラフは知られていない．Moore バウンドに非常に近い頂点数を持つ有向グラフが以下で紹介する de Bruijn グラフや Kautz グラフである．

De Bruijn グラフ　De Bruijn グラフは，次のように定義される有向グラフである．



(a) $G_B(2,1)$ (b) $G_B(2,2)$

図 2　De Bruijn グラフ $(d=2)$

定義 2 (De Bruijn グラフ). De Bruijn グラフ $G_B(d,k)$ は，アルファベット Σ_d 上の長さ k の文字列の集合 Σ_d^k を用いて，次のように定義される．

$$V(G_B(d,k)) = \{v_1v_2\dots v_k \mid v_1v_2\dots v_k \in \Sigma_d^k\}, \tag{7}$$

$$\begin{aligned} E(G_B(d,k)) = \{&(u_1u_2\dots u_k,\, v_1v_2\dots v_k) \mid \\ &u_2u_3\dots u_k = v_1v_2\dots v_{k-1}, \\ &u_1u_2\dots u_k,\, v_1v_2\dots v_k \in V(G_B(d,k))\}. \end{aligned} \tag{8}$$

図 2 に，$d=2$ の de Bruijn グラフの例を示す．

De Bruijn グラフはライングラフを用いて定義可能なことが知られている．具体的には，整数 $k \geq 1$ に対して次のように再帰的に定義される．

$$G_B(d,1) = K_d^+,\, G_B(d,2) = L(G_B(d,1)),\, \dots,\, G_B(d,k+1) = L(G_B(d,k). \tag{9}$$

$G_B(d,k)$ は d-正則かつ $(d-1)$-強連結であり，その直径は k である．また，$G_B(d,k)$ の頂点数は d^k であり，(5) の最大項に一致する．そして，直径 $D(G_B(d,k))$ は，(6) の下界に一致することが知られている．

Kautz グラフ　Kautz グラフは，次のように定義される有向グラフである．

定義 3 (Kautz グラフ). Kautz グラフ $G_K(d,k)$ は，アルファベット Σ_{d+1} 上の

長さ k の文字列の集合 Σ_{d+1}^k を用いて，次のように定義される．

$$\begin{aligned} V(G_K(d,k)) = \{&v_1v_2\dots v_k \mid v_1 \neq v_2,\, v_2 \neq v_3,\, \dots,\, v_{k-1} \neq v_k, \\ &v_1v_2\dots v_k \in \Sigma_{d+1}^k\}, \end{aligned} \tag{10}$$

$$\begin{aligned} E(G_K(d,k)) = \{&(u_1u_2\dots u_k,\, v_1v_2\dots v_k) \mid \\ &u_2u_3\dots u_k = v_1v_2\dots v_{k-1}, \\ &u_1u_2\dots u_k,\, v_1v_2\dots v_k \in V(G_K(d,k))\}, \end{aligned} \tag{11}$$

アルファベット Σ_{d+1} 上の長さ k の文字列 $\boldsymbol{s} = s_1s_2\dots s_k$ が $s_1 \neq s_2,\, s_2 \neq s_3,\, \dots,\, s_{k-1} \neq s_k$ を満たすとき，$\boldsymbol{s}$ を Kautz 文字列と呼ぶ．

図 3 に $d = 2$ の Kautz グラフの例を示す．

De Bruijn グラフと同様，Kautz グラフはライングラフを用いて定義可能である．具体的には，整数 $k \geq 1$ に対して次のように再帰的に定義される．

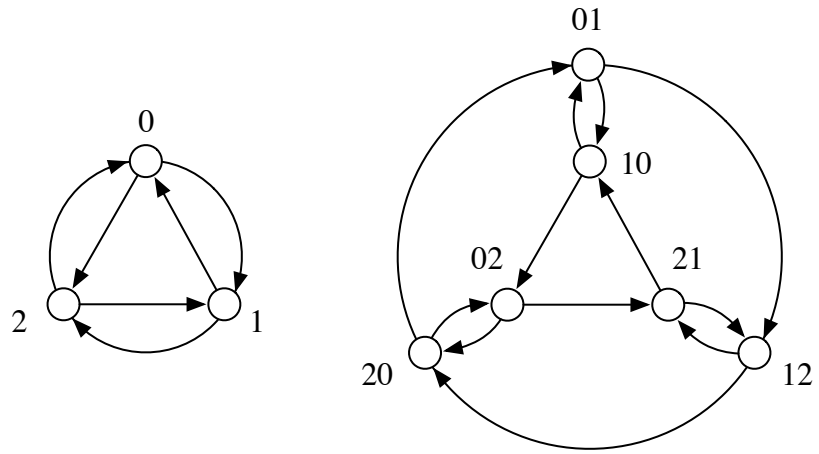
$$G_K(d,1) = K_{d+1},\, G_K(d,2) = L(G_K(d,1)),\, \dots,\, G_K(d,k+1) = L(G_K(d,k)). \tag{12}$$

$G_K(d,k)$ は d-正則かつ d-強連結であり，その直径は k である．また，$G_K(d,k)$ の頂点数は $d^k + d^{k-1}$ であり，de Bruijn グラフよりも (5) に近い値となる．直径 $D(G_K(d,k))$ は，(6) の下界に一致することが知られている．

2.3 オーバレイネットワーク

Peer-to-Peer (P2P) システムは，分散システムのひとつとして定義される．P2P システムを構成する計算機は，ノード，あるいは，ピアと呼ばれ，それらはシステムにおいて対等な役割を持つ．P2P システムは，非集中的であり，耐障害性やスケーラビリティに優れるといった性質を持つため，ユビキタスコンピューティングやクラウドコンピューティング，エッジコンピューティングといったあらゆる分野で活用されている [3]．

P2P システムにおけるノードは，IP ネットワークなどの既存のネットワーク (アンダーレイネットワーク) の上位で論理的なネットワーク (オーバレイネットワーク) を形成する．ここで，トポロジとは，ネットワークの構造を意味する．



(a) $G_K(2,1)$　　(b) $G_K(2,2)$

図 3　Kautz グラフ $(d = 2)$

P2P システムで形成されるオーバレイネットワークのうち，トポロジが一定の数学的アルゴリズムに基づいて決定されるものを構造化オーバレイネットワークと呼ぶ．

n 個のノードからなる構造化オーバレイネットワークを考える．このとき，ノードの集合を N と表記する．また，ノードは，ノードそのものとノードが持つ識別子の両方を指すこととする．ノードは，識別子空間 S から識別子が与えられている．識別子空間には，距離メトリック ρ が定義されている．距離メトリックは，(1) $u \neq v$ となるすべての $u, v \in S$ に対して $\rho(u, v)$ を満たし，(2) すべての $u \in S$ に対して $\rho(u, u) = 0$ を満たす [1]．不等式 $\rho(u, w) > \rho(v, w)$ は，ノード v がノード u よりもノード w に近いことを表す．例えば，Chord [1] では，各ノードは自身の IP アドレスのハッシュ値である k ビットの整数を識別子を持つ．Chord における識別子空間は $S = [0, 2^k)$ で，距離メトリックは $\rho(u, v) = (v - u) \bmod 2^k$ を用いる．

ノード v は，経路表 $E_v = \{e_1, e_2, \dots\}$ を保持する．経路表とは，隣接ノードの識別子とその IP アドレスの組である経路表エントリ $e_1, e_2, \dots$ の集合である．

[1] 距離の公理に含まれる対称律と三角不等式は必ずしも満たさなくてよい．

オーバレイネットワークのトポロジは，有向グラフとしてモデル化される．すなわち，$v \in E_u$ は $u \in E_v$ と独立する事象である．ここで，ノード v の入次数とは $|\{u \mid v \in E_u, u \in N\}|$ で，出次数とは $|E_v|$ を指す．Chord では，ノード v は，サクセッサとプレデセッサと呼ばれる隣接ノードを保持する．サクセッサとプレデセッサは，

$$\mathsf{succ}_N(v) = \arg\min_{w \in N} \rho(v, w), \tag{13}$$

$$\mathsf{pred}_N(v) = \arg\min_{u \in N} \rho(u, v), \tag{14}$$

で得られるノードである．また，v はフィンガーと呼ばれる以下の k 個の隣接ノードを経路表に保持する．

$$\mathsf{finger}_N^{(i)}(v) = \mathsf{succ}_N(2^i), \qquad i = 0, 1, \ldots, k-1. \tag{15}$$

分散ハッシュ表 (Distributed Hash Tables, DHT) は，構造化オーバレイネットワークのひとつである．DHT は，オブジェクトのキーを指定してその値を得る操作 $\mathsf{get}(\mathsf{key})$ と，オブジェクトをある特定のノードに格納する操作 $\mathsf{put}(\mathsf{key}, \mathsf{value})$ を提供する．多くの DHT では，コンシステントハッシュ [20] と呼ばれる手法に基づいてキーが対応するノードを一意に決定する．コンシステントハッシュでは，キー r のハッシュ値を計算してキー空間 R をノードの識別子空間 S に写像し，

$$v = \arg\min_{u \in N} \rho(u, r), \tag{16}$$

によって決定されたノード v を r の担当ノードとする．すなわち v は，キー集合

$$\{r \mid \rho(v, r) < \rho(u, r), \forall u \in N, u \neq v, r \in R\}, \tag{17}$$

の担当ノードとなる．

構造化オーバレイネットワークにおけるルーティングは，距離メトリック ρ を用いて貪欲的に行われる．宛先の識別子を d とし，ルーティングによって得られる長さ ℓ の経路を $w_1 \to w_2 \to \cdots \to w_{\ell+1}$ とするとき，経路上の各ノード w_i $(1 \leq i \leq \ell)$ は自身の経路表 E_{w_i} から

$$w_{i+1} = \arg\min_{w' \in E_{w_i}} \rho(w', d), \quad (1 \leq i \leq \ell), \tag{18}$$

を選んで次ホップとする．Chord [1] や Kademlia [2] をはじめとする従来の構造化オーバレイネットワークでは，ノードの出次数は $O(\log n)$，探索の平均経路長は $O(\log n)$ となる．

3. 関連研究

本節では，正則グラフに基づく構造化オーバレイネットワークのうち代表的なものを取り上げる．また，従来の構造化オーバレイネットワークにおいてノード間のネットワーク近接性を考慮する手法を紹介する．そして，従来の近接性の考慮手法が正則グラフに基づくオーバレイネットワークに対して容易に適用することができないことを示す．

3.1 正則グラフに基づく構造化オーバレイネットワーク

3.1.1 Koorde

De Bruijn グラフに基づく構造化オーバレイネットワークとして，Koorde [4] が提案されている．Koorde は，Chord [1] を 2-正則 de Bruijn グラフのトポロジに従うように変更したものである．Koorde は，Chord と同様に k ビットの整数で構成される識別子空間 S を用いる．

ノード v は，次の 2 つの隣接ノードを経路表に保持する．

- $\mathsf{succ}_N(v)$
- $\mathsf{pred}_N(2v)$

ここで，$\mathsf{pred}_N(2v)$ は v の識別子を左に 1 ビットシフトした識別子を担当するノードとなる．Koorde は，de Bruijn グラフのルーティングを仮想的に実現する．De Bruijn グラフのノードに対応するノードがオーバレイネットワーク上に存在しない場合，そのプレデセッサをたどる．このルーティング方式により，探索に必要なメッセージ数は高確率で $O(\log_2 n)$ となる．

ノード v が $d = O(\log n)$ 個の隣接ノードを保持する場合，Koorde ネットワークの直径は $O(\log n - \log\log n)$ となる．このとき，v は，次の隣接ノードを経路表に保持する．

- $\mathsf{succ}_N(v)$

- $\mathsf{pred}_N(dv)$, $\mathsf{succ}_N(\mathsf{pred}_N(dv))$, ..., $\mathsf{succ}_N^{d-1}(\mathsf{pred}_N(dv))$

Koorde では，de Bruijn グラフの構造を仮想的に実現するために，各ノードの隣接ノードが厳密に決められている．決められたノード以外を隣接ノードとすると，de Bruijn グラフの構造が破壊されるため，以降で説明する近接性の考慮手法を Koorde に適用することは容易でない．また，de Bruijn グラフのノードに対応するノードがオーバレイネットワーク上に存在しない場合，Koorde では，仮想的なホップが引き起こされ，結果として経路長が大きくなるという問題点がある．提案手法は，分散ライングラフをベースとするアプローチであるため，de Bruijn グラフや Kautz グラフの頂点に対応するノードがオーバレイネットワーク上に存在する．したがって，Koorde のように仮想的なホップが発生することはない．

3.1.2 FissionE

Kautz グラフに基づく構造化オーバレイネットワークとして代表的なものに FissionE [6] がある．FissionE は，2-正則 Kautz グラフをトポロジとする構造化オーバレイネットワークである．FissionE のノードは，直交座標系の特定のゾーンに対応する．ゾーンは，アルファベット $\Sigma_3 = \{0, 1, 2\}$ 上の Kautz 文字列を識別子として持つ．それぞれのゾーンは，Kautz グラフの構造に従って隣接している．CAN [21] と同様に，座標空間はノードの参加や離脱によって動的に分割され，ゾーンの識別子もそれに伴って変化する．

FissionE のノード，すなわち，ゾーン $\boldsymbol{v} = v_1 v_2 \dots v_k$ は，次の 2 つの隣接ノードを持つ．

出隣接ノード $\alpha_1, \alpha_2, \dots, \alpha_m \in \Sigma_3$, $v_k \neq \alpha_1$, $\alpha_1 \neq \alpha_2$, ..., $\alpha_{m-1} \neq \alpha_m$ とする．ゾーン $\boldsymbol{v}$ に対応するノードは，識別子が $\boldsymbol{w} = v_2 v_3 \dots v_k \alpha_1 \dots \alpha_m$ $(0 \leq m \leq 2)$ となるゾーン $\boldsymbol{w}$ を出隣接ノードとして経路表に保持する．

入隣接ノード $\beta \in \Sigma_3$, $\beta \neq v_1$ とする．ゾーン $\boldsymbol{v}$ に対応するノードは，識別子が $\boldsymbol{u} = \beta v_1 v_2 \dots u_\ell$ $(k-2 \leq \ell \leq k)$ となるゾーン $\boldsymbol{u}$ を入隣接ノードとして経路表に保持する．

FissionE のルーティング方式は，Kautz グラフのルーティング方式に等しい．また，FissionE のネットワークの入次数と出次数の和の平均は 4 となり，これを上回る入次数や出次数のネットワークを構築することはできない．FissionE のネットワークの直径は $O(\log_2 n)$ となることが知られている．

FissionE のゾーンの隣接関係は，Kautz グラフの構造に基づいて厳密に決められている．これに対して，提案手法は，分散ライングラフに基づいて Kautz グラフの性質を持つトポロジのオーバレイネットワークを構築するため，ノードの参加や離脱の際に引き起こされるノード融合やノード分割において，隣接ノードを柔軟に選択することができる．

3.2 近接性を考慮した構造化オーバレイネットワーク

構造化オーバレイネットワークにおいてノード間のネットワーク近接性を考慮する手法は，大きく以下の 3 つに分けられる．

- 近接隣接ノード選択 (Proximity Neighbor Selection, PNS) [11, 12, 13, 14]
- 近接経路選択 (Proximity Route Selection, PRS) [12, 15]
- 近接識別子選択 (Proximity Identifier Selection, PIS) [16]

PNS では，それぞれのノードが候補の中から近接性に基づいて隣接ノードを選択する．PRS では，探索の際にそれぞれのノードが探索遅延を最小化するような次ホップを経路表から選択する．PIS では，ノードの識別子が下位ネットワークの近接性に基づいて決定される．PIS により，下位ネットワークで近接するノードがオーバレイネットワークにおいて近い識別子を与えられ，オーバレイネットワークで近接するようになる．

これらの手法は，隣接ノードや探索経路を局所的かつ適応的に選択して下位ネットワークの近接性を考慮する手法と捉えることができる．下位ネットワークにおける近接性は，各ノードで計測可能な近接性メトリック [22] と相関がある．近接性メトリックは，ノード間の通信遅延や通信帯域を計測することによって計算される値である．

オーバレイネットワーク上のノード集合を N とする．あるノード $u \in N$ において，ノード $v \in N$ に関して計測される遅延メトリックを $\tau_u(v)$ とする．ここで，オーバレイネットワークの遅延直径 Δ は以下のように定義される．

$$\Delta = \max_{u,v \in N} \tau_u(v). \tag{19}$$

ここで，$\tau_u(v)$ は，下位ネットワークで計測される値であるため，オーバレイネットワークにおける u と v の隣接関係とは無関係に計測される．したがって，Δ はオーバレイネットワークにおける隣接ノードへの遅延の上界を表す．また，ノード s からノード t への ℓ-ホップ経路 $W : s = w_1 \to w_2 \to\to \cdots \to w_{\ell+1} = t$ を考えたとき，オーバレイネットワークにおける探索遅延は

$$L(W) = \sum_{i=1}^{\ell} \tau_{w_i}(w_{i+1}), \tag{20}$$

と定義され，$L(W)$ の上界は $\ell\Delta$ で与えられる．

先に述べた近接性に基づく選択手法 (PNS，PRS，PIS) では，隣接ノードや探索経路の次ホップを遅延メトリックに基づいて最適化する．これにより，探索経路の各ホップにおいて高確率で $\tau_{w_i}(w_{i+1}) \ll \Delta$ となり，結果として探索遅延 $L(W)$ が低減される．

3.2.1 探索時無作為抽出

PNS に基づく近接性の考慮手法の代表的なものに LPRS-Chord がある．LPRS-Chord は，探索時無作為抽出 (Lookup-Parasitic Random Sampling, LPRS) と呼ばれる手法に基づいて隣接ノードの遅延を最適化する．

