大規模DDSネットワークにおける起動時間短縮のための改善手法の提案

富重 亮佑[1]　樽谷 優弥[2]　高野 祐輝[3]　三浦 秀芳[1]　横平 徳美[4]

概要：ROS（Robot Operating System）の次世代バージョンである ROS 2 を用いた自動運転システムなどの大規模な分散システムは，多数のノードから構成され，ノード間の Publish/Subscribe 通信に Data Distribution Service（DDS）を利用している．DDS では，通信開始前に通信相手や通信条件などの情報を交換する Discovery と呼ばれる手続きを行う．しかし，DDS 標準の Discovery では各ノードが分散的に情報を交換するため，大規模なシステムでノードやエンドポイントの数が増加すると通信量が増大し，Discovery の完了とシステムの起動に長い時間を要するという問題がある．本研究では，この問題を解決するため，リーダー選出を用いた Centralized Discovery Protocol（CDP）を提案する．提案手法では，選出されたリーダーを中心に Discovery 情報を集約・配布することで，Discovery に伴うメッセージ数を削減する．また，リーダー選出機構を導入することで，特定のノードを単一障害点とすることなくリーダーの交代後も情報交換を継続可能な構成とする．提案手法を実装し，DDS 標準の Discovery と比較評価した結果，従来手法では Discovery の完了に 6 分以上を要する大規模な DDS ネットワークにおいて，DDS アプリケーションの起動時間を約 86%短縮できることを確認した．

A Proposal for Reducing Startup Time in Large-Scale DDS Networks

Ryosuke Tomishige[1]　Yuya Tarutani[2]　Yuuki Takano[3]　Hideyoshi Miura[1]　Tokumi Yokohira[4]

Abstract: Large-scale distributed systems, such as autonomous driving systems based on ROS 2 (the next-generation version of the Robot Operating System), consist of numerous nodes and use the Data Distribution Service (DDS) for inter-node Publish/Subscribe communication. Before starting communication, DDS performs a procedure called Discovery to exchange information about communication peers and parameters. However, because standard DDS Discovery exchanges information in a distributed manner among all nodes, increasing the number of nodes and endpoints in a large-scale system causes a surge in communication traffic, resulting in long Discovery completion and system startup times. To address this problem, we propose a Centralized Discovery Protocol (CDP) using leader election. The proposed method reduces the number of Discovery messages by aggregating and distributing Discovery information through an elected leader. Furthermore, incorporating a leader election mechanism avoids making any specific node a single point of failure and allows information exchange to continue after a leader change. We implemented the proposed method and compared it with standard DDS Discovery. Experimental results show that in a large-scale DDS network where the conventional method requires more than six minutes to complete Discovery, the proposed method reduces the startup time of DDS applications by approximately 86%.

1. はじめに

ROS 2 を用いた自動運転システムなどの大規模な分散システムは，1 台または複数台の計算機（以下，ホス

1 岡山大学
Okayama University
2 大阪大学
The University of Osaka
3 株式会社ティアフォー
TIER IV, Inc.
4 岡山理科大学
Okayama University of Science

ⓒ2026 Information Processing Society of Japan

ト）上で動作する多数のノードから構成され，ノード間の Publish/Subscribe（Pub/Sub）通信に Data Distribution Service（DDS）[1] を利用している．DDS は Object Management Group（OMG）が標準化を行っているエンドツーエンドの Pub/Sub ミドルウェアである．各ノードは，データを出版する DataWriter やデータを購読する DataReader といったエンドポイントを複数持つ．Message Queuing Telemetry Transport（MQTT）などの Pub/Sub ミドルウェアと異なり，DDS はブローカーを介さずエンドポイント同士が直接エンドツーエンドに通信を行うため，通信開始前に各ノードが他のノードの位置（IP アドレスとポート番号のペア）やどの Topic のエンドポイントを持っているかを把握しておく必要がある．この手続きを DDS では Discovery と呼ぶ．

自動運転システム Autoware[2] のような多数のノードから構成されるシステムでは，Discovery にかかる時間がボトルネックとなり，システムの起動完了までに時間を要するという問題がある．これは DDS 標準の Discovery では各ノードが分散的に情報を交換するため，ノード数やエンドポイント数の増加に伴って通信量が急増し，Discovery の完了が遅延するからである．