LPRS-Chord は，オーバレイネットワークにおける探索方式の 1 つである再帰探索を仮定している．再帰探索とは，経路 W 上の各ノード w_i $(1 \leq i \leq \ell)$ が経路表から次ホップ w_{i+1} を選択し，探索メッセージを w_{i+1} に転送する方式である．P 上のノード w_i は，自身の IP アドレスを探索メッセージに埋め込む．探索メッセージが目的のノード t に到達した際，t は自身の IP アドレスを探索メッセージに埋め込まれた IP アドレスを用いて $w_1, w_2, \ldots, w_\ell$ に対して通知する．

そして，$w_1, w_2, \ldots, w_\ell$ は，それぞれ近接性メトリック $\tau_{w_i}(t)$ を計測し，それに基づいてそれぞれのノードの経路表を更新する．

正則グラフに基づくオーバレイネットワークでは，隣接ノードが正則グラフのトポロジによって厳密に決定されるため，目的のノード t は経路上のノード $w_1, w_2, \ldots, w_\ell$ の隣接ノードになり得ない．したがって，正則グラフに基づくオーバレイネットワークに対して LPRS を直接適用することは困難である．

3.2.2 適応的近接経路選択

適応的近接経路選択 (Adaptive Proximity Route Selection, APRS) [15] は，PRS に基づいて次ホップを決定する．APRS は，再起探索の Chord を仮定する．APRS は，各隣接ノードからいくつかの識別子までの探索遅延を見積もり，探索の際に探索遅延を最小化するような次ホップを選択する．

いま，ノード v がキー r への探索クエリを受信したことを考える．このとき，次ホップ候補の集合 $C_v(r)$ は，フィンガーの経路表 $E_v = \{e_0, e_2, \ldots, e_{k-1}\} = \left\{\mathsf{finger}_N^{(0)}, \mathsf{finger}_N^{(1)}, \ldots, \mathsf{finger}_N^{(k-1)}\right\}$ から次式のように決定される．

$$C_v(r) = \{e_i \mid e_i \in (v, r],\, e_i \in E_v\}. \tag{21}$$

$C_v(r)$ の中で探索遅延を最小化する次ホップを決定するために，APRS では，ノード v のフィンガーの経路表エントリ e_i $(0 \leq i \leq k-1)$ に次の情報を付加する．

$$c_v^{(i)}(e_i),\, c_v^{(i+1)}(e_i),\, \ldots,\, c_v^{(k)}(e_i). \tag{22}$$

$c_v^{(i)}(e_i), c_v^{(i+1)}(e_i), \ldots, c_v^{(k)}(e_i)$ は定期的に更新されるコスト値である．コスト値の更新は，次のように行われる．

$$c_v^{(i)}(e_i) := (1-\alpha) \times c_v^{(i)}(e_i) + \alpha \times \left(\tau_v(e_i) + \min_{x \in \text{neighbors of } e_i} c_{e_i}^{(i)}(x)\right). \tag{23}$$

ここで，α は $0 \leq \alpha \leq 1$ を満たす実数である．ノード v が受信したキー r への探索クエリは，$r \in [v + 2^j, v + 2^{j+1})$ $(0 \leq j \leq k-1)$ を満たす j に基づいて，

$$w := \arg\min_{c \in C_v(r)} c_v^{(j)}(c), \tag{24}$$

で得られるノード w に転送される．

De Bruijn グラフや Kautz グラフに基づく構造化オーバレイネットワークでは，最短経路を導くルーティング方式が明らかである．APRS のように，最短経路以外の次ホップを選択すると，これらの構造化オーバレイネットワークにおいては，経路長がその分だけ増加する．APRS を用いて探索遅延を低減するためには，ホップ数とホップごとの通信遅延のトレードオフを適切に決定する必要がある．また，APRS では，ネットワーク直径が増加するという問題がある．これに対して，提案手法は，正則グラフのネットワーク直径の維持と探索遅延の低減を両立するものである．

4. 分散ラングラフ

分散ラングラフ (Distributed Line Graph, DL グラフ) [17] は，任意の正則グラフに基づく構造化オーバレイネットワークの構築手法である．DL グラフは，分散ライン反復 (Distributed Line Iteration, DL 反復) によって得られる．構造化オーバレイネットワークのトポロジは，DL 反復によって生成された複数の頂点を 1 つの頂点に融合 (頂点融合)，あるいは融合されている頂点を 2 つの頂点に分割 (頂点分割) することによって得られる．

提案手法は，DL グラフを近接性を考慮するように拡張したものである．本節では，DL 反復と頂点融合，頂点分割，そして，オーバレイネットワークの構築手法について述べる．

4.1 分散ラングラフと分散ライン反復

DL グラフは，ある正則グラフに対して DL 反復を行うことによって得られる．初期グラフを 2-正則完全有向グラフ K_3 とする DL グラフを図 4 に示す．DL グラフと DL 反復は，次のように定義される．

定義 4 (分散ラングラフと分散ライン反復)．頂点数 p の d-正則有向グラフ G_0 を初期グラフとする．有向グラフ G_{i-1} の頂点 v に関する分散ラングラフ $G_i = DL(G_{i-1}, v)$ $(i \geq 1)$ は，すべての頂点 $u \in N_{G_{i-1}}(v)$ に対して $|v| \leq |u|$ を満たすような v に対して次のように求められる．

$$V(G_i) = V(G_{i-1}) \setminus \{v\} \cup \left\{ u \circ v \mid u \in N^-_{G_{i-1}}(v) \right\}, \tag{25}$$

$$\begin{aligned} E(G_i) = E(G_{i-1}) &\setminus \left\{ (u, v) \mid u \in N^-_{G_{i-1}}(v) \right\} \setminus \left\{ (v, w) \mid w \in N^+_{G_{i-1}}(v) \right\} \\ &\cup \left\{ (u, u \circ v) \mid u \in N^-_{G_{i-1}}(v) \right\} \\ &\cup \left\{ (u \circ v, w) \mid u \in N^-_{G_{i-1}}(v),\, w \in N^+_{G_{i-1}}(v) \right\}. \end{aligned} \tag{26}$$

上式を分散ライン反復と呼ぶ．分散ラングラフに対して分散ライン反復を行なって得られるグラフもまた分散ラングラフである．ここで，演算子 $\circ$ は，

$m \geq n$ を満たす頂点 $u = u_1u_2 \ldots u_m$, $v = v_1v_2 \ldots v_n$ に対して，$u \circ v = \boldsymbol{u} \circ \boldsymbol{v} = u_{m-n+1}v_1v_2 \ldots v_n$ と定義される．

DL グラフ $G' = DL(G, v)$ では，G の頂点 v を元に d 個の新たな頂点が G' に生成され，v が G' から削除される．同一の DL 反復によって生成された頂点を兄弟頂点と呼ぶ．それぞれの兄弟頂点で入隣接頂点は異なるが，出隣接頂点は等しく，出次数は d となる．また，入次数は 1 以上 d^2 以下であり，その平均は d となることが示されている．DL グラフ G の直径 $D(G)$ は，d-正則な初期グラフ G_0 の頂点数を n_0，DL グラフの頂点数を n として，次式の条件を満たす．

$$D(G) < 2(\log_d n - \log_d n_0 + D(G_0) + 1). \tag{27}$$

DL グラフの頂点数は，DL 反復ごとに $d-1$ ずつ増加する．d が $d > 2$ の場合，DL 反復ごとに頂点数が 2 以上増加するため，DL グラフをオーバレイネットワークのトポロジとして直接利用することはできない．そこで，兄弟頂点を 2 つの頂点に融合したり，融合されている頂点を 2 つの頂点に分割することによってこの問題を解決する．

4.2 頂点融合と頂点分割

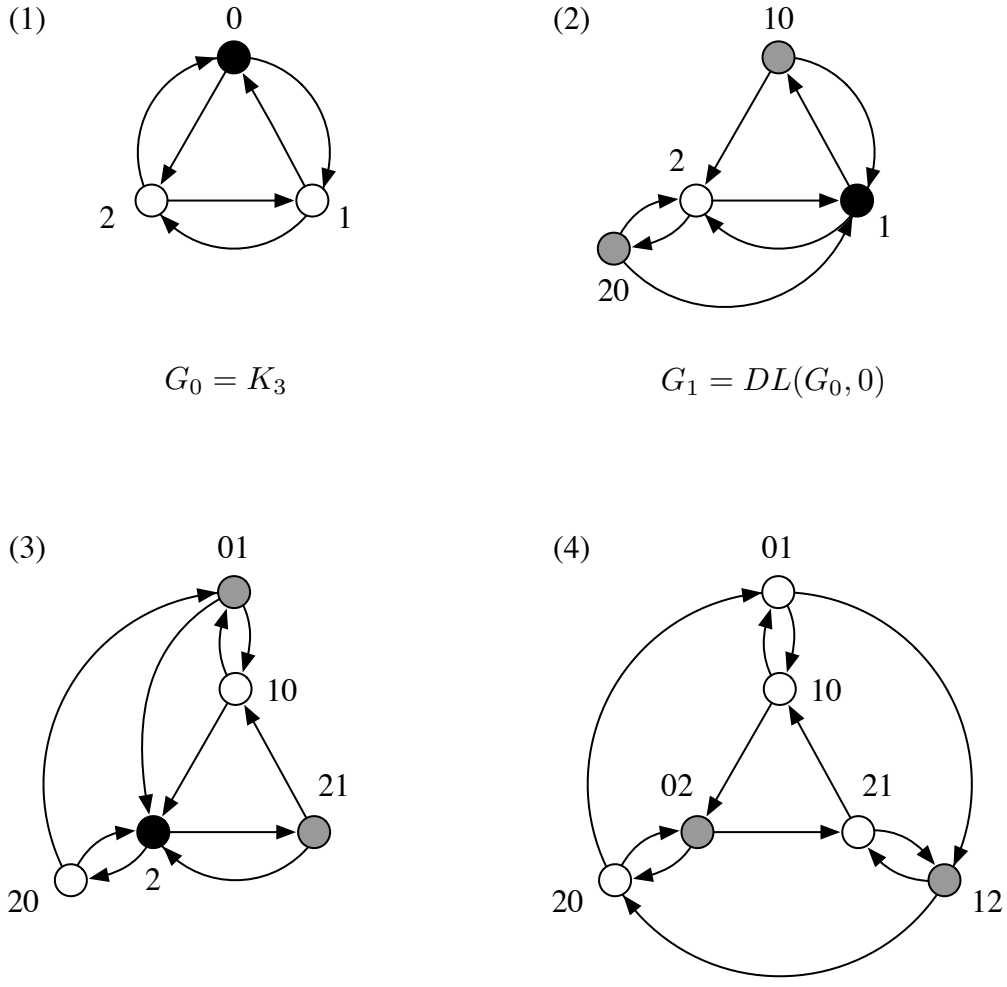
頂点融合とは，グラフ G のいくつかの頂点を 1 つの頂点に融合して，新たなグラフ G' を得ることをいう．G の q 個の頂点を $v_1, v_2 \ldots, v_q$ とし，これらを融合した頂点を v とするとき，頂点融合によって得られるグラフ G' は，頂点 v に関して次の隣接関係を持つ．

$$N^-_{G'}(v) = \bigcup_{i=1}^{q} N^-_G(v_i), \qquad N^+_{G'}(v) = \bigcup_{i=1}^{q} N^+_G(v_i). \tag{28}$$

ここで，頂点 v に融合されている頂点数を $\|v\|$ で表す．

これに対して，頂点分割は，あるグラフにおいて 1 つの頂点に融合されている頂点をいくつかの頂点に分割して，新たなグラフを得ることをいう．

DL 反復により生成される兄弟頂点に対する頂点融合と頂点分割の手法はいくつか提案されている [17, 23]．Zhang ら [17] は，兄弟頂点の識別子の最初の 1 文



$G_0 = K_3$　　$G_1 = DL(G_0, 0)$

$G_2 = DL(G_1, 1)$　　$G_3 = DL(G_2, 2)$

図 4　初期グラフ K_3 の DL グラフ

字のアルファベット順に基づいて頂点融合と頂点分割を行う DL^{+} グラフを提案している．これに対し，眞田ら [23] は，[17] における頂点融合と頂点分割を柔軟にした $DL^{\sharp}$ グラフを提案している．

4.3 オーバレイネットワークの構築手法

DL グラフに基づく構造化オーバレイネットワークの構成手法は，これまでにいくつか提案されている．DLG-de Bruijn [17] は，ループ付き完全有向グラフを初期グラフとする DL グラフにより，de Bruijn グラフに近似するトポロジを持つ構造化オーバレイネットワークを実現した．また，DLG-Kautz (SKY) [17, 24] は，完全有向グラフを初期グラフとする DL グラフにより，Kautz グラフに近似するトポロジを持つ構造化オーバレイネットワークを実現した．他にも，バタフライグラフに近似するトポロジを持つ DLG-Butterfly [17] やハイパーツリーグラフに近似するトポロジを持つ DLG-Hypertree [17, 25] などが提案されている．

DL グラフに基づく構造化オーバレイネットワークにおけるトポロジのメンテナンス手法は [24] で提案されている．しかしながら，紹介されているアルゴリズムは，グラフ理論の表記を多用した記述となっており，分散アルゴリズムとして実装することが困難である．本節では，d-正則完全有向グラフを初期グラフとする DL グラフをトポロジとするオーバレイネットワークについて，そのメンテナンス手法を遠隔手続き呼出し (Remote Procedure Call, RPC) の形式で説明する．

4.3.1 ノードが保持する局所的隣接情報

オーバレイネットワーク上の物理ノード (Physical Node) の集合を N とする．オーバレイネットワーク上のそれぞれの物理ノードには，1 つ以上の仮想ノード (Virtual Node) が融合されている．ここで，物理ノード (あるいは単にノード) は，特定の IP アドレスにより IP ネットワークへ接続している計算機とし，仮想ノードは，DL グラフの頂点に対応するプロセスとする．

仮想ノード $v \in V(G)$ は，表 1 の情報を保持する．v の識別子 $v.identifier$ は，アルファベット Σ_{d+1} 上の Kautz 文字列で，その長さは $|v|$ と表記される．v の

表 1　仮想ノード v が保持する情報とその表記

表記	説明
$v.identifier$	仮想ノードの識別子
$\|v\|$	識別子長
$v.successors$	出隣接ノード (経路表)
$v.predecessors$	入隣接ノード
$v.neighbors$	隣接ノード

経路表 $v.successors$ は，グラフにおける v の出隣接頂点集合に対応する．同様に，$v.predecessors$ は，グラフにおける v の入隣接頂点集合に対応する．また，$v.neighbors$ は，$v.successors \cup v.predecessors$ とする．ここで，$v.successors$, $v.predecessors$, $v.neighbors$ は，仮想ノードの識別子と物理ノードの IP アドレスの組を要素とする集合である．

物理ノード $p \in N$ は，表 2 の情報を保持する．$p.address$ は，p に割り当てられている IP アドレスである．$p.substances$ は，p に融合されている仮想ノードの集合で，その数を $\|p\|$ で表す．$p.identifiers$ は，$p.substances$ のすべての仮想ノードの識別子の集合である．また，$p.substances$ のすべての仮想ノードは同じ長さの識別子を持つと仮定し，その長さを $|p|$ で表す．$p.successors$ は，$p.substances$ すべての出隣接ノードへの経路表の和集合，すなわち $\bigcup_{v \in p.substances} p.successors$ である．同様に，$p.predecessors$, $p.neighbors$ は，それぞれ $\bigcup_{v \in p.substances} v.predecessors$, $\bigcup_{v \in p.substances} p.neighbors$ である．

4.3.2 初期ネットワークの作成

初期ネットワークを作成することは，DL グラフの初期グラフをトポロジとするオーバレイネットワークを構築することに等しい．初期グラフを d-正則完全有向グラフ K_{d+1} としたとき，初期ネットワークを構築する手続きは，アルゴリズム 1 のようになる．

表 2　物理ノード p が保持する情報とその表記

表記	説明
$p.address$	ノードの IP アドレス
$p.substances$	ノードに融合されている仮想ノードの集合
$\|\|p\|\|$	$\|p.substances\|$
$p.identifiers$	$\{v.identifier \mid v \in p.substances\}$
$\|p\|$	仮想ノードの識別子長 [2]
$p.successors$	$\bigcup_{v \in p.substances} v.successors$
$p.predecessors$	$\bigcup_{v \in p.substances} v.predecessors$
$p.neighbors$	$\bigcup_{v \in p.substances} v.neighbors$

アルゴリズム 1. 初期ネットワークの作成

Let $N \leftarrow \{p_1, p_2, \ldots, p_{d+1}\}$ be a physical node set of the initial network
for $i \leftarrow 1$ **to** $d+1$ **:**
    Create a new virtual node v
    $\boldsymbol{i} \leftarrow i$ as a Kautz string
    $v.identifier \leftarrow \boldsymbol{i}$
    for $j \leftarrow 1$ **to** $d+1$ **:**
        if $i = j$ **:**
            continue
        $\boldsymbol{j} \leftarrow j$ as a Kautz string
        $v.successors \leftarrow v.successors \cup \{(\boldsymbol{j}, p_j.address)\}$
        $v.predecessors \leftarrow v.predecessors \cup \{(\boldsymbol{j}, p_j.address)\}$
    $p_i.substances \leftarrow \{v\}$

[2] ここで，すべての異なる 2 つ仮想ノード $v, v' \in p.substances$ に対して $|v| = |v'|$ が成立することに注意.

4.3.3 ネットワーク上の探索

オーバレイネットワーク上からある識別子を持つノードを探索することを考える．探索の際には，2 つの識別子 $\boldsymbol{u} = u_1u_2\ldots u_m$, $\boldsymbol{v} = v_1v_2\ldots v_n$ の論理的な距離を表すメトリック

$$\begin{aligned}\rho(\boldsymbol{u},\boldsymbol{v}) = \max\{i \mid 1 \leq i \leq \min(|\boldsymbol{u}|,|\boldsymbol{v}|),\\ \forall j(0 \leq j \leq i-1)(u_{m-i+j} = v_j)\}.\end{aligned} \tag{29}$$

を用いる．$\rho(\boldsymbol{u},\boldsymbol{v})$ は，$\boldsymbol{u}$ における $\boldsymbol{v}$ に関するマッチングインデックス [17] と呼ばれ，$\boldsymbol{u} \restriction \ell = \boldsymbol{v} \restriction \ell$ を満たす ℓ の最大値を表している．

DL 反復により，ある識別子 $\boldsymbol{v}$ に対して $\rho(u,\boldsymbol{v})$ が最大となるノード u が複数存在する場合がある．このとき，$\boldsymbol{v}$ に対応する u を一意に決定するために，次の値を用いる．

$$\varphi(\boldsymbol{u},\boldsymbol{v}) = \left|\theta(u_{m-\rho(\boldsymbol{u},\boldsymbol{v})}, u_{m-\rho(\boldsymbol{u},\boldsymbol{v})+1}) - \theta(v_{\rho(\boldsymbol{u},\boldsymbol{v})+1}, v_{\rho(\boldsymbol{u},\boldsymbol{v})})\right|. \tag{30}$$

ここで，θ は 2 つの整数 a, b $(\in \Sigma_{d+1})$ に対して定義される値で次のように定義される．

$$\theta(x,y) = \begin{cases} x & \text{if } x < y, \\ x-1 & \text{otherwise.} \end{cases} \tag{31}$$

オーバレイネットワーク上の探索アルゴリズムをアルゴリズム 2 に示す．ノード p が宛先 $\boldsymbol{t} = t_1t_2\ldots$ を探索することを考える．まず，p は $p.substances$ の中から $\boldsymbol{t}$ に関するマッチングインデックスが最大となる仮想ノード s を選択する．そして，s は $s.successors$ の中からマッチングインデックスを最大化するように貪欲的に探索クエリの転送先を決める．探索メッセージがマッチングインデックスを最大化するノードに到達したとき，探索は終了する．

4.3.4 ノードの参加

ノードの参加は，オーバレイネットワークのトポロジに対応する物理ノードグラフの頂点数を 1 だけ増加させることに等しい．これは，DL 反復と頂点融合，

アルゴリズム 2. オーバレイネットワーク上の探索

p. **lookup($\boldsymbol{t}$):**

```
s ← arg max_{v∈p.substances} ρ(v.identifier, t)
for w ∈ s.successors :
    if ρ(w.identifier, t) > ρ(s.identifier, t) :
        return w.lookup(t)
for α ∈ Σ_{d+1} :
    s' ← α(s.identifier ↑ |s| − 1)
    if (ρ(s, t) = ρ(s', t)) ∧ (φ(s', t) = 0) :
        return p.lookup(s')
return p
```

頂点分割によって行われる．ノードの参加アルゴリズムは，アルゴリズム 3, 4 に示すとおりである．

ノード p がネットワークに参加するとき，p はネットワーク上に既に存在しているノード p' (ブートストラッピングノード) を介して，DL 反復が適用可能なノード r (レスポンシブルノード) を探索する．r の探索は，出力が Kautz 文字列となるハッシュ関数に基づいて決定された宛先を探索することによって行われる．DL 反復を行うため，r はすべての隣接するノード $s \in r.neighbors$ に対して $|r| \leq |s|$ を満たす必要がある．また，出次数を d に保つために，物理ノードに融合されている仮想ノードの数も考慮しなければならない．したがって，r が満たすべき条件 [24] は，次のようになる．

$$(|r| < |s|) \vee (|r| = |s|) \wedge (\|r\| > \|s\|), \quad \text{for all } s \in r.neighbors. \tag{32}$$

この条件を満たす r は，アルゴリズム 3 にしたがって探索される．

レスポンシブルノード r の探索後，DL 反復とノード融合・ノード分割を行なって，ネットワークトポロジの拡張を行う．$\|r\| > 1$ の場合は r に融合されている仮想ノードを分割し，参加ノード p とレスポンシブルノード r で共有する．$\|r\| = 1$ の場合は，r に融合されている 1 つの仮想ノードに対して DL 反復を行

い，兄弟頂点に対応する d 個の仮想ノードを r 上に生成する．r は，生成した仮想ノードを参加ノード p とレスポンシブルノード r に融合する．これらは，アルゴリズム 4 にしたがって実行される．

アルゴリズム 3. ノード参加

```
p. join(p′):
    r ← p′.find_responsible_node(p)
    if |r| > 1 :
        r.split_and_allocate(p)
    else:
        r.merge_and_allocate(p)

p. find_responsible_node(p′):
    t ← Hash value of p′.address as a Kautz string
    r ← p′.lookup(t)
    while True :
        s ← r.find_shortest_node()
        r ← s.find_abundant_node()
        if r = s :
            break
    return r

p. find_shortest_node():
    for q ∈ p.neighbors :
        if |p| > |q| :
            return q.find_shortest_node()
    return p

p. find_abundant_node():
    for q ∈ p.neighbors :
        if |p| = |q| ∧ ‖p‖ < ‖q‖ :
            return q
    return p
```

アルゴリズム 4. DL 反復とノードの融合・分割

p. **merge_and_allocate(p'):**

p.generate_virtual_nodes()

p.split_and_allocate(p')

p. **split_and_allocate(p'):**

$S \leftarrow$ Sort $v \in p.substances$ in the alphabetical order of the first symbol of $v.identifier$

$p.substances \leftarrow$ First $\lceil d/2 \rceil$ nodes of S

$p'.substances \leftarrow$ Remaining $\lfloor d/2 \rfloor$ nodes of S

p. **generate_virtual_nodes():**

Let v be a virtual node in $p.substamces$

$p.substances \leftarrow \{\varepsilon\}$

for $w \in v.successors$ **:**

  Delete routing information to v from $w.predecessors$

for $u \in v.predecessors$ **:**

  Delete routing information to v from $u.successors$

for $u \in v.predecessors$ **:**

  Create a new virtual node v

  $s.identifier \leftarrow u.identifier \circ v.identifier$

  if $s.identifier \notin p.identifiers$ **:**

    $p.substances \leftarrow p.substances \cup \{s\}$

    $s.successors \leftarrow v.successors$

    for $w \in v.successors$ **:**

      Add $(s.identifier, p.address)$ on $w.predecessors$

  Add $(u.identifier, u.address)$ on $s.predecessors$

  Add $(s.identifier, p.address)$ on $u.successors$

5. 近接性を考慮した分散ライングラフ

本節では，近接性を考慮した分散ライングラフ (Proximity-Aware Distributed Line Graph, PDL グラフ) を提案する．PDL グラフは，探索遅延を低減するために，DL グラフをノード間の近接性を考慮するように拡張したものである．したがって，DL グラフと同様に，PDL グラフは任意の正則グラフを初期グラフに対して適用することができる．そのため，PDL グラフに基づく構造化オーバレイネットワークは，初期グラフの性質をオーバレイネットワーク上で再現することでき，それに加えて探索遅延の低減が可能である．

PDL グラフは，再帰探索を仮定する手法である．PDL グラフでは，ノードの参加や離脱でオーバレイネットワークのトポロジを拡大，あるいは縮小する際に，ノード間の近接性を考慮する．より厳密には，DL グラフにおける頂点融合と頂点分割を近接性メトリックを用いて行う．図 5 は，PDL グラフの概略図を示している．PDL グラフは，可能なトポロジ候補のグラフから近接性に基づいてオーバレイネットワークに対応するグラフを選択する近接グラフ選択 (Proximity Graph Selection, PGS) として考えることができる．

単純のため，ここでは，ノード u にて観測されるノード v に関する近接性メトリック $\tau_u(v)$ が，次の性質を満たすことを仮定する．

時不変性 $\tau_u(v)$ が時間に伴って変化しないこと

対称性 $\tau_u(v) = \tau_v(u)$ を満たすこと

また，2 つのノード v と v' について，$\tau_u(v) < \tau_u(v')$ は v が v' よりも近接性の観点で近いことを意味する．近接性メトリックは，通信遅延や通信帯域を計測し，それに基づいて決定される値である．例えば，UNIX の ping コマンドを用いてラウンドトリップタイムを 10 回計測し，その平均値を近接性メトリックとすることなどが考えられる．

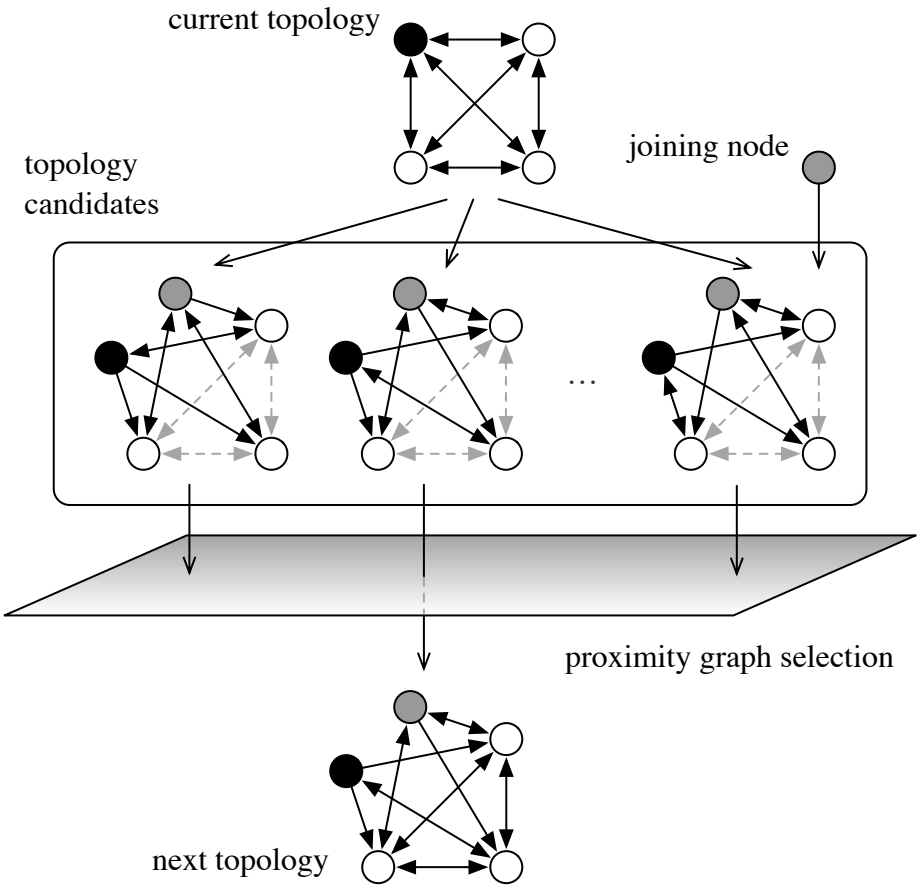


図 5　PDL グラフの概略図

5.1 近接性を考慮したノード融合とノード分割

5.1.1 近接性を考慮したノード融合

初期グラフを d-正則有向グラフとする分散ラングラフ G を考える．ある DL 反復により生成された d 個の兄弟頂点に対応する仮想ノードの集合を $S := \{s_1, s_2, \ldots, s_d\}$ とする．これら仮想ノードは，2 つの物理ノード p と q に融合される．ここでは，簡単のために $P := p.substances$，$Q := q.substances$ と表記することとする．

仮想ノード $v \in S$ を考える．v に関して p で計測される平均近接性メトリックとは，すべての $u \in v.predecessors$，すなわちすべての $u \in N_G^-(v)$ に対して近接性メトリックを計測して得られる値で，具体的には次のように定義される．

$$T_p(v) = \frac{\sum_{u \in N_G^-(v)} \tau_p(u)}{\delta_G^-(v)}. \tag{33}$$

同様にして，v に関して q で計測される平均近接性メトリックは，次のように定義される．

$$T_q(v) = \frac{\sum_{u \in N_G^-(v)} \tau_q(u)}{\delta_G^-(v)}. \tag{34}$$

P と Q は近接性メトリックに基づいて決定される．より厳密には，P と Q は次の目的関数を最小化するように決められる．

$$c(P, Q) = \sum_{v \in P} T_p(v) + \sum_{v \in Q} T_q(v). \tag{35}$$

ここで，(35) は次のように変形することができる．

$$c(P, Q) = \sum_{v \in P} (T_p(v) - T_q(v)) + \sum_{v \in P \cup Q} T_q(v). \tag{36}$$

(36) の第 2 項は定数である．したがって，(35) を最小化することは，次の関数を最小化することに等しい．

$$c'(P) = \sum_{v \in P} (T_p(v) - T_q(v)). \tag{37}$$

(37) を最小化する P と Q は，すべての仮想ノード $v \in S$ を次の値

$$h(v) = T_p(v) - T_q(v), \tag{38}$$

の昇順に並び替え，最初の $\lceil d/2 \rceil$ 個の仮想ノードを P に，残りの $\lfloor d/2 \rfloor$ 個の仮想ノードを Q に割り当てればよい．このように決定された P と Q は，可能な組み合わせの中で (35) を最小化する．

5.1.2 近接性を考慮したノード分割

物理ノード p に $m \geq 2$ 個以上の仮想ノード $p.substances = \{v_1, v_2, \ldots, v_m\}$ が融合されている場合，近接性を考慮して m 個の仮想ノードが物理ノード p と q に分割される．このノード分割の手順は，5.1.1 節の S を $S := \{v_1, v_2, \ldots, v_m\}$ に置き換えることにより得られる．

5.2 オーバレイネットワークの構築手法

5.2.1 ノードの参加と離脱

ノードの参加は，近接性を考慮したノード融合 (5.1.1 節) とノード分割 (5.1.2 節) に基づいて行われる．

PDL グラフにおけるノード参加のアルゴリズムは，アルゴリズム 5 に示すとおりである．p を参加ノードとし，p' をブートストラッピングノードとする．まず，p は p' を介してレスポンシブルノード r の探索を行う．$\|r\| = 1$ のとき，DL 反復により d 個の仮想ノードを生成し，それらを近接性に基づいて p' と r に融合する．$\|r\| > 1$ のとき，$r.substances$ を近接性に基づいて p' と r に分割する．

ノードの離脱は，アルゴリズム 5 の逆の手順を実行すればよい．

5.2.2 トポロジのメンテナンスコスト

ここでは簡単のため，d-正則完全グラフ K_{d+1} を初期グラフとする DL グラフに基づくオーバレイネットワークを構築することを考える．

アルゴリズム 5. 近接性を考慮したノード参加

p. **join(p'):**

$r \leftarrow p'$.find_responsible_node(p)
if $|r| > 1$ **:**
  r.split_and_allocate(p)
else:
  r.merge_and_allocate(p)

p. **merge_and_allocate(p'):**

p.generate_virtual_nodes()
p.split_and_allocate(p')

p. **split_and_allocate(p'):**

$S \leftarrow$ sort $v \in p.substances$ in ascending order of
  $h(v) = p$.proximity$(v) - p'$.proximity(v)
$p.substances \leftarrow$ first $\lceil d/2 \rceil$ nodes of S
$p'.substances \leftarrow$ remaining $\lfloor d/2 \rfloor$ nodes of S

p. **proximity(v):**

$T \leftarrow \sum_{u \in v.predecessors} \tau_p(u)$ / $|v.predecessors|$
return T

ノードの参加において，近接性の考慮に必要なメッセージ数は，DL 反復により生成される兄弟頂点の数とそれらの兄弟頂点の入隣接頂点の数に大きく関連する．DL 反復により生成される兄弟頂点は d 個である．また，兄弟頂点の入次数は次の性質を満たすことがいえる．

定理 2. d-正則の初期グラフから生成された DL グラフを G とし，グラフ G の頂点 $v = v_1 v_2 \dots v_\ell$ に関する DL グラフを $G' = DL(G, v)$ とする．ここで生成される兄弟頂点を $s = \alpha v_1 v_2 \dots v_\ell$ $(\alpha \in \Sigma \setminus \{v_1\})$ とするとき，s の入次数は次の条件を満たす．

$$1 \leq \delta^-_{G'}(s) \leq d. \tag{39}$$

証明. [23] の定理 4 より，頂点 v は，入隣接頂点 $w \in N^-_G(v)$ に対して次式を満たしている．

$$|w| \geq |v| = \ell. \tag{40}$$

すなわち，w は次のいずれかの形式となる．

$$w = \begin{cases} w^{(1)} = \alpha v_1 v_2 \dots v_{\ell-1}, \\ w^{(2)} = \beta \alpha v_1 v_2 \dots v_{\ell-1}. \end{cases} \tag{41}$$

ここで，$\alpha \in \Sigma \setminus \{v_1\}$，$\beta \in \Sigma \setminus \alpha$ である．

$w^{(1)}$ は G' において s の唯一の入隣接頂点となる．$w^{(2)}$ は G' において s の入隣接頂点となり，s は $w^{(2)}$ の形式の入隣接頂点をたかだか d 個持つ．

したがって，$\delta^-_{G'}(s)$ はたかだか d となる． □

ノード分割 (アルゴリズム 5) の際，参加ノードとレスポンシブルノードは，兄弟頂点に対応する仮想ノードに対して平均近接性メトリック (すなわち (33) と (34)) を計測する．1 つの仮想ノードに関する平均近接性メトリックの計測に必要なメッセージの数は，仮想ノードの入次数に依存する．これは，定理 2 より $O(d)$ となることが言える．平均近接性メトリックは，d 個の仮想ノードすべてに対して計測されるため，ノードの参加の際に近接性の考慮に必要なメッセージ数は $O(d^2)$ となる．このメッセージ数は，オーバレイネットワーク上のノード数 N によらない．