本研究では，この問題を解決するため，リーダー選出を用いた Centralized Discovery Protocol（CDP）を提案する．提案手法では，各ホスト上で動作するノードの中からホストごとに 1 つのリーダーを選出し，リーダーを介して同一ホスト内のエンドポイント情報を集約・配布することで，Discovery に伴うメッセージ数を削減する．また，リーダー選出機構を導入することで，特定のノードを単一障害点とすることなくリーダーの交代後も情報交換を継続可能な構成とする．提案手法を実装した DDS と標準の Discovery を用いた DDS の比較評価を通じて Discovery 完了時間の短縮効果を検証する．

2. DDS と RTPS

DDS は Pub/Sub ミドルウェアの API 仕様を定めた標準規格であり，その Pub/Sub 通信を実現するための Wire プロトコルとして RTPS[3] が規定されている．これらに準拠した DDS 実装は複数存在し，ROS 2 のデフォルト実装である eProsima's Fast DDS[4] や，Autoware で採用されている Eclipse Cyclone DDS[5]，実験的な Rust 製の実装である TIER IV の Umber DDS[6] などがある．

DDS の Pub/Sub 通信では，通信が成立する論理的なネットワーク範囲を DDS Domain（以下，ドメイン）として分離しており，同一ドメインに属する各ノード（DDS における DomainParticipant）の間でのみ Discovery とデータの送受信が行われる．各ノードは，どのデータを送受信するかに基づいてデータをやりとりする．やりとりするデータの対象は，名前とデータ型のペアである Topic として表現される．各ノードは，ある Topic のデータを出版するために DataWriter を作成し，ある Topic のデータを購読するために DataReader を作成する．これら DataWriter と DataReader は総称してエンドポイントと呼ばれる．また，DDS では Quality of Service（QoS）により，データの履歴管理や生存期間，通信の信頼性といった通信の挙動をエンドポイントごとに制御できる．

通信を行うにあたり，DataWriter は同一 Topic の DataReader の位置や QoS を，DataReader は同一 Topic の DataWriter の位置や QoS を Discovery Protocol によって把握する．同一 Topic の DataWriter と DataReader は，Discovery で相手の情報を取得し，QoS の互換性などの通信条件を満たすとマッチする [3]．各エンドポイントがマッチした相手エンドポイントの個数をマッチ数と呼ぶ．DataWriter のマッチ数はマッチした DataReader の数であり，DataReader のマッチ数はマッチした DataWriter の数である．

QoS のうち，通信の信頼性を定めるポリシーが Reliability QoS である．Reliability QoS には Best-Effort と Reliable の 2 種類があり，Best-Effort はデータの到達を保証しない設定であるのに対し，Reliable はシーケンス番号と確認応答を用いてデータの到達を保証する設定である．Reliable におけるデータの到達確認と再送制御は，トランスポート層に UDP を用いる場合も TCP を用いる場合も，RTPS 層で行われる．RTPS では，1 つの RTPS メッセージがヘッダと複数のサブメッセージ（Submessage）から構成される．DataWriter が送信する Topic のデータは DATA サブメッセージとして格納され，各 DATA サブメッセージには TCP と同様にシーケンス番号が付与される．DataWriter は送信済みのシーケンス番号の範囲を HEARTBEAT サブメッセージで定期的に通知し，DataReader はその範囲と受信済みのシーケンス番号を比較した結果を ACKNACK サブメッセージで返信する．未達の DATA サブメッセージがある場合，DataWriter が該当データを再送することで信頼性を確保する．

2.1 Discovery Protocol

RTPS 標準の Discovery Protocol として，Simple Discovery Protocol（SDP）が規定されている．SDP は，ノードを発見する Simple Participant Discovery Protocol（SPDP）と，エンドポイントを発見する Simple Endpoint Discovery Protocol（SEDP）の 2 段階で構成される．各ノードはまず，SPDP によって自ノードの情報を含む SPDP メッセージを定期的にネットワークへ送信する．他ノードからの SPDP メッセージを受信してその存在を発見すると，続いて自ノードが持つ DataWriter/DataReader の情報を含む SEDP メッセージをそのノードへ送信する．これらの SPDP メッセージおよび SEDP メッセージを送信する