ノードの離脱の際，近接性の考慮に必要なメッセージ数についても同様に求められる．

6. 評価

本節では，シミュレーションにより，近接性を考慮した分散ライングラフ (PDL グラフ) と分散ライングラフ (DL グラフ) の比較を行い，PDL グラフにおける近接性の考慮によって探索遅延が低減されることを示す．また，PDL グラフが探索遅延を低減する仕組みを明らかにする．

シミュレーションでは，簡単のため初期グラフを完全グラフとし，DL グラフに基づく構造化オーバレイネットワーク (DLG-Kautz) と，PDL グラフに基づく構造化オーバレイネットワーク (PDLG-Kautz) を Python 3 を用いて実装した．DLG-Kautz と PDLG-Kautz は，あらかじめ決められた特定のネットワークトポロジ上に構築する．構築されたオーバレイネットワークにおいて，探索遅延と隣接ノードの通信遅延の 2 つの観点から評価を行う．下位ネットワークトポロジの構築については，6.1 節で述べる．探索遅延に関する評価は 6.2 節で，隣接ノードの通信遅延に関する評価は 6.3 でそれぞれ詳細に述べる．

6.1 下位ネットワークのトポロジ

シミュレーションの設定として，下位ネットワークにおけるノード間の通信遅延を設定した．ここでは，下位ネットワークをインターネットと仮定し，そのトポロジをトランジットスタブモデル (Transit-Stub Model, TS モデル) [26] に基づいて決定した．TS モデルは，インターネットのルータレベルのおおまかな構造を表すモデルである．TS モデルの概略図を図 6 に示す．TS モデルは，トランジットノードとスタブノードと呼ばれるノードから構成されるネットワークモデルである．トランジットノードは特定のトランジットドメインに属し，他のトランジットドメインに属するトランジットノードと接続している．また，トランジットノードはいくつかのスタブドメインに接続している．あるスタブドメインには，いくつかのスタブノードが属している．スタブドメインはインターネットサービスプロバイダに相当し，トランジットドメインはその上位のインターネットサービスプロバイダに相当する．また，スタブノードはユーザがインターネットに接続する際のエンドポイントに相当する．

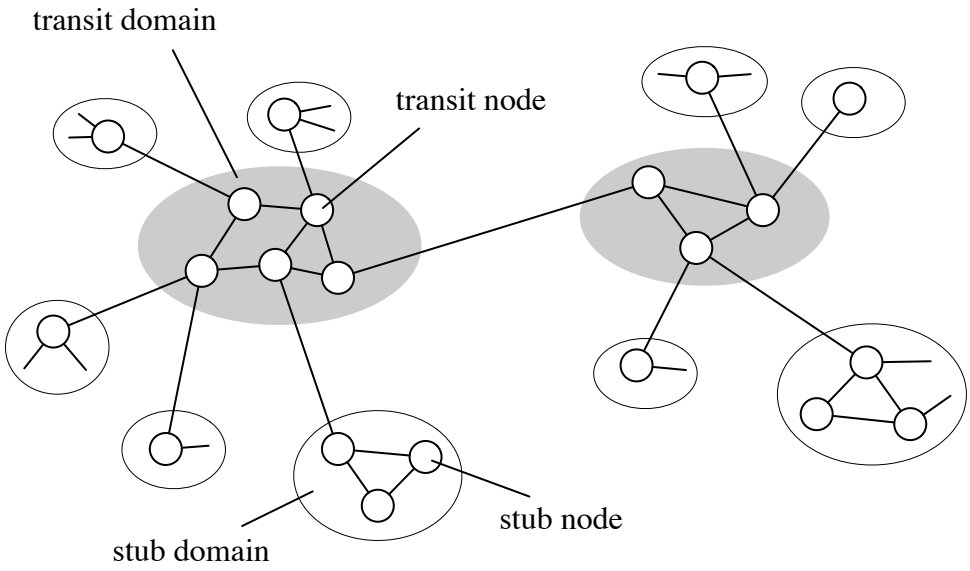


図 6　TS モデル

シミュレーションでは，トランジットドメイン数を 10，トランジットドメインごとのトランジットノード数の平均を 5，トランジットノードが接続しているスタブドメイン数の平均を 10，スタブドメインごとのスタブノード数の平均を 10 として，TS モデルに基づくネットワークトポロジを GT-ITM [3] により 10 種類生成した．オーバレイネットワークにおける物理ノードは，生成したネットワークのスタブノードにランダムに割り当て，それらの間の通信遅延は生成したネットワークで Dijkstra 法により最短経路を求めて決定した．生成した 10 種類のネットワークトポロジについて，スタブノード間の通信遅延の分布は図 7 のようになった．ここで決定したスタブノード間の通信遅延の値は，実際のインターネットとは大きく異なる．シミュレーションでは，通信遅延や探索遅延の削減値ではなく，それらの削減率によって評価を行う．そのため，通信遅延の値が実際のインターネットとはかけ離れていたとしても，妥当な評価を行うことができると考えられる．

[3]入手先: https://www.cc.gatech.edu/projects/gtitm/

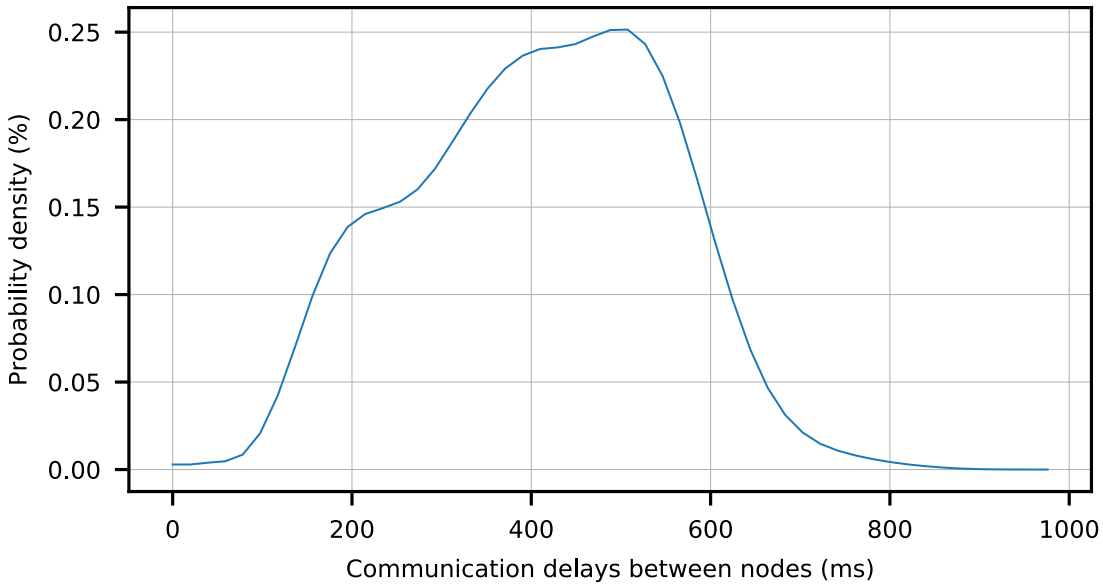


図 7　スタブノード間の遅延の確率密度

6.2　平均探索遅延と融合されている仮想ノードの数

本節では，DL グラフと PDL グラフを探索遅延の観点から比較する．

まず，6.1 節で述べたように 10 種類の下位ネットワークトポロジを生成した．そして，d-正則完全有向グラフ K_{d+1} $(d = 2, 4, 8, 16)$ を初期グラフとして，DL グラフと PDL グラフに基づく構造化オーバレイネットワークを構築した．簡単のため，これらのネットワークをそれぞれ DLG-Kautz，PDLG-Kautz と表記する．このとき，ノード数は $N = 2^8, 2^9, \ldots, 2^{16}$ とした．

オーバレイネットワークの構築後，ランダムに選択したノードからランダムに選択したノードへの探索メッセージを 10^6 回 [4]送信した．そして，それぞれの場合で探索遅延と探索ホップ数，各ノードに融合されている仮想ノードの数を測定した．シミュレーションの結果を表 3 から表 5 に示す．上記の結果に基づき，探索遅延の平均と融合されている仮想ノードの数の平均の関係を図 8 に示す．

表 4, 5 より，探索ホップ数と融合されている仮想ノードの数は DLG-Kautz と

[4] ノード数より十分に大きい値とした．

PDLG-Kautz で変化しないことがいえる．

表 3 と図 8 より，$d = 2$ の場合，平均探索遅延の削減率はノード数によらず 10% 程度である．一方，$d \neq 2$ の場合，平均探索遅延の削減率はノード数に伴って変化している．いずれの場合も，PDLG-Kautz の平均探索遅延は DLG-Kautz の平均探索遅延よりも低いことがわかる．したがって，PDLG-Kautz における近接性を考慮したノードの参加アルゴリズムが探索遅延を低減することがいえる．

図 8 より，PDLG-Kautz では，各ノードに融合されている仮想ノードの数が小さい場合に平均探索遅延の削減率が小さく，融合されている仮想ノードの数が大きい場合に平均探索遅延の削減率が大きいことが確認できる．各ノードに融合されている仮想ノードの数が大きいという状況は，近接性を考慮したノード融合が多数実行されている状況に等しい．また，各ノードに融合されている仮想ノードの数が小さいという状況は，近接性を考慮したノード分割が多数実行されている状況に等しい．すなわち，近接性を考慮したノード融合が探索遅延の低減に大きく影響していることがわかる．この現象については，次節で詳細に分析する．

表 3　探索遅延の平均 (ミリ秒)

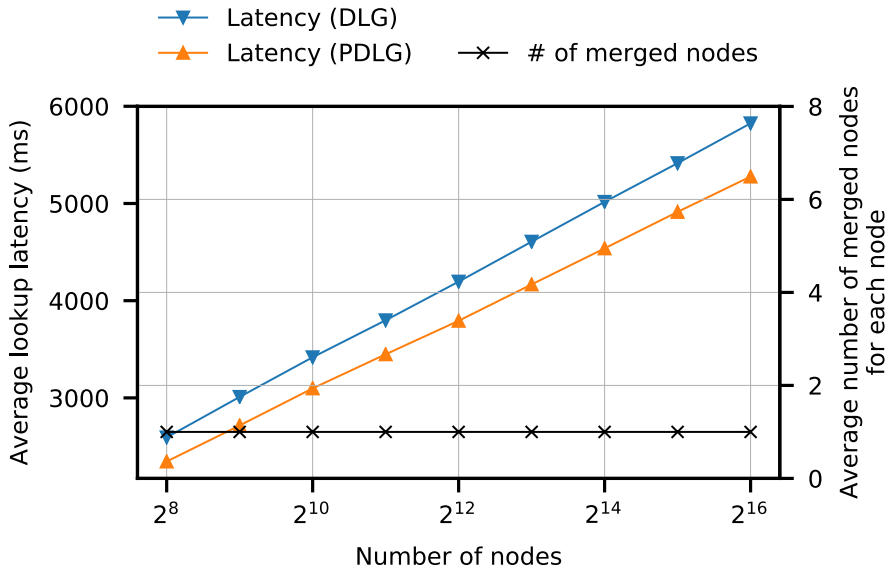
	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$	$N = 2^{15}$	$N = 2^{16}$
A. DLG-Kautz ($d = 2$)	2593.241	3009.812	3417.918	3798.312	4195.953	4607.048	5016.480	5414.776	5824.640
B. PDLG-Kautz ($d = 2$)	2345.392	2716.382	3097.809	3448.334	3793.007	4167.200	4538.063	4913.333	5276.917
$(\mathrm{A} - \mathrm{B})/(\mathrm{A} + \mathrm{B})$	9.557%	9.749%	9.366%	9.214%	9.603%	9.547%	9.537%	9.261%	9.404%
A. DLG-Kautz ($d = 4$)	1477.451	1804.044	1898.487	2193.426	2305.602	2606.575	2712.699	3015.994	3119.117
B. PDLG-Kautz ($d = 4$)	1390.835	1633.291	1764.767	2005.807	2124.827	2367.834	2508.341	2743.000	2879.365
$(\mathrm{A} - \mathrm{B})/(\mathrm{A} + \mathrm{B})$	5.863%	9.465%	7.044%	8.554%	7.841%	9.159%	7.533%	9.052%	7.687%
A. DLG-Kautz ($d = 8$)	1176.523	1172.819	1517.434	1572.236	1578.054	1925.691	1978.698	1981.658	2331.427
B. PDLG-Kautz ($d = 8$)	1077.376	1110.572	1382.195	1449.988	1490.892	1747.155	1825.291	1879.245	2117.530
$(\mathrm{A} - \mathrm{B})/(\mathrm{A} + \mathrm{B})$	8.427%	5.307%	8.912%	7.775%	5.523%	9.271%	7.753%	5.168%	9.175%
A. DLG-Kautz ($d = 16$)	801.204	1172.366	1200.941	1196.386	1199.171	1572.053	1605.108	1603.435	1605.379
B. PDLG-Kautz ($d = 16$)	769.980	1061.196	1101.459	1131.620	1156.459	1425.525	1476.376	1514.993	1549.104
$(\mathrm{A} - \mathrm{B})/(\mathrm{A} + \mathrm{B})$	3.897%	9.483%	8.284%	5.413%	3.562%	9.321%	8.020%	5.516%	3.505%

表 4　探索ホップ数の平均

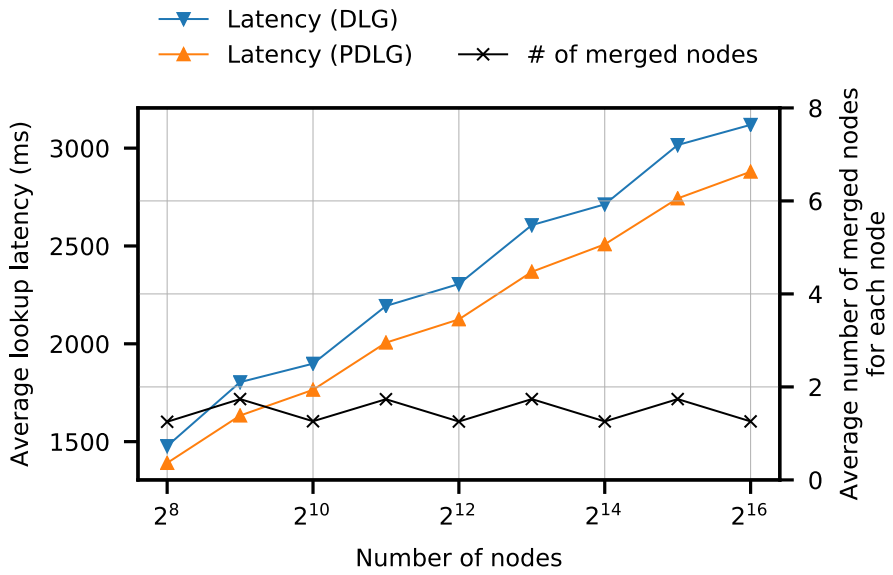
	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$	$N = 2^{15}$	$N = 2^{16}$
DLG-Kautz ($d = 2$)	6.319	7.308	8.301	9.292	10.289	11.292	12.290	13.287	14.288
PDLG-Kautz ($d = 2$)	6.315	7.311	8.296	9.293	10.292	11.289	12.287	13.286	14.287
DLG-Kautz ($d = 4$)	3.646	4.390	4.655	5.391	5.651	6.395	6.654	7.395	7.654
PDLG-Kautz ($d = 4$)	3.648	4.391	4.656	5.393	5.655	6.395	6.655	7.396	7.655
DLG-Kautz ($d = 8$)	2.845	2.859	3.717	3.853	3.864	4.723	4.854	4.860	5.723
PDLG-Kautz ($d = 8$)	2.845	2.860	3.718	3.853	3.861	4.723	4.855	4.862	5.723
DLG-Kautz ($d = 16$)	1.941	2.862	2.929	2.932	2.941	3.862	3.933	3.933	3.940
PDLG-Kautz ($d = 16$)	1.937	2.859	2.930	2.932	2.940	3.862	3.933	3.933	3.940

表 5　各ノードに融合されている仮想ノードの数の平均

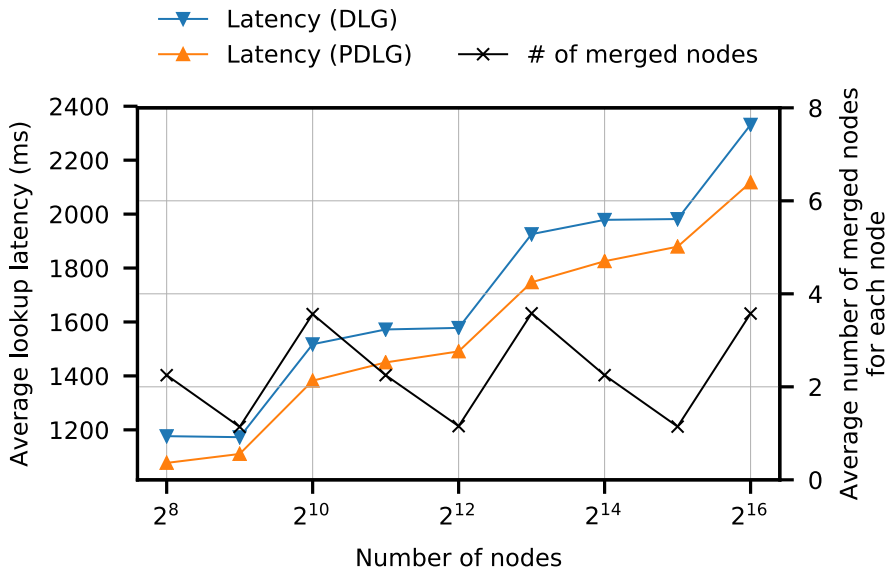
	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$	$N = 2^{15}$	$N = 2^{16}$
DLG-Kautz ($d = 2$)	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
PDLG-Kautz ($d = 2$)	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
DLG-Kautz ($d = 4$)	1.250	1.738	1.262	1.735	1.256	1.738	1.258	1.738	1.258
PDLG-Kautz ($d = 4$)	1.251	1.742	1.263	1.737	1.260	1.740	1.259	1.739	1.259
DLG-Kautz ($d = 8$)	2.250	1.139	3.563	2.250	1.154	3.581	2.250	1.145	3.576
PDLG-Kautz ($d = 8$)	2.250	1.144	3.576	2.250	1.148	3.584	2.250	1.149	3.579
DLG-Kautz ($d = 16$)	1.121	7.533	4.250	2.125	1.125	7.440	4.250	2.125	1.111
PDLG-Kautz ($d = 16$)	1.092	7.504	4.250	2.125	1.119	7.443	4.250	2.125	1.117


Latency (DLG)
Latency (PDLG)
# of merged nodes
Average lookup latency (ms)
6000
5000
4000
3000
Average number of merged nodes for each node
8
6
4
2
0
2^8
2^{10}
2^{12}
2^{14}
2^{16}
Number of nodes

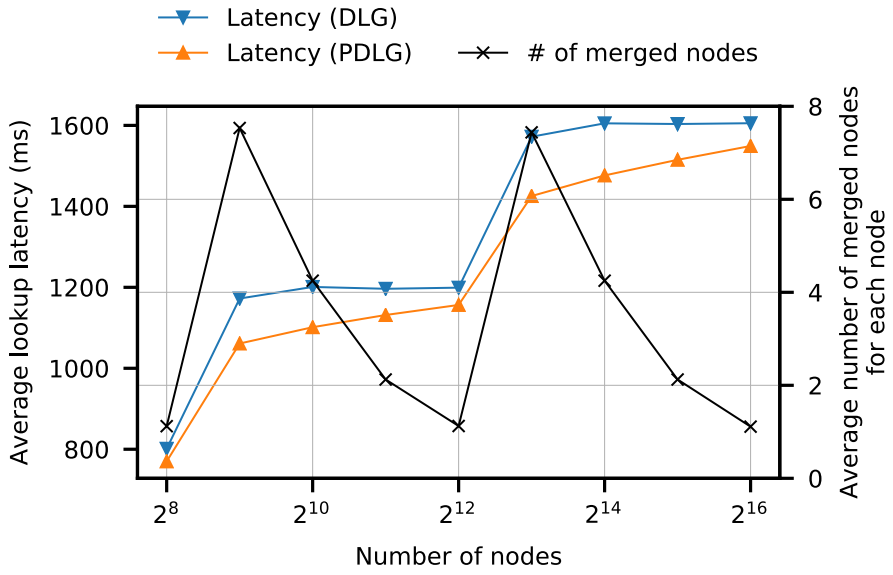
(a) $d = 2$


Latency (DLG)
Latency (PDLG)
# of merged nodes
Average lookup latency (ms)
3000
2500
2000
1500
Average number of merged nodes for each node
8
6
4
2
0
2^8
2^{10}
2^{12}
2^{14}
2^{16}
Number of nodes

(b) $d = 4$



(c) $d = 8$



(d) $d = 16$

図 8　平均探索遅延と各ノードに融合されている仮想ノードの数の関係

6.3 各ノードとその出隣接ノード間の通信遅延

本節では，PDLG-Kautz における近接性を考慮したノードの融合と分割が探索遅延に与える影響を詳細に分析する．具体的には，DLG-Kautz と PDLG-Kautz について，各ノードとその出隣接ノード間の通信遅延を分析し，近接性を考慮したノードの融合と分割の影響を調査する．

シミュレーションでは，d-正則完全有向グラフ K_{d+1} ($d = 2, 4, 8, 16$) を初期グラフとして，DLG-Kautz と PDLG-Kautz を構築した．ノード数 $N = 2^8, 2^9, \ldots, 2^{16}$ のそれぞれの場合で，各ノードとその出隣接ノード間の通信遅延を計測し，その分布を調べた．各ノードとその出隣接ノード間の通信遅延の平均を表 6，標準偏差を表 7 に示す．また，各ノードとその出隣接ノード間の通信遅延の分布を図 9 から図 12 に示す．

図 9 は，$d = 2$ のときの各ノードとその出隣接ノード間の通信遅延の分布である．$d = 2$ の DLG-Kautz と PDLG-Kautz では，ノード融合によりネットワークトポロジが拡大される．ノード融合の際，1 つの仮想ノードに関して DL 反復が行われ $d = 2$ 個の仮想ノードが生成される．生成された 2 つの仮想ノードは，参加ノードとレスポンシブルノードの 2 つの物理ノードで共有されるため，それぞれのノードは必ず 1 つの仮想ノードに対応し，いかなる場合もノード分割が行われることはない．ここで，参加ノードにはランダムに決められたスタブノードが割り当てられており，レスポンシブルノードは一様分布するハッシュ値に基づく探索により得られたノードである．そして，2 つの仮想ノードは参加ノードやレスポンシブルノードとは無関係に生成される．そのため，参加ノードで観測される仮想ノードに関する平均近接性メトリックと，レスポンシブルノードで観測される仮想ノードに関する平均近接性メトリックは，高確率で大きく異なる．PDLG-Kautz では，このような場合に各ノードとその出隣接ノード間の通信遅延を大きく削減可能である．表 6 の $d = 2$ の場合や，それをグラフにした図 9 のすべての場合で，PDLG-Kautz における各ノードとその出隣接ノード間の通信遅延の分布は DLG-Kautz のものよりも優れていることからも，この事実は明らかである．

図 10, 11, 12 は，$d \neq 2$ のときの各ノードとその出隣接ノード間の通信遅延

の分布である．6.2 節で述べたとおり，PDLG-Kautz では，ノード分割が多数実行された場合に平均探索遅延の削減率が小さく，ノード融合が多数実行された場合に平均探索遅延の削減率が大きい．先に述べたとおり，PDLG-Kautz は，参加ノードで観測される仮想ノードに関する平均近接性メトリックと，レスポンシブルノードで観測される仮想ノードに関する平均近接性メトリックが高確率で大きく異なる場合に，各ノードとその出隣接ノード間の通信遅延を低減することができる．$d \neq 2$ の DLG-Kautz，PDLG-Kautz では，ノード融合，あるいはノード分割によりネットワークトポロジが拡大される．アルゴリズム 3 からわかるように，DLG-Kautz や PDLG-Kautz では，ノード分割可能なノードが存在すればそのノードを優先して参加ノードに対応するレスポンシブルノードとしている．すなわち，DLG-Kautz や PDLG-Kautz には，オーバレイネットワーク上のノード数が増加するに伴って，

(1) ノード融合が頻繁に発生する段階

(2) ノード分割が頻繁に発生する段階

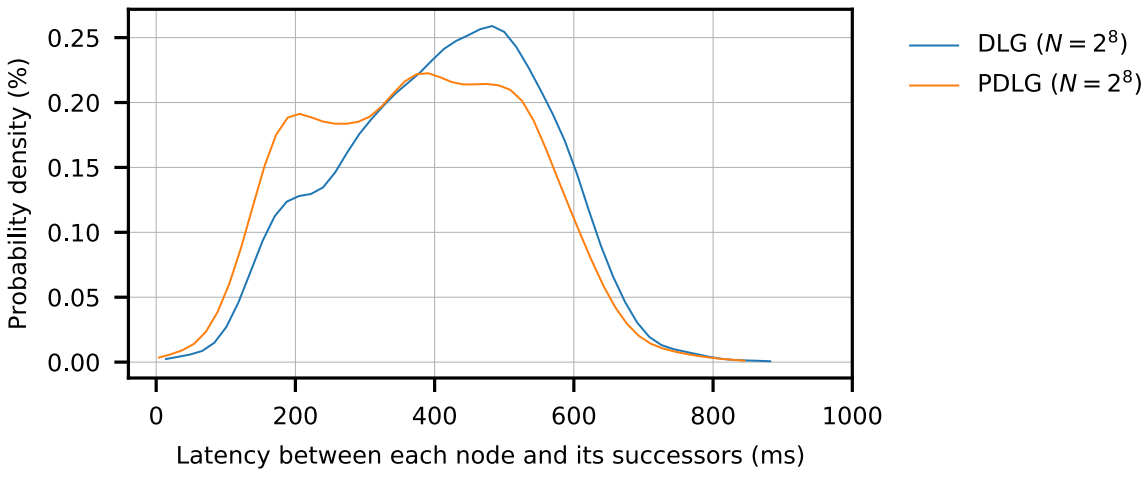
の 2 つの段階が存在する．(1) では，DL 反復により d 個の新たな仮想ノードが生成される．この場合，d 個の仮想ノードは参加ノードやレスポンシブルノードとは無関係に生成されるため，参加ノードやレスポンシブルノードで観測される仮想ノードに関する平均近接性メトリックの分散は大きい．一方，(2) では，仮想ノードは近接性に基づいて最適な物理ノードに割り当てられている．したがって，1 つの物理ノードに割り当てられている仮想ノードに関して参加ノードやレスポンシブルノードから観測される平均近接性メトリックの分散は，(1) の場合よりも小さくなる．図 10, 11, 12 と表 5, 6 からわかるように，ノード融合が頻繁に発生する段階 (すなわち，1 つのノードに割り当てられている仮想ノード数が大きい場合) では，PDLG-Kautz における各ノードとその出隣接ノード間の通信遅延の分布は DLG-Kautz よりも優れていることが確認できる．一方，表 6, 7 から，ノード分割が頻繁に発生する段階 (すなわち，1 つのノードに割り当てられている仮想ノード数が小さい場合) では，PDLG-Kautz における各ノードとその出隣接ノード間の通信遅延の分布は DLG-Kautz における通信遅延の分布に近づいていくことがわかる．

表 6　各ノードとその出隣接ノード間の通信遅延の平均 (ミリ秒)

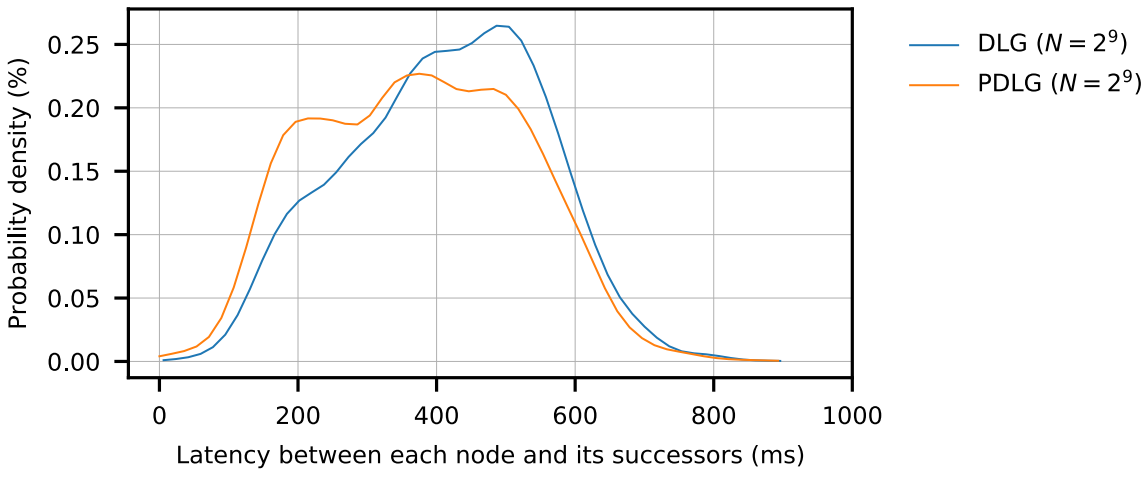
	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$	$N = 2^{15}$	$N = 2^{16}$
A. DLG-Kautz ($d = 2$)	411.603	412.280	411.541	409.046	407.781	407.603	408.000	407.355	407.644
B. PDLG-Kautz ($d = 2$)	375.497	374.896	376.834	374.435	372.311	372.684	373.009	373.501	373.020
$(\mathrm{A} - \mathrm{B})/(\mathrm{A} + \mathrm{B})$	4.587%	4.749%	4.402%	4.418%	4.547%	4.475%	4.480%	4.336%	4.435%
A. DLG-Kautz ($d = 4$)	405.738	409.955	407.790	406.746	408.161	407.303	407.605	407.955	407.494
B. PDLG-Kautz ($d = 4$)	383.291	375.385	381.622	375.415	378.420	373.847	379.928	374.329	379.188
$(\mathrm{A} - \mathrm{B})/(\mathrm{A} + \mathrm{B})$	2.845%	4.402%	3.315%	4.006%	3.781%	4.283%	3.514%	4.298%	3.598%
A. DLG-Kautz ($d = 8$)	413.585	410.297	408.124	407.872	408.331	407.617	407.552	407.716	407.399
B. PDLG-Kautz ($d = 8$)	380.253	389.296	374.119	378.097	387.582	372.430	378.071	387.967	372.489
$(\mathrm{A} - \mathrm{B})/(\mathrm{A} + \mathrm{B})$	4.199%	2.626%	4.347%	3.788%	2.607%	4.511%	3.753%	2.482%	4.476%
A. DLG-Kautz ($d = 16$)	412.774	409.795	410.190	408.066	407.583	407.173	408.092	407.671	407.426
B. PDLG-Kautz ($d = 16$)	398.280	372.592	377.026	386.822	393.743	370.745	376.674	386.134	393.758
$(\mathrm{A} - \mathrm{B})/(\mathrm{A} + \mathrm{B})$	1.787%	4.755%	4.213%	2.673%	1.727%	4.683%	4.003%	2.713%	1.706%

表 7　各ノードとその出隣接ノード間の通信遅延の標準偏差

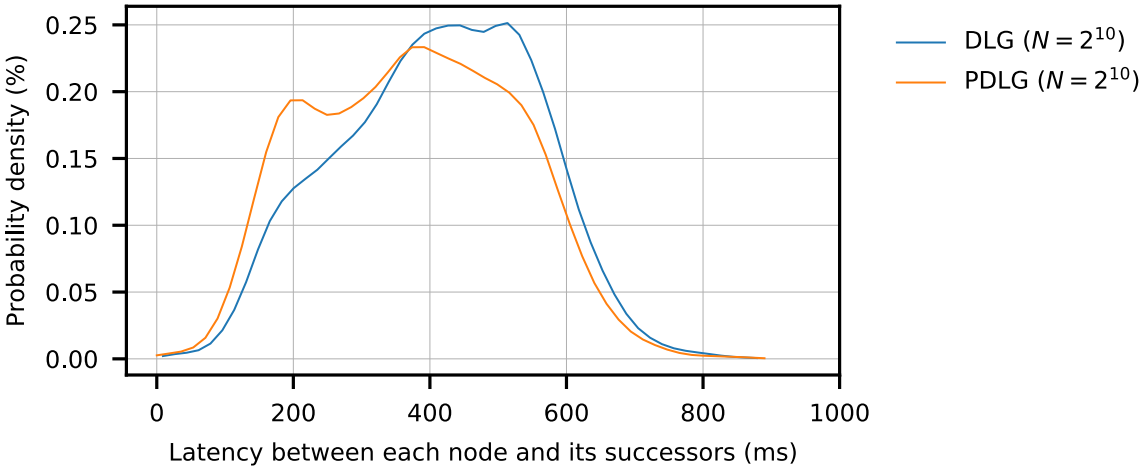
	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$	$N = 2^{15}$	$N = 2^{16}$
DLG-Kautz ($d = 2$)	142.737	140.974	142.353	143.723	143.409	143.876	143.844	144.241	143.670
PDLG-Kautz ($d = 2$)	149.419	148.505	147.154	148.353	148.583	148.477	149.065	148.738	148.815
DLG-Kautz ($d = 4$)	143.305	143.368	143.558	143.733	143.400	143.858	144.117	143.797	143.893
PDLG-Kautz ($d = 4$)	147.756	146.983	146.432	147.822	147.592	148.144	148.281	148.326	148.274
DLG-Kautz ($d = 8$)	141.512	143.184	143.568	143.407	143.544	143.695	144.156	143.886	143.883
PDLG-Kautz ($d = 8$)	147.631	147.131	146.946	147.979	147.542	147.257	148.418	147.751	147.249
DLG-Kautz ($d = 16$)	142.393	142.722	142.399	143.514	143.558	143.829	143.890	143.933	143.887
PDLG-Kautz ($d = 16$)	145.899	145.261	147.597	147.783	146.464	146.485	148.313	148.003	146.758


0.25
0.20
0.15
0.10
0.05
0.00
Probability density (%)
0
200
400
600
800
1000
Latency between each node and its successors (ms)
DLG ($N = 2^8$)
PDLG ($N = 2^8$)