ⓒ2026 Information Processing Society of Japan

手段は，通常のユーザデータを送信する手段と同じであり，あらかじめ定義された特別な Topic（Built-in Topic）の DataWriter/DataReader を用いて行われる．このうち SEDP の送信は，エンドポイント情報の到達を保証するため，Reliability QoS を Reliable として行われる．

Discovery Protocol は，初回の発見だけでなく，発見済みノードとエンドポイントの存在情報の管理にも用いられる．SPDP はノードの生存確認を担う．各ノードは生存確認の有効期間である lease duration を通知する．受信側は SPDP の受信により有効期限を更新し，更新されないまま期限を過ぎると，相手ノードが存在しなくなったと判定する [3]．本稿では，この判定によって発見済みノードの情報が削除されることをノードの喪失と呼ぶ．これは実際のプロセス停止だけでなく，通信の損失や受信処理の遅延によっても発生し得る．

エンドポイントの作成・更新・削除は SEDP で通知される．相手エンドポイントの削除を検出した場合や，そのエンドポイントが属するノードの喪失を検出した場合には，受信側は該当するエンドポイント情報を削除し，対応するマッチを解除する [3]．

このような手順をとる SDP は小規模から中規模のネットワークを対象としており，大規模なネットワークでの使用は想定されていない [3]．SDP では全ノードが分散的に情報を交換する設計となっており，特に SEDP では各ノードが発見した他のすべてのノードに対して自ノードの全エンドポイント情報を個別に送信し，パケットドロップ発生時には確認応答と再送を繰り返す．そのため，総ノード数を P，各ノードの平均エンドポイント数を E，各ノードでの平均パケットドロップ率を D とすると，SEDP によるエンドポイント発見のメッセージ複雑度は $O(P^2ED^2)$ と表され，ノード数やエンドポイント数の増加に伴ってメッセージ数が急激に増大する [7]．実際，先行研究 [7] では，2 台のホスト上で同一 Topic の DataWriter と DataReader を 1 つずつ持つノードを合計 204 個（計 408 エンドポイント）同時に立ち上げる評価実験を Fast DDS[4] で行い，SDP のスケーラビリティの問題を検証している．この実験では，SEDP によるエンドポイント発見の段階で大量のメッセージが集中することで UDP ソケットバッファにおけるパケットドロップが多発し，再送処理が繰り返される結果，全エンドポイントの発見完了までに約 50 s を要することが確認されている．

2.2 Discovery の高速化に関する先行研究

DDS の Discovery を高速化する先行研究として，Park ら [8] は，SPDP によるノード情報の周期的な交換が大規模システムで負荷を増大させる点に着目した．同研究では，起動時の送信をランダムな待ち時間によって分散する方式と，ノードの参加・離脱の状態に応じて SPDP の送信周期を調整する方式を提案している．これは，エンドポイント情報を交換する SEDP よりも，ノード情報を交換する SPDP を改善対象とするアプローチである．

一方，先行研究 [7] は，SPDP によるノード発見ではなく，SEDP によるエンドポイント発見段階を発見遅延の主な要因と分析している．同研究では，エンドポイントの新規発見が進行する時間帯と UDP ソケットでのパケットドロップが多発する時間帯が一致することから，SEDP のメッセージ数と再送に着目して遅延を説明している．

実際，SPDP は Discovery 完了後も継続するため，その周期送信の負荷だけで処理能力を超過するのであれば，完了後にもノードの喪失やマッチ数の減少などが継続すると予想される．しかし，本研究の実験では，Discovery 完了後に SPDP に起因すると推測される不安定化は観測されず，SPDP の送信間隔を変更した検証でも，Park らの研究で示されたほどの改善は確認できなかった．なお，この検証は Park らの方式全体の再現評価ではなく，起動時に SEDP と競合する SPDP の影響自体を否定するものでもない．しかし，起動時に集中する SEDP のエンドポイント情報交換と再送が完了することで負荷が減るという点では先行研究 [7] の分析と整合するため，本研究では SPDP の送信周期の調整よりも，SEDP のエンドポイント情報交換と欠落修復を主な改善対象とする．