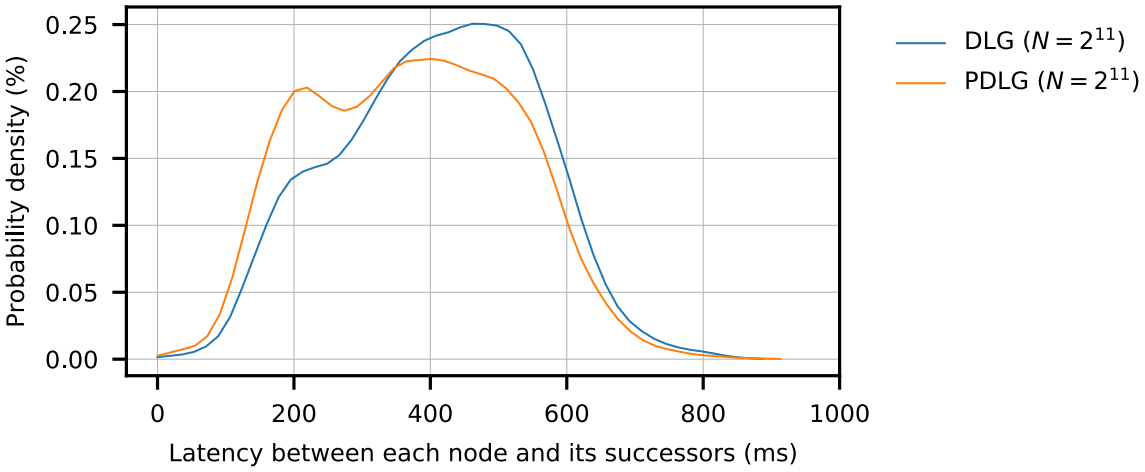
(a) $N = 2^8$


0.25
0.20
0.15
0.10
0.05
0.00
Probability density (%)
0
200
400
600
800
1000
Latency between each node and its successors (ms)
DLG ($N = 2^9$)
PDLG ($N = 2^9$)

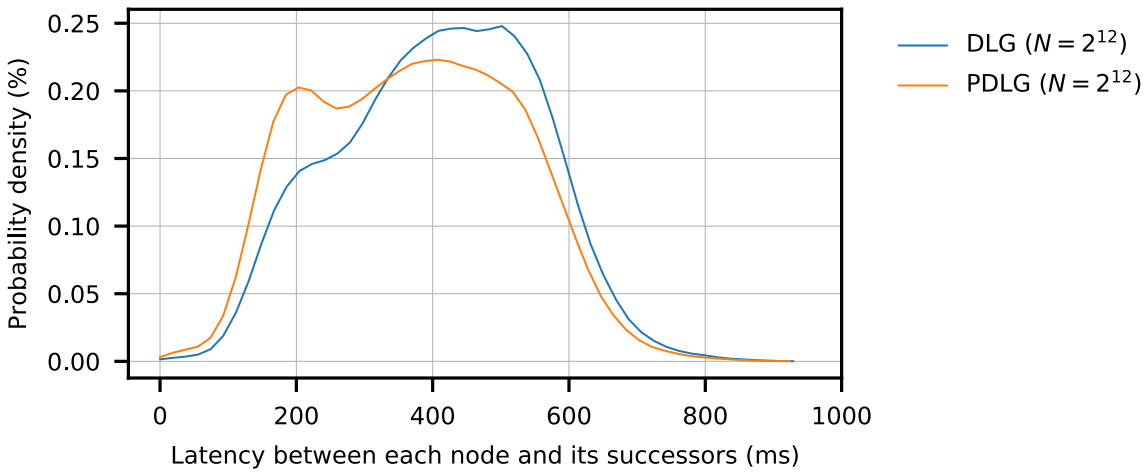
(b) $N = 2^9$


0.25
0.20
0.15
0.10
0.05
0.00
Probability density (%)
0
200
400
600
800
1000
Latency between each node and its successors (ms)
DLG ($N = 2^{10}$)
PDLG ($N = 2^{10}$)

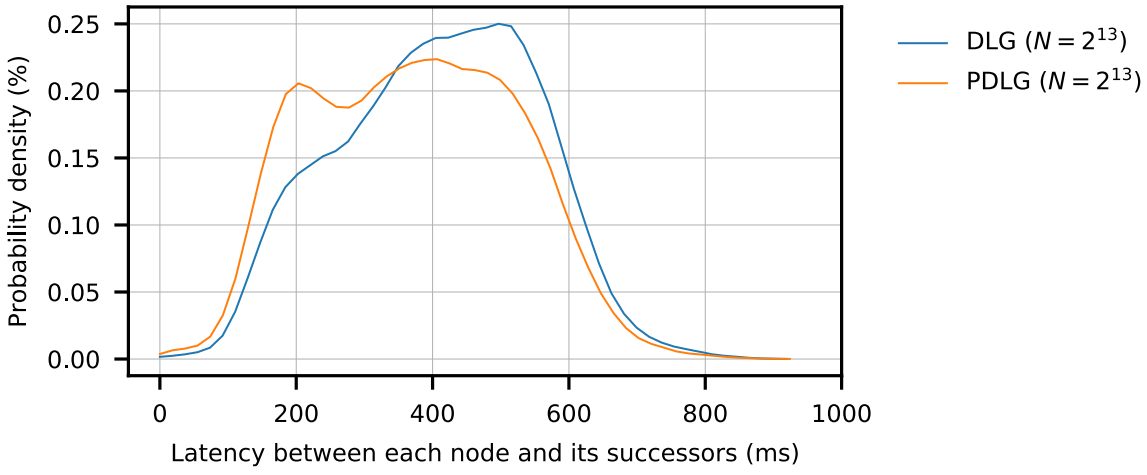
(c) $N = 2^8$


0.25
0.20
0.15
0.10
0.05
0.00
Probability density (%)
0
200
400
600
800
1000
Latency between each node and its successors (ms)
DLG ($N = 2^{11}$)
PDLG ($N = 2^{11}$)

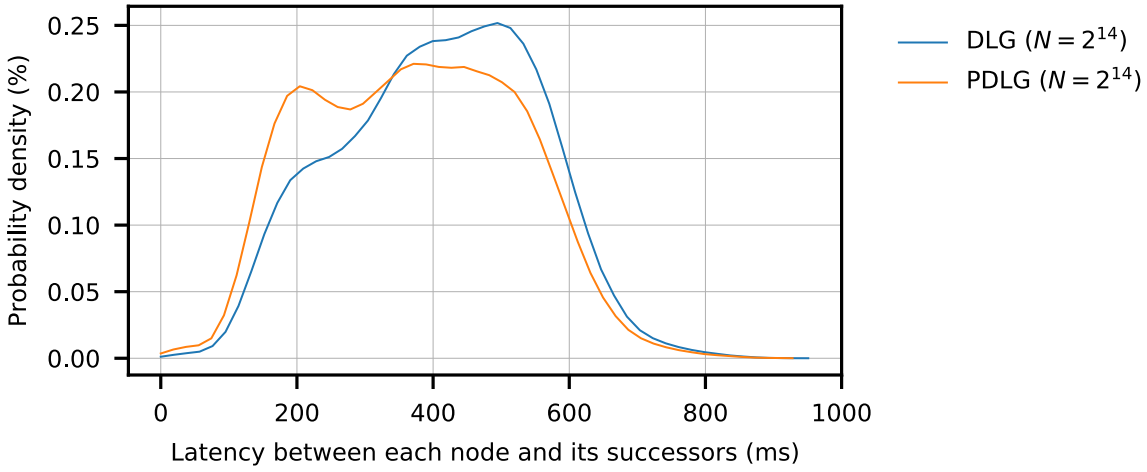
(d) $N = 2^9$


0.25
0.20
0.15
0.10
0.05
0.00
Probability density (%)
0
200
400
600
800
1000
Latency between each node and its successors (ms)
DLG ($N = 2^{12}$)
PDLG ($N = 2^{12}$)

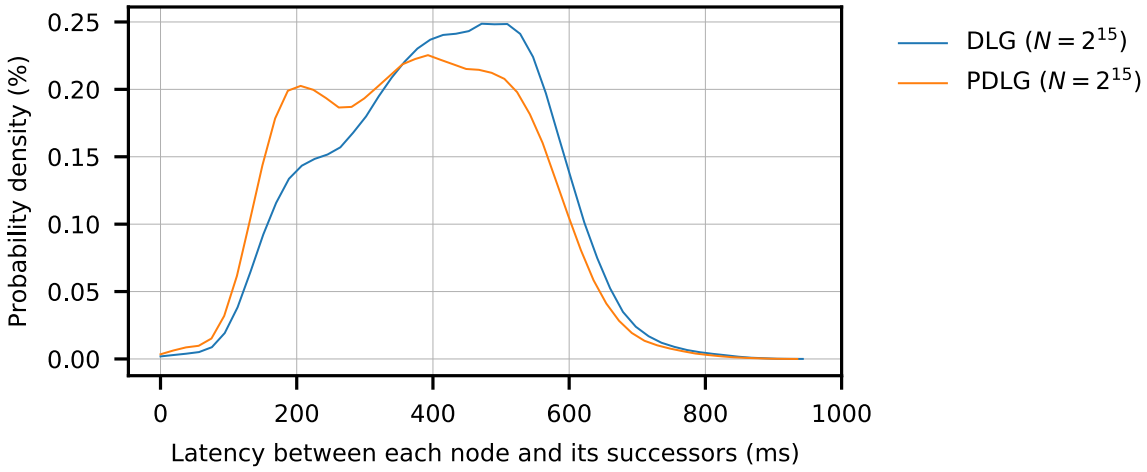
(e) $N = 2^8$


0.25
0.20
0.15
0.10
0.05
0.00
Probability density (%)
0
200
400
600
800
1000
Latency between each node and its successors (ms)
DLG ($N = 2^{13}$)
PDLG ($N = 2^{13}$)

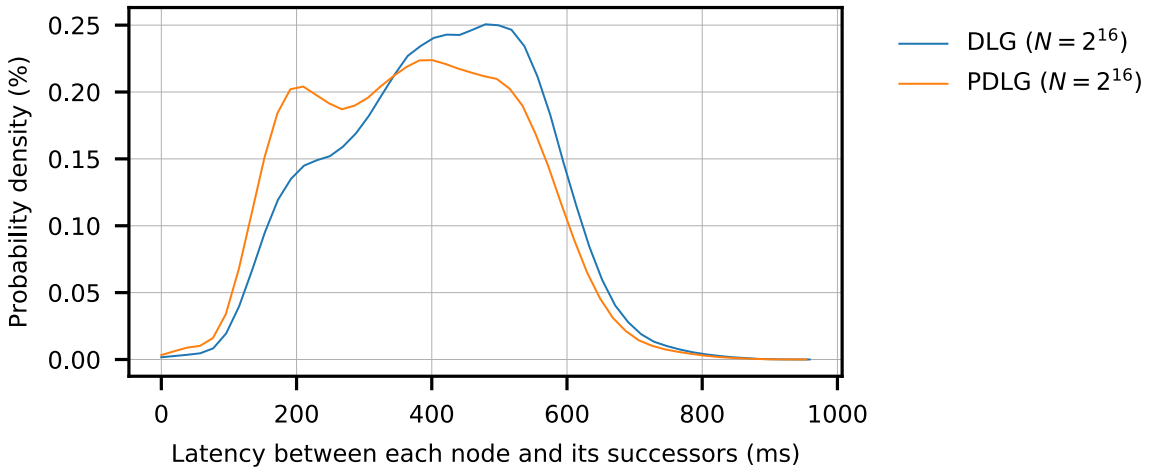
(f) $N = 2^9$


0.25
0.20
0.15
0.10
0.05
0.00
Probability density (%)
0
200
400
600
800
1000
Latency between each node and its successors (ms)
DLG ($N = 2^{14}$)
PDLG ($N = 2^{14}$)

(g) $N = 2^8$

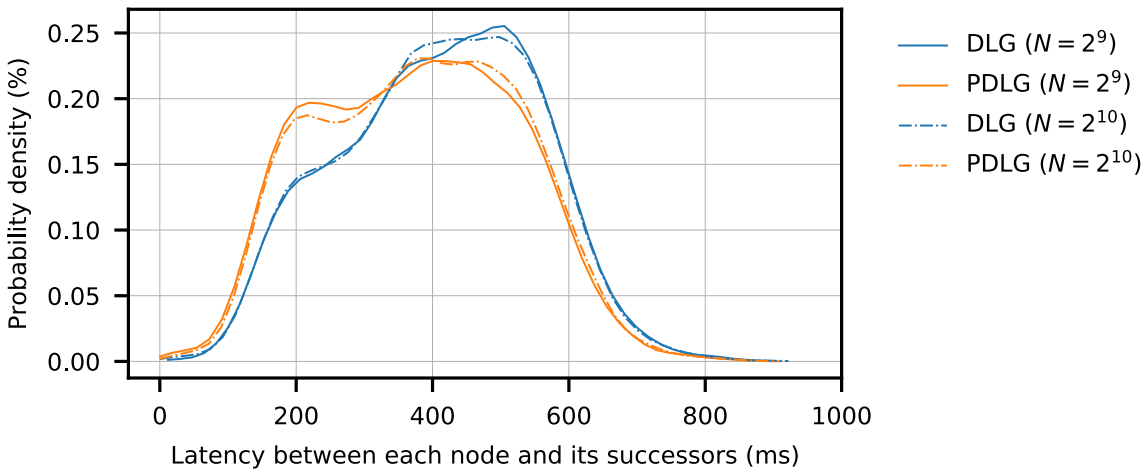

0.25
0.20
0.15
0.10
0.05
0.00
Probability density (%)
0
200
400
600
800
1000
Latency between each node and its successors (ms)
DLG ($N = 2^{15}$)
PDLG ($N = 2^{15}$)

(h) $N = 2^9$

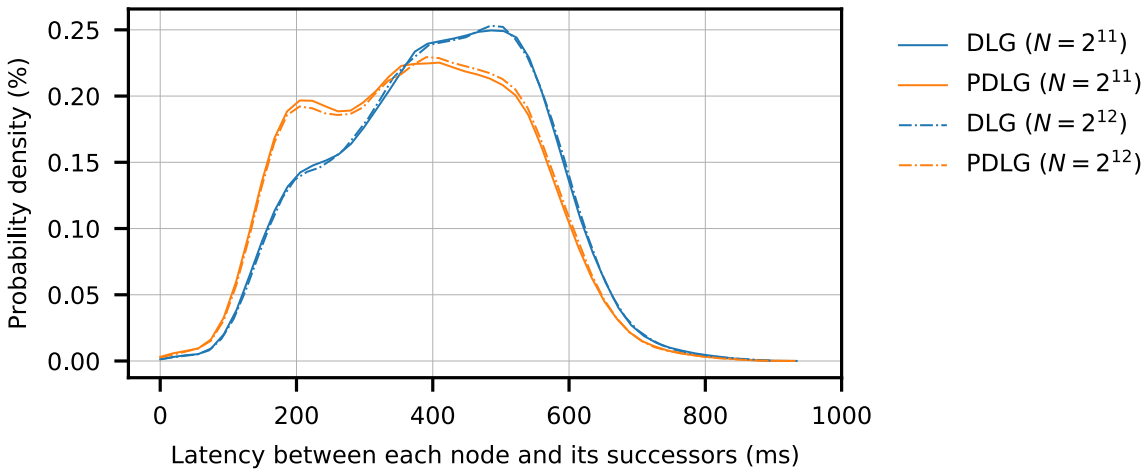


(i) $N = 2^{16}$

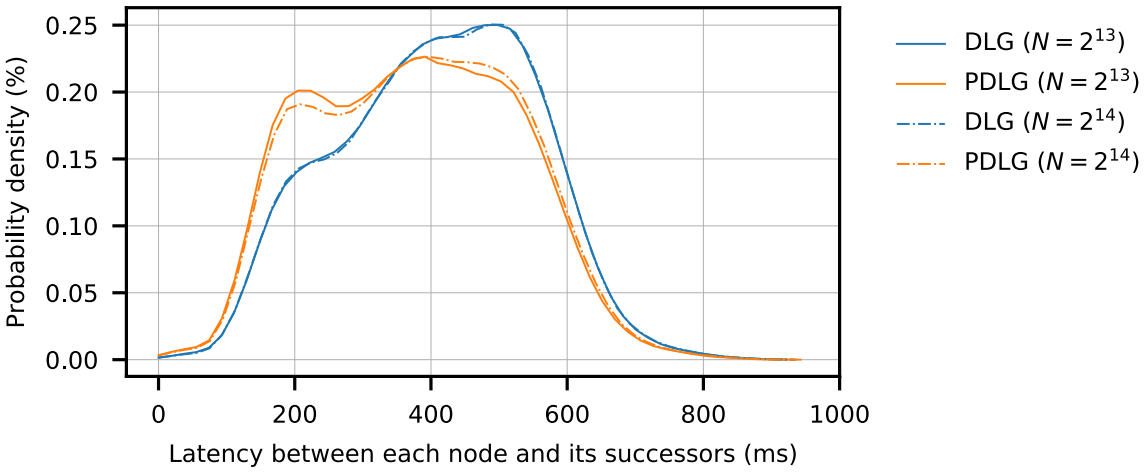
図 9　各ノードとその出隣接ノード間の通信遅延の確率密度 $(d = 2)$



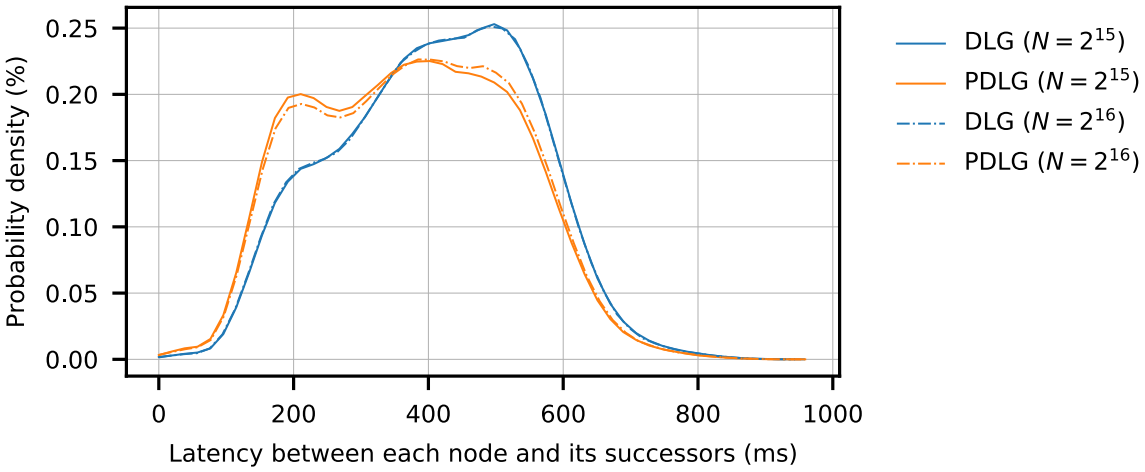
(a) $N = 2^9, 2^{10}$



(b) $N = 2^{11}, 2^{12}$

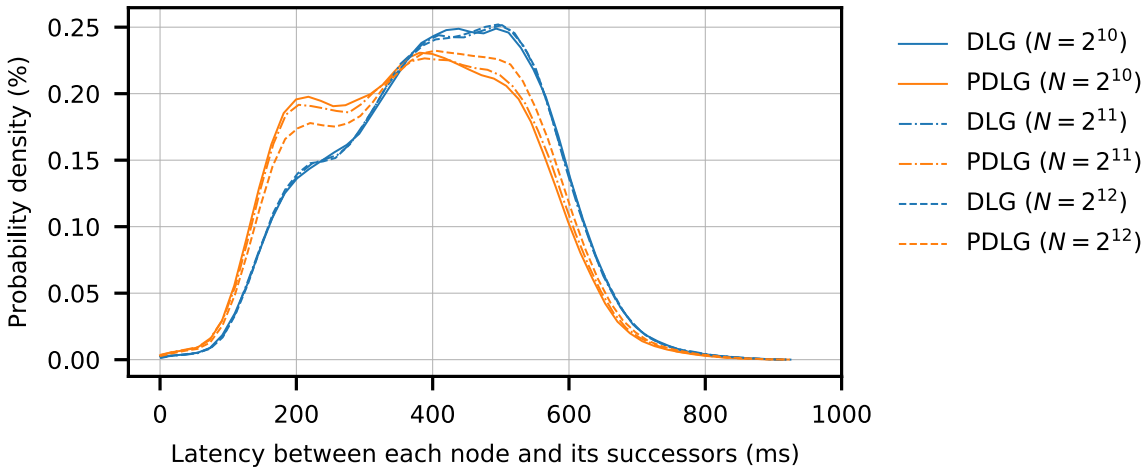


(c) $N = 2^{13}, 2^{14}$

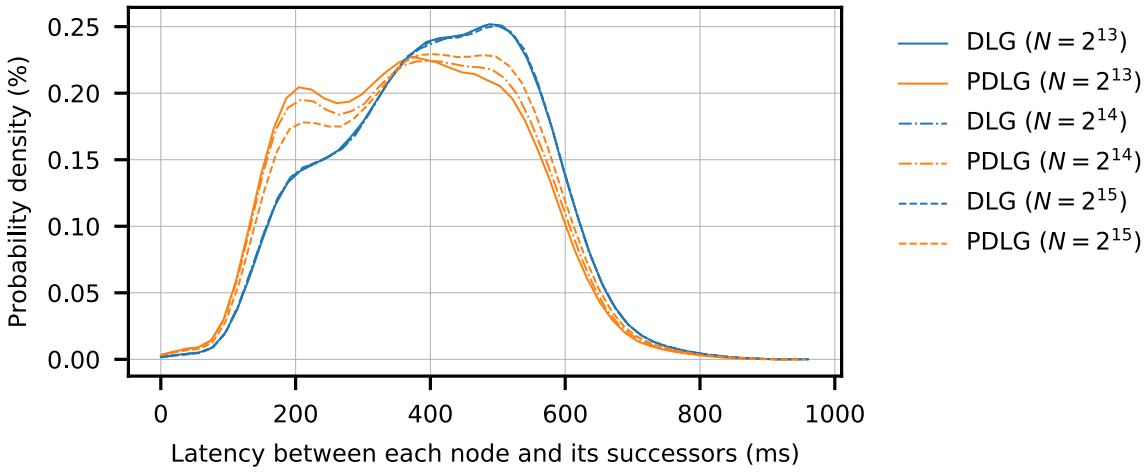


(d) $N = 2^{15}, 2^{16}$

図 10　各ノードとその出隣接ノード間の通信遅延の確率密度 $(d = 4)$

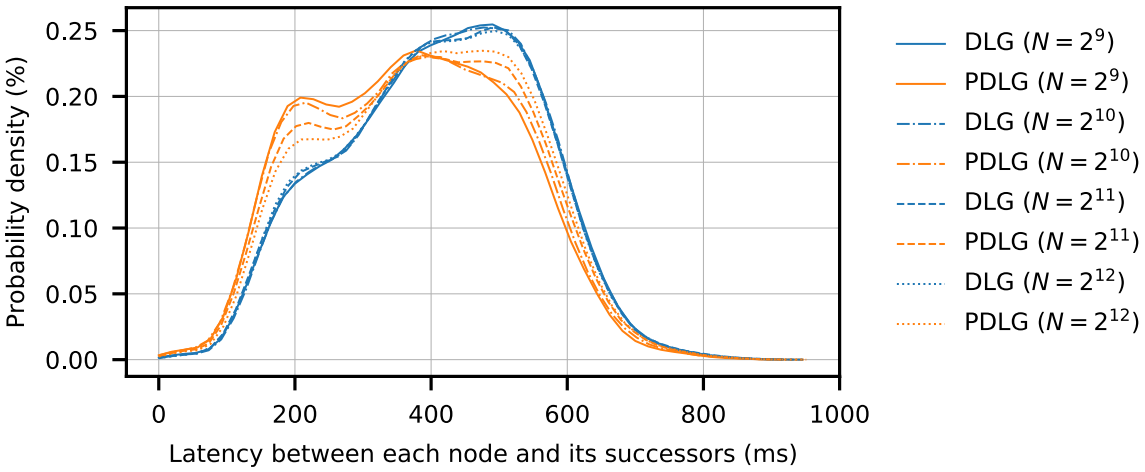


(a) $N = 2^{10}, 2^{11}, 2^{12}$

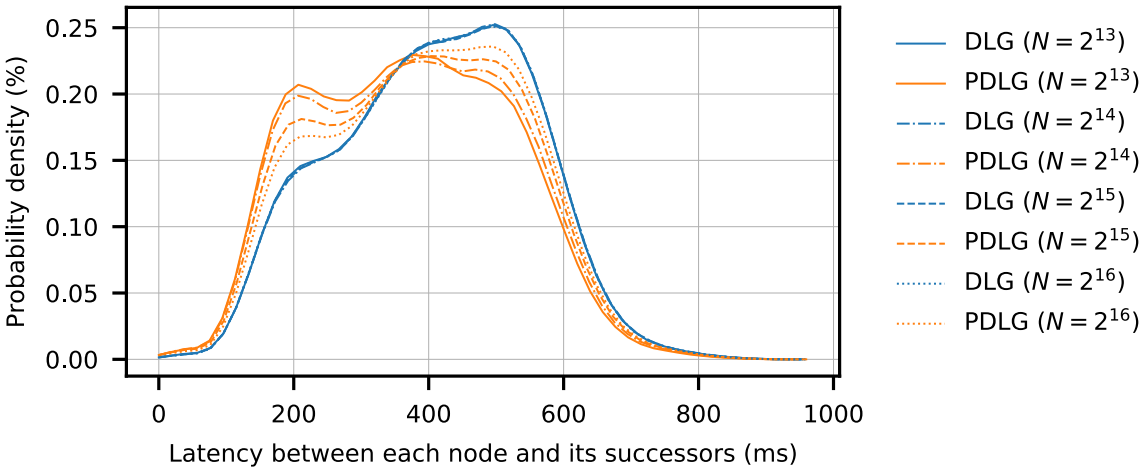


(b) $N = 2^{13}, 2^{14}, 2^{15}$

図 11　各ノードとその出隣接ノード間の通信遅延の確率密度 $(d = 8)$



(a) $N = 2^9, 2^{10}, 2^{11}, 2^{12}$



(b) $N = 2^{13}, 2^{14}, 2^{15}, 2^{16}$

図 12　各ノードとその出隣接ノード間の通信遅延の確率密度 $(d = 16)$

7. 応用

7.1 エッジコンピューティング環境におけるオブジェクトストア

Multi-Access Edge Computing (MEC) やフォグコンピューティングをはじめとするエッジコンピューティングと呼ばれるパラダイムは，リアルタイム性を必要とする Internet of Things (IoT) サービスの要件を満たすために設計された．低遅延が求められるリアルタイムサービスの実現には高速なストレージが必要であり，エッジコンピューティングにおけるオブジェクトストアの需要は極めて高い．このようなエッジコンピューティング環境において，IoT サービスをサポートするためのストレージを実現するための研究が行われている [27, 28]．

Confais ら [27] が提案しているオブジェクトストアは，Scale-out Network Attached Storage (SONAS) のひとつである RozoFS [5] と，DHT (厳密には Kademlia [2] の実装) のひとつである InterPlanetary File System (IPFS) [6] を用いて構成される．図 13 に Confais らのオブジェクトストアの概略図を示す．Confais らが提案している分散オブジェクトストアは，挿入したオブジェクトや取得したオブジェクトをクライアントが存在しているサイトと同一のサイト (ローカルサイト) のノードに格納し，オブジェクトが格納されているノードの接続情報を分散管理している．

オブジェクトの挿入，取得を行う際のトラフィックの流れを図 14 に示す．図 14 (a) は，ローカルサイトのストレージノードにオブジェクトの格納を行う場合のトラフィックの流れである．クライアントは，ローカルサイトから RozoFS のノードを一つ選択し，そのノードを介して RozoFS にオブジェクトを格納する．このとき，クライアントにより選択されたノードは，RozoFS のメタデータサーバー (MDS) にオブジェクトの格納先ノードの問い合わせを行う．また，オブジェクトの格納先ノードは，格納しているオブジェクトの識別子と自身の接続情報を IPFS に登録する．図 14 (b) は，ローカルサイトのノードからオブジェクトの取得を行う場合のトラフィックの流れである．クライアントは，ローカルサイトからノー

[5] https://www.rozosystems.com/
[6] https://ipfs.io/

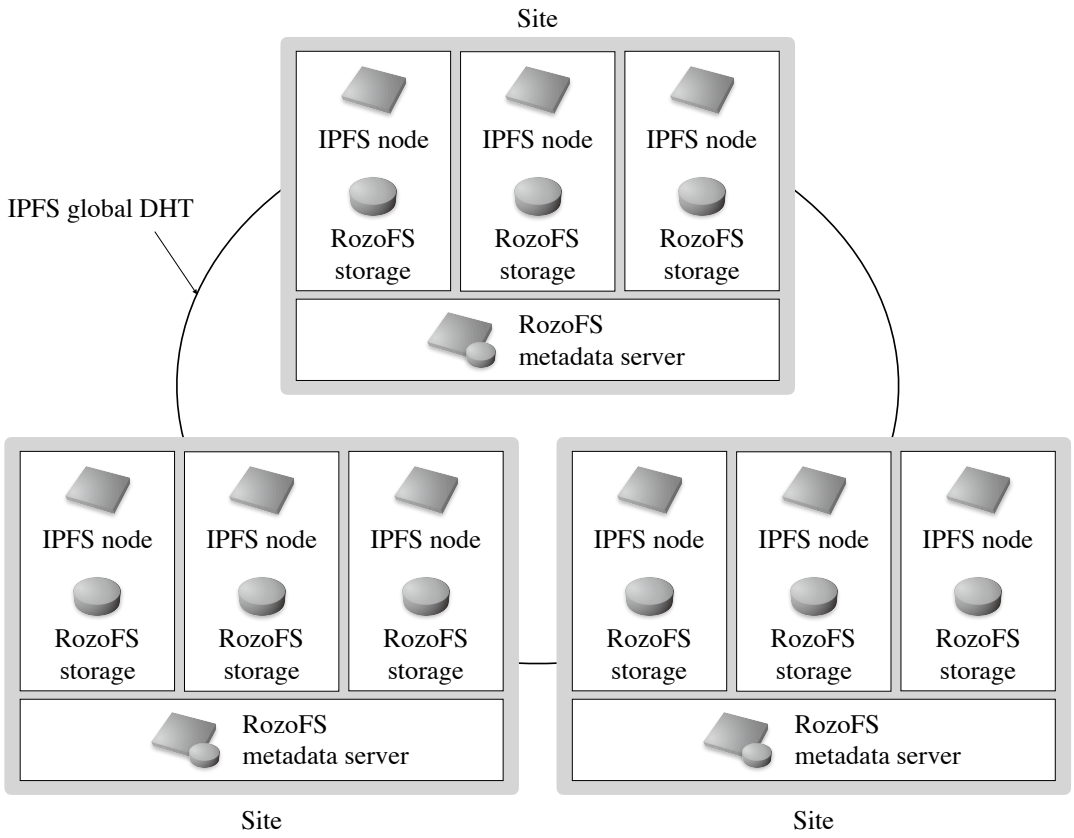
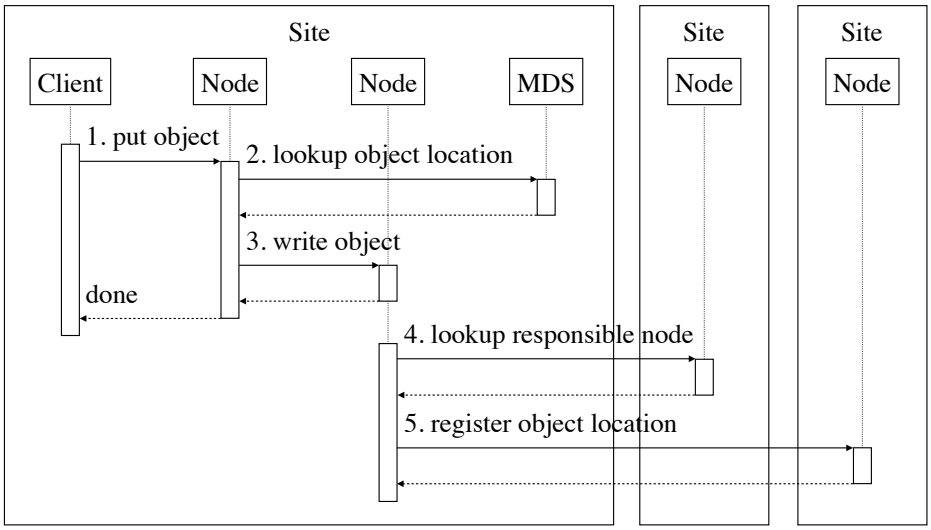


図 13　Confais らが提案しているオブジェクトストアの概略図

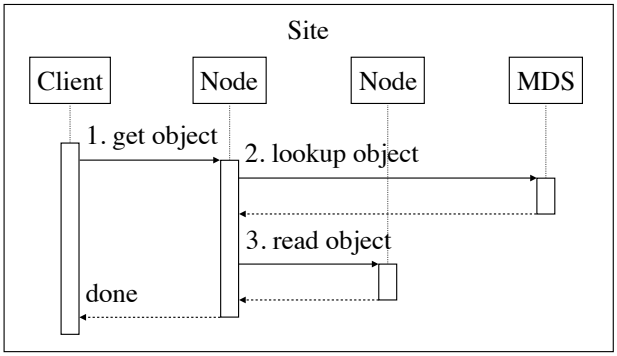
ドを一つ選択し，そのノードを介して RozoFS からオブジェクトを取得する．このとき，MDS によりオブジェクトが存在するノードの解決を行う．図 14 (c) は，リモートサイトのノードからオブジェクトの取得を行う場合のトラフィックの流れである．クライアントは，ローカルサイトからノードを一つ選択し，MDS を介して目的のオブジェクトがローカルサイトのノードに存在しないことを確認する．次に，クライアントにより選択されたノードは，目的のオブジェクトを格納しているノードの接続情報を IPFS から取得し，オブジェクトを格納しているノードに対してオブジェクトの取得のクエリを送信する．クエリを受信したノードは，図 14 (b) と同様の手順でオブジェクトを取得し，クライアントに結果を返す．取得したオブジェクトは，キャッシュとしてローカルサイトの RozoFS に保存される．

Confais らの手法は，比較的サイズの大きいオブジェクトそのものを DHT で管理せず，オブジェクトが格納されているノードの接続情報のみを DHT で管理する．また，クライアントが挿入したオブジェクトや一度アクセスしたリモートサイトのオブジェクトを同一サイトの SONAS にキャッシュしておくことで低遅延でのアクセスを可能にしている．しかしながら，DHT にノードの接続情報を登録したり取得したりする際に，遠方のサイトのノードとの通信が頻繁に発生し，オブジェクトの挿入やリモートオブジェクトの初回の取得の際に大きな遅延が生じるという問題点がある．