Discovery 情報を集約する既存手法として，Fast DDS の Discovery Server がある [9]．本方式では，サーバーとして動作するノードがクライアントから Discovery 情報を受け取り，他のクライアントへ配布する．Discovery にマルチキャストを必要とせず，トランスポートとして UDP だけでなく TCP も選択できるため，マルチキャストを利用できないネットワークにも適用できる．

一方，サーバーと接続先を明示的に設定する構成では，サーバーとして動作するノードを事前に決め，各クライアントの起動時にその接続先アドレスとポート番号などを指定する必要がある．このため，サーバーを別ホストへ移設した場合や接続先アドレスが変わる場合には，クライアント側の接続先設定も管理・更新する必要があり，構成変更に伴う運用負担が生じる．なお，Fast DDS にはホストごとのサーバーを自動起動する Easy Mode もあるが，別ホストのサーバーへ接続するための IP アドレス指定は残る [10]．本研究では，マルチキャストが利用可能なネットワークを対象に，情報集約を担当するノードの事前指定と，その接続先設定の配布を不要にすることを目指す．

また，Discovery Server を利用するノードと，標準 SDP を利用するノードの混在にも制約がある．Discovery Server に対応し，クライアントとして有効化したノードは，標準 SDP のまま動作するノードとは，通常の構成では相互発見・マッチしない [11]．したがって，機能への対応だけでは相互運用は成立せず，標準 SDP のみを利用するノード

ⓒ2026 Information Processing Society of Japan

を変更せずに段階的に導入する場合には，両方式の間で相互運用を実現する仕組みが別途必要となる．

3. 提案手法

提案する Centralized Discovery Protocol（CDP）は，ホストごとに選出したリーダーへ同一ホスト内のエンドポイント情報を集約し，リーダーがその情報を同一ドメイン内のノードへマルチキャストで配布する Discovery プロトコルである．本研究では，同一ドメイン内のすべてのノードが CDP に対応することを前提とする．受信者ごとに同じエンドポイント情報を個別に送信する処理を減らす一方，リーダーの交代後も情報交換を継続できる構成とする．集約対象は Discovery 情報だけであり，ユーザデータは元の DataWriter と DataReader の間で直接送受信する．

3.1 ホスト単位の情報集約

CDP では，同一ホストのノードから情報の集約と配布を担当するリーダーを 1 個選出し，それ以外をフォロワーとする．各ノードはホスト名の前後の空白を除き，小文字に正規化した文字列の 128 bit ハッシュ値をホスト識別子として用いる．ノード自身の識別には RTPS の GUID prefix を用いるため，起動前にノード数や各ノードの識別子一覧を設定する必要はない．ただし，正規化後のホスト名がホスト間で一意であり，ホスト識別子が衝突しないことを前提とする．

リーダー選出やエンドポイント情報の通知・配布など，CDP の制御メッセージの送信には，同一ドメインの全ノードが参加する専用の UDP マルチキャストグループを用いる．フォロワーは自分に属するエンドポイントの情報を同一ホストのリーダーへ通知する．リーダーはその情報を受理すると，同一ドメイン内の他のノードに送信する．受信側は送信元のホスト識別子，リーダーの GUID prefix，世代情報を確認して，当該ホストのリーダーによる配布かどうかを判定する．フォロワーによるリーダー宛ての更新は同一ホストのリーダーだけが処理するが，リーダーによる配布は同一ホストのフォロワーに加え，他ホストのリーダーとフォロワーも受信して登録する．したがって，現実装では他ホストのフォロワーへ配布するために，受信側ホストのリーダーが同じ情報を再送する必要はない．

なお，CDP が変更するのはエンドポイント情報の配布方法のみであり，SPDP によるノードの発見・生存管理や，各ノードにおける通信相手のエンドポイント情報の保持は従来どおり行う．

3.2 リーダー選出と世代管理

各ノードは，自身の GUID prefix，ホスト識別子，役割，認識しているリーダーとその世代を含む通知を定期的に送信する．受信側は同一ホストの送信元ごとに最終受信時刻を保持し，所定の有効期間内にこの通知やその他の制御メッセージを受信したノードのうち，GUID prefix が最大のものをリーダー候補とする．自身が候補となった場合はリーダーとして動作し，通常の通知に加えてリーダーの生存通知を送信する．現実装の既定値では通知周期を 1 s，選出に用いる有効期間を 15 s とする．この有効期間はリーダー選出用のものであり，SPDP におけるノードの lease duration とは別に管理する．

リーダーの世代は，カウンタとリーダー候補の GUID prefix の対である term によって識別する．新たにリーダーとなるノードは，観測済み term のカウンタを 1 増やし，自身の GUID prefix と組み合わせて新しい term とする．term はカウンタを優先し，同値なら GUID prefix で比較する．同一ホストからより大きな term を受信したノードは世代情報を更新し，他者がリーダーであればフォロワーとして動作する．他ホストについてもホストごとにリーダーと term を記録し，その記録が有効な間は，より古い term による配布を受理しない．一方，フォロワーが同一ホストのリーダーへ送る自身のエンドポイント情報の更新は，選出の途中で作成されたことだけを理由に破棄しないよう，古い term でもリーダーが受理できるようにする．

起動直後や通信が遅延した区間では，複数のノードが一時的にリーダーとして動作する可能性がある．同一ホスト内で参加集合の認識が一致し，通知が継続的に到達する状態になれば，最大の GUID prefix を持つノードへ選出結果が収束する．ただし，本方式は多数決による合意を行わないため，ネットワーク分断中もリーダーが常に一意であることや，所定の時間内に交代が完了することは保証しない．

3.3 エンドポイント情報の配布と登録

CDP で配布する情報の単位をエンドポイントレコードとする．各レコードは，エンドポイントが属するノードの識別子，そのノードが動作するホストの識別子，エンドポイントの GUID，更新番号である revision，DataWriter または DataReader の種別，操作種別，および SEDP と同じパラメータリスト形式で直列化したエンドポイント定義を含む．エンドポイント定義には Topic 名，型名，QoS，送受信先のアドレスとポート番号を格納する．各ノードは，自身のエンドポイント情報から配布用のレコードを作成する際に revision を付与し，受信側はエンドポイントの GUID と revision を用いて新しいレコードと重複を区別する．送信時には複数のレコードをまとめ，レコードの境界で UDP データグラムへ分割する．

受信側ノードは，発見した相手エンドポイントの情報を保持する既知エンドポイント一覧を持つ．レコードに含まれるエンドポイント情報は，そのエンドポイントが属するノードの識別子と関連付けて，受信側ノード自身の既知エンドポイント一覧に登録する．エンドポイントが属する

ⓒ2026 Information Processing Society of Japan

ノードを SPDP でまだ発見していない場合はレコードを保留し，SPDP でそのノードを発見した後に，保留したエンドポイント情報の同一覧への登録を再試行する．レコードの受信状態と既知エンドポイント一覧への登録状態を別々に管理することで，受信済みでも同一覧に未登録の情報を，同一 revision のレコードの再受信時に登録できるようにする．リーダー自身も，受信済みだが同一覧に未登録のエンドポイント情報について定期的に登録を再試行する．

受信側ノードは，自身のエンドポイントを作成する前に受信した相手エンドポイントの情報も，既知エンドポイント一覧に保持する．自身の DataReader または DataWriter を作成した際には，同一覧の相手エンドポイントの情報を用いて，Topic 名，型名，Topic 種別，QoS の適合性を確認する．これにより，新たな情報配布を待たずに，通信条件が適合する相手エンドポイントとマッチできる．ノードの生存は引き続き SPDP で判定し，その期限切れによって既知エンドポイント一覧から相手エンドポイントの情報を削除した場合には，CDP 側の受信済み・登録済み状態も無効化する．再び SPDP で発見した後は，同一 revision のレコードでも復元対象とする．

3.4 欠落情報の修復とリーダー交代

UDP で失われた更新を修復するため，CDP はエンドポイントが属するノードごとに情報を照合する．エンドポイントレコード群をまとめて配布するメッセージを Snapshot と呼ぶ．照合のために，ノードごとのエンドポイントレコード件数と，レコードごとのハッシュ値を排他的論理和で集約したダイジェストを管理する．リーダーは既定で 1 s ごとに，同一ホストの各ノードへ，そのノードに属するエンドポイントのレコード件数とダイジェストを問い合わせる．リーダーが保持する値と異なる場合は，該当するノードを指定して Snapshot を要求する．逆に，リーダーはノードごとの件数とダイジェストの一覧を配布する．フォロワーは自身が登録したエンドポイント情報と比較し，件数またはダイジェストが一致しないノードをリーダーへ報告する．リーダーが報告されたノードに属するエンドポイント情報の Snapshot を再配布することで，同じ情報を欠く複数のフォロワーを一度の配布で修復する．

リーダーが持たない情報をフォロワーが保持している場合も修復対象とする．フォロワーは保持しているエンドポイントレコードの件数とダイジェストを，エンドポイントが属するノードごとにリーダーへ提示する．リーダーは修復対象のエンドポイントが属するノードごとに，提供元のフォロワーを 1 個だけ指定して Snapshot を取得する．応答が得られず修復が進まない場合は，別の提供元へ順に切り替える．同じノードに属するエンドポイントについての Snapshot 要求や修復配布，およびフォロワーによる保持情報の提示と Snapshot 応答には既定で 5 s の間隔制限を設け，複数フォロワーが同時に全レコードを送り返すことを抑える．

定期的な全体照合では，リーダーは既定で 30 s ごとに同一ホスト内のノードから，そのノードに属するエンドポイント情報の Snapshot を取得し，同一ドメインへ再配布する．全ノードから一斉に応答させず，GUID prefix から同一ホストのノードを 4 群に分け，周期ごとに 2 群を対象とし，対象群を順に切り替える．この周期的な配布によって，他ホスト側で欠落した情報にも再取得の機会を設ける．照合と配布は損失時にも繰り返すが，ダイジェストが一致していればホスト内の個別修復のために全エンドポイント情報を送り直す必要はない．

リーダーが離脱すると，最終受信時刻が有効期間を過ぎた時点で選出対象から外れ，残るノードから新しいリーダーを選出する．新リーダーは自身が保持していたエンドポイント情報を利用し，Snapshot の取得とダイジェストの照合によってドメイン全体の情報を再同期する．再同期は全情報の一斉再送ではなく，群ごとの取得とエンドポイントが属するノードごとの欠落修復を組み合わせて進める．リーダーの交代だけを理由に，既にマッチしたユーザデータの通信経路は変更しない．この回復には，残ったノード間で制御メッセージが到達することが必要である．

4. 提案手法の評価

4.1 評価環境

提案手法による Discovery 完了時間の短縮効果を調べるため，提案手法である CDP を実装した Umber DDS と，標準の SDP を用いる Cyclone DDS 0.10.5 を比較する．各試行ではドメイン内の全ノードに同じ Discovery 方式を用い，CDP と標準 SDP を混在させない．評価プログラムは，先行研究 [7] で Discovery トラヒックを測定するために用いたプログラムを拡張したものである．先行研究のプログラムは各ノードに 1 個ずつの DataWriter と DataReader を生成するのに対し，本評価のプログラムは複数の Topic に対して DataWriter と DataReader を生成し，エンドポイントのマッチ後にユーザデータを送受信する．

表 1 に評価用ホストの構成を示す．同一構成のホスト 2 台を Ethernet で `1Gbps` で接続して用いる．各ノードを独立したプロセスとして実行し，ホスト 1 台につき 102 プロセス，合計 204 プロセスを起動する．

表 2 に各試行の負荷条件を示す．各ノードは 15 個の Topic について DataWriter と DataReader を 1 個ずつ生成するため，1 ノード当たりのエンドポイント数は 30，システム全体のエンドポイント数は 6,120 となる．各 DataWriter は，同じ Topic の DataReader を 1 個以上発見した後，8,192 byte のユーザデータを 500 ms 間隔で送信する．このユーザデータサイズは，画像や LiDAR の点群データといった大きなデータを想定して MTU（1,500 byte）より

表 1 評価用ホストの構成
Table 1 Specifications of the evaluation hosts.

項目	構成
台数	2 台
CPU	Intel Core i5-8500（3.00 GHz，6 コア）
LLC 容量	9,216 KB
メモリ容量	7.5 GiB
OS カーネル	Linux 6.8.0-138-generic
NIC	Intel Ethernet Connection（7）I219-LM
MTU	1,500 byte

表 2 評価時のノード，エンドポイントおよびユーザデータの条件
Table 2 Workload parameters for nodes, endpoints, and user data.

項目	設定値
ノード数	102/host（計 204）
プロセス当たりのノード数	1
Topic 数	15
DataWriter 数	15/ノード
DataReader 数	15/ノード
ユーザデータサイズ	8,192 byte
ユーザデータ送信間隔	500 ms
ノードの lease duration	40 s
SPDP 送信間隔	3 s

大きく，かつ大きなデータを RTPS 層で DATA_FRAG サブメッセージに分割する機能が未実装である Umber DDS でも扱えるよう，RTPS メッセージの最大サイズ未満の値として選択した．したがって，ユーザデータは両実装ともに 1 つの DATA サブメッセージとして送信され，IP 層でフラグメント化される．ユーザデータは，両実装ともマルチキャストで配送される．ノードの lease duration は予備実験において両実装が安定して動作することを確認できた 40 s，SPDP の送信間隔は Umber DDS のデフォルト値である 3 s に統一する．これらの条件により，Discovery トラヒックだけでなく，エンドポイントの発見に伴って増加するユーザトラヒックが存在する状況で両方式を比較する．

ユーザデータの QoS には，両実装の既定値を用いる．Reliability QoS は DataWriter が Reliable，DataReader が Best-Effort であり，Durability QoS は Volatile，History QoS は KeepLast（depth: 1）である [12]．したがって，本評価のユーザデータ通信では受信側が Reliable による到達保証を要求しない．

UDP ソケットのバッファについて，**表 3** に実装から確認した設定を示す．Umber DDS は，通常の RTPS 用の受信ソケットごとに 4 MiB，CDP 専用の受信ソケットごとに 64 MiB を要求し，送信バッファのサイズは明示的に変更しない．Cyclone DDS の設定ファイルでは，バッファサイズを上書きせず，0.10.5 の既定設定を用いる．受信側は 1 MiB への拡大を試み，要求を満たせなくても確保できたサイズで処理を継続する．送信側は最低 64 KiB を要求し，OS が初期設定したサイズがこれを満たす場合は変更しない [12]．

表 3 UDP ソケットバッファの設定 (ソケット当たりの要求値)
Table 3 UDP socket buffer settings (requested size per socket).

対象	受信	送信
Umber DDS（RTPS）	4 MiB	OS 既定
Umber DDS（CDP）	64 MiB	OS 既定
Cyclone DDS（既定）	1 MiB	最低 64 KiB

ただし，両ホストでは全試行を通じて，受信バッファの要求上限を定める `net.core.rmem_max` は 212,992 byte（208 KiB）であった．表の受信サイズはいずれもこの上限を超えるため，通常の `SO_RCVBUF` による拡大要求は 208 KiB に制限される．したがって，表に示す 64 MiB を CDP 専用ソケットで確保していたわけではない．

Linux では，設定時に管理領域を考慮して受信バッファの値を倍増し，`getsockopt` はその倍増後の値を返す [13]．上記の上限で設定した場合，取得値は 425,984 byte（416 KiB）となる計算である．これは設定と OS の仕様から算出した値であり，実験時の各ソケットで取得した実測値ではない．また，管理領域を含むため，同量のユーザデータを格納できることを意味しない．

通常のシステムではノードごとに使用する Topic が異なるため，各ノードが Discovery によって把握した既知の DataWriter/DataReader 数と，各エンドポイントのマッチ数は一致しない．これに対し本評価では，全ノードが同一の 15 個の Topic すべてに対して DataWriter と DataReader を 1 個ずつ持つ，通常ではとらない Topic