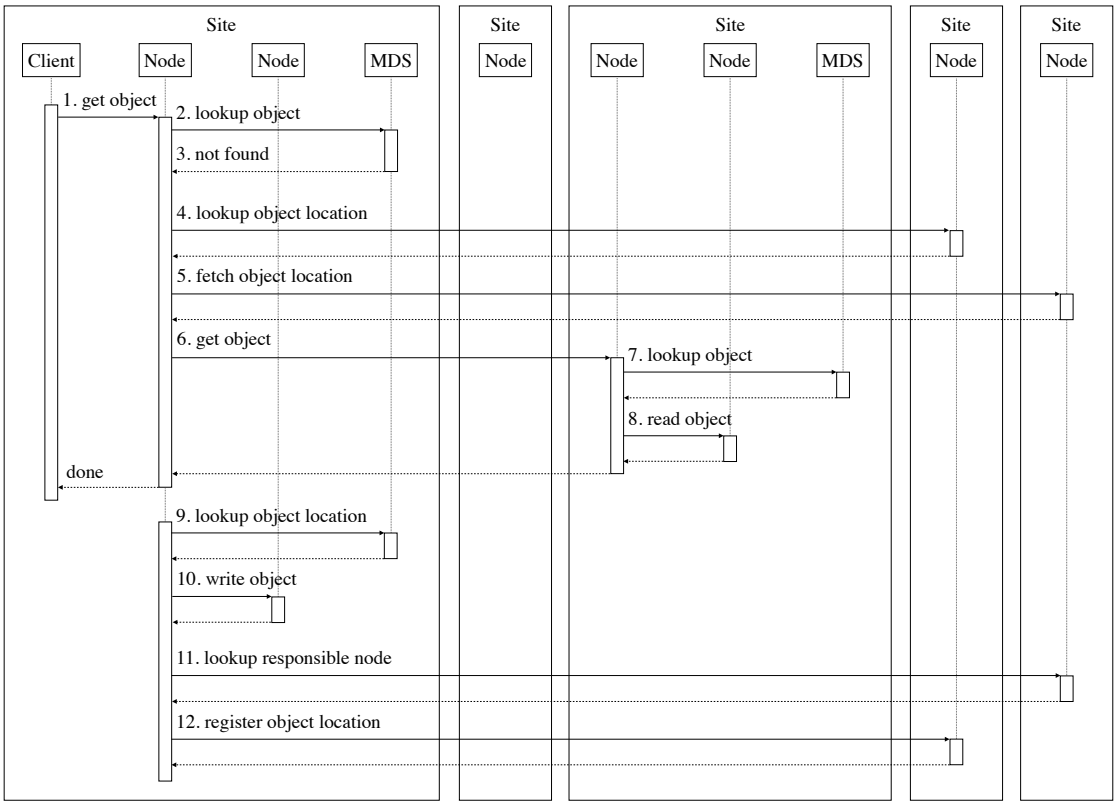
本論文で提案した近接性を考慮した分散ライングラフに基づき DHT を構成し，IPFS の代わりにこれを用いることで，(1) オブジェクトが格納されているノードの情報を登録する際 (図 14 (a) の手順 4 と手順 5)，(2) オブジェクトが格納されているノードの情報を取得する際 (図 14 (c) の手順 4 と手順 5，および，手順 11 と手順 12) に通信遅延の大きなノードと通信を行う頻度を軽減することができる．特に，図 14 (c) の手順 4 と手順 5 の遅延を削減することは，オブジェクトの取得にかかる時間を直接的に低減する．



(a) オブジェクトの格納



(b) ローカルサイトからのオブジェクトの取得



(c) リモートサイトからのオブジェクトの取得

図 14　Confais らが提案しているオブジェクトストアのシーケンス図

8. 考察

本節では，6 節で行なった評価と 7 節で議論した応用を踏まえて，提案手法の有用性について考察する．

8.1 正則グラフの性質の維持

5 節で述べたように，PDL グラフは DL グラフに基づく手法である．図 15 に DL グラフと PDL グラフの関係を示す．DL グラフに対して頂点融合や頂点分割を行なったものが PDL グラフ，DL^{+} グラフ [17]，$DL^{\sharp}$ [23] グラフである．PDL グラフは，DL グラフに対して頂点融合や頂点分割を行なって得られるすべての可能なトポロジの候補から隣接ノード間の通信遅延を最適化するものを選択する手法であり，DL グラフの性質を失うことなく構築される．そのため，PDL グラフは，DL グラフの持つ以下の性質を保持する．

- 任意の正則グラフを初期グラフとすることが可能
  - ループ付き完全有向グラフを初期グラフとすることで，de Bruijn グラフの性質を持つグラフを構成可能
  - 完全有向グラフを初期グラフとすることで，Kautz グラフの性質を持つグラフを構成可能
  - 現在発見されていない正則グラフであっても，それを PDL グラフの初期グラフとすることで近接性の考慮が可能
- 出次数が定数
  - 構造化オーバレイネットワークのトポロジのメンテナンスコストはネットワーク上のノード数と無関係
  - 経路表に保持する隣接ノードはネットワーク上のノード数と無関係

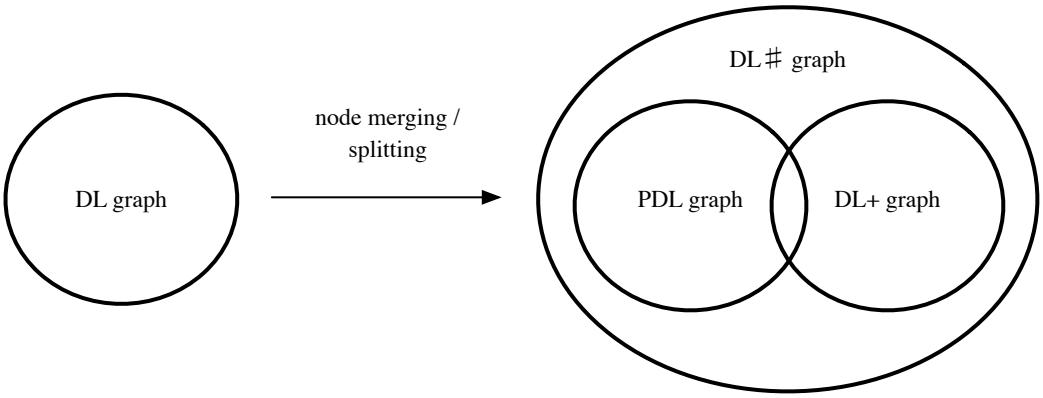


図 15　DL グラフと PDL グラフの関係

8.2 探索遅延の低減

6 節で述べたように，PDL グラフに基づく構造化オーバレイネットワークは近接性を考慮しないものと比較して，探索遅延を低減することができる．特に，2-正則グラフを初期グラフとする場合，オーバレイネットワークの探索遅延は常に 10% 程度削減可能であった．それ以外の場合でも，近接性を考慮しないものと比較していずれの場合も探索遅延を削減できていることから，PDL グラフの有用性が明らかである．

7 節では，PDL グラフが現実のシステムで有用であることを示した．他にも，構造化オーバレイネットワークを用いて構築されるシステムであれば，該当箇所を PDL グラフで置き換えることによって，探索遅延を低減することができる．このことからも，PDL グラフの有用性は明らかである．

8.3 今後の課題

PDL グラフで探索遅延を低減するためには，ノード間の近接性を正確に計測することが必要となる．PDL グラフでは，各ノードで計測される近接性メトリックに時不変性と対称性を前提条件として課したが，実際に近接性メトリックを計測する際は現実のネットワークの様々な制約を考慮する必要がある．例えば，2 つのノード間で通信を行う際，下位ネットワークのトラフィックは同一の経路を

通るとは限らない．また，通信の混雑により，計測される通信遅延は時間により変化する．このような状況下でノード間の近接性を正確に把握する手法は，過去に研究されている [22]．これらの手法が，PDL グラフの探索遅延に与える影響をより詳細に調査することが今後の課題として挙げられる．

また，PDL グラフでは，ノードの参加や離脱の際に，近接性を考慮してトポロジの拡張や縮小を行う．近接性メトリックの変化に応じて，ノードの参加や離脱以外に定期的に通信遅延の観点からトポロジの改善を行う仕組みを考える必要がある．これは，ある物理ノードに融合されている仮想ノードを，その兄弟ノードが融合されている他の物理ノードに移すことによって実現できると考えられる．

9. 結論

本論文では，定数次数かつ低遅延なオーバレイネットワークを構築する手法である近接性を考慮した分散ライングラフ (PDL グラフ) を提案した．トポロジの拡大や縮小の際に近接性を考慮する手法を提案した．PDL グラフは，トポロジの拡大や縮小の際にノード間の近接性に基づいて頂点融合や頂点分割を行う．シミュレーションにより，PDL グラフに基づくオーバレイネットワークが近接性を考慮しないものと比較して探索遅延を低減することを示した．また，PDL グラフをエッジコンピューティングにおけるオブジェクトストアに応用することを考え，IoT サービスを支えるストレージを実現するにあたって提案手法が重要な役割を果たすことを考察した．

今後の課題として，近接性の他にノードの信頼性や計算リソースなどのメトリックを考慮することでさらなる探索遅延の低減を目指す．また，複数のメトリックを組み合わせる手法についても検討を行う．

謝辞

本学の門林雄基教授，安本慶一教授，笠原正治教授，妙中雄三准教授には，研究について鋭いご指摘や的確なアドバイスをいただくとともに，本論文をまとめるにあたって熱心なご指導を賜りました．心より感謝申し上げます．

主指導教員である門林雄基教授には，分野における豊富な経験に基づく鋭いご指摘をいただきました．副指導教員である妙中雄三准教授には，工学とはなんたるかについて根気強く討論に付き合っていただきました．また，サイバーレジリエンス構成学研究室の樫原茂助教，Doudou Fall 助教には，研究の方向性や位置づけについて熱心にご指導いただくとともに，定期的に声をかけて励ましていただきました．先生方のご指導なくしては，この研究は成立しませんでした．深く感謝申し上げます．

学外の学生である私を温かく迎えてくださり，懇切丁寧に討論に付き合っていただいた湘南工科大学の眞田亜紀子講師に心から感謝申し上げます．

サイバーレジリエンス構成学研究室に在籍していた学生には，研究室内での活発な討論を通じて貴重な意見を頂きました．情報基盤システム学研究室の皆様には，論文執筆の際に大変お世話になりました．感謝申し上げます．

最後に，遠方から私を支援してくださった両親に心より感謝を申し上げます．

参考文献

[1] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *IEEE Transactions on Networking*, vol. 11, no. 1, pp. 17–32, 2003.

[2] P. Maymounkov and D. Mazières, "Kademlia: A peer-to-peer information system based on the xor metric," in *Peer-to-Peer Systems*, (Berlin, Heidelberg), pp. 53–65, Springer Berlin Heidelberg, 2002.

[3] A. Malatras, "State-of-the-art survey on P2P overlay networks in pervasive computing environments," *Journal of Network and Computer Applications*, vol. 55, pp. 1–23, 2015.

[4] M. F. Kaashoek and D. R. Karger, "Koorde: A simple degree-optimal distributed hash table," in *Peer-to-Peer Systems II* (M. F. Kaashoek and I. Stoica, eds.), (Berlin, Heidelberg), pp. 98–107, Springer Berlin Heidelberg, 2003.

[5] P. Fraigniaud and P. Gauron, "D2B: A de Bruijn based content-addressable network," *Theor. Comput. Sci.*, vol. 355, pp. 65–79, apr 2006.

[6] D. Li, X. Lu, and J. Wu, "FISSIONE: A scalable constant degree and low congestion DHT scheme based on Kautz graphs," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 3, pp. 1677–1688 vol. 3, mar 2005.

[7] Deke Guo, Jie Wu, Yunhao Liu, Hai Jin, Hanhua Chen, and Tao Chen, "Quasi-Kautz digraphs for peer-to-peer networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, pp. 1042–1055, jun 2011.

[8] D. Malkhi, M. Naor, and D. Ratajczak, "Viceroy: A scalable and dynamic emulation of the butterfly," in *Proceedings of the Twenty-first Annual Sym-*

posium on Principles of Distributed Computing, PODC '02, (New York, NY, USA), pp. 183–192, ACM, 2002.

[9] A. Kumar, S. Merugu, J. Xu, and X. Yu, "Ulysses: A robust, low-diameter, low-latency peer-to-peer network," in *11th IEEE International Conference on Network Protocols, 2003. Proceedings.*, pp. 258–267, nov 2003.

[10] D. Li, X. Lu, and J. Su, "Graph-theoretic analysis of Kautz topology and dht schemes," in *Network and Parallel Computing* (H. Jin, G. R. Gao, Z. Xu, and H. Chen, eds.), (Berlin, Heidelberg), pp. 308–315, Springer Berlin Heidelberg, 2004.

[11] H. Zhang, A. Goel, and R. Govindan, "Incrementally improving lookup latency in distributed hash table systems," *SIGMETRICS Perform. Eval. Rev.*, vol. 31, pp. 114–125, jun 2003.

[12] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica, "The impact of DHT routing geometry on resilience and proximity," in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, (New York, NY, USA), pp. 381–394, ACM, 2003.

[13] H. Shen and C.-Z. Xu, "Hash-based proximity clustering for efficient load balancing in heterogeneous DHT networks," *Journal of Parallel and Distributed Computing*, vol. 68, no. 5, pp. 686–702, 2008.

[14] T. Miyao, H. Nagao, and K. Shudo, "A method for designing proximity-aware routing algorithms for structured overlays," in *2013 IEEE Symposium on Computers and Communications (ISCC)*, pp. 508–514, jul 2013.

[15] J. Song, S. Park, and J. Yang, "An adaptive proximity route selection scheme in DHT-based peer to peer systems," in *Parallel and Distributed Computing: Applications and Technologies* (K.-M. Liew, H. Shen, S. See, W. Cai, P. Fan,

and S. Horiguchi, eds.), (Berlin, Heidelberg), pp. 778–781, Springer Berlin Heidelberg, 2005.

[16] K. Park, S. Pack, and T. Kwon, "Proximity based peer-to-peer overlay networks (P3ON) with load distribution," in *Information Networking. Towards Ubiquitous Networking and Services* (T. Vazão, M. M. Freire, and I. Chong, eds.), (Berlin, Heidelberg), pp. 234–243, Springer Berlin Heidelberg, 2008.

[17] Y. Zhang and L. Liu, "Distributed line graphs: A universal technique for designing DHTs based on arbitrary regular graphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, pp. 1556–1569, sep 2012.

[18] M. Miller and J. VSirávn, "Moore graphs and beyond: A survey of the degree/diameter problem," *Electronic Journal of Combinatorics, Dynamic survey*, vol. 14, pp. 1–61, 2013.

[19] M. Imase and M. Itoh, "Design to minimize diameter on building-block network," *IEEE Transactions on Computers*, vol. C-30, pp. 439–442, jun 1981.

[20] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, "Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web," in *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, STOC '97, (New York, NY, USA), pp. 654–663, ACM, 1997.

[21] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content-addressable Network," *SIGCOMM Comput. Commun. Rev.*, vol. 31, pp. 161–172, aug 2001.

[22] T. S. E. Ng, Y. . Chu, S. G. Rao, K. Sripanidkulchai, and H. Zhang, "Measurement-based optimization techniques for bandwidth-demanding peer-to-peer systems," in *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, vol. 3, pp. 2199–2209 vol.3, mar 2003.

[23] A. Manada and H. Morita, "Graph theoretical analysis on distributed line graphs for peer-to-peer networks," *IEEE Transactions on Network Science and Engineering*, pp. 1–1, 2018.

[24] Y. Zhang, X. Lu, and D. Li, "SKY: efficient peer-to-peer networks based on distributed Kautz graphs," *Science in China Series F: Information Sciences*, vol. 52, pp. 588–601, apr 2009.

[25] L. Gui, R. Shen, Y. Yu, D. Feng, Y. Peng, and W. Liu, "DLG-Hypertree: A low-diameter, server-centric datacenter network architecture," in *2015 IEEE 8th International Conference on Cloud Computing*, pp. 1077–1080, June 2015.

[26] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proceedings of IEEE INFOCOM '96. Conference on Computer Communications*, vol. 2, pp. 594–602 vol.2, mar 1996.

[27] B. Confais, A. Lebre, and B. Parrein, "An object store service for a fog/edge computing infrastructure based on IPFS and a scale-out nas," in *2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, pp. 41–50, 2017.

[28] B. Confais, A. Lebre, and B. Parrein, "Performance analysis of object store systems in a fog and edge computing infrastructure," in *Transactions on Large-Scale Data- and Knowledge-Centered Systems XXXIII* (A. Hameurlain, J. Küng, R. Wagner, R. Akbarinia, and E. Pacitti, eds.), pp. 40–79, Berlin, Heidelberg: Springer Berlin Heidelberg, 2017.

付録

A. 発表リスト

国際会議等

1. Youki Shiraishi, Akiko Manada, Yuzo Taenaka, and Youki Kadobayashi, "A Structured Overlay Network Based on a Proximity-aware Distributed Line Graph," in *Proc. 2019 IEEE 4th International Conference on Computer and Communication Systems*, pp. 430–435, February 2019.

2. Youki Shiraishi and Michiko Harayama, "A Load Balancing Method for Range-Queryable Structured Overlays," *9th EAI MobiCASE Student Workshop 2018*, February 2018.

研究会等

3. 白石 裕輝, 眞田 亜紀子, 妙中 雄三, 門林 雄基, "下位ネットワークの近接性を考慮した分散ラインググラフに基づく構造化オーバレイネットワーク," 信学技報, Vol. 118, No. 360, IA2018-51, pp. 71–77, 2018 年 12 月. (学生研究奨励賞)

4. 白石 裕輝, 樫原 茂, Doudou Fall, "エッジコンピューティング基盤のための局所性を考慮したオブジェクトストアの提案," 研究報告マルチメディア通信と分散処理 *(DPS)*, Vol. 2018-DPS-174, No. 29, pp. 1–6, 2018 年 2 月.

5. 白石 裕輝, "範囲探索可能な構造化オーバレイにおける負荷分散手法," 電子情報通信学会 関西支部 第 *23* 回学生会研究発表講演会, 2018 年 3 月.

口頭発表

6. 白石 裕輝, “P2P ネットワークトポロジとしての de Bruijn 族グラフ,” 第10回広域センサーネットワークとオーバレイネットワークに関するワークショップ, 2018年9月.

7. 白石 裕輝, “セキュアな P2P システムの実現に向けて,” 第9回広域センサーネットワークとオーバレイネットワークに関するワークショップ, 2017年6月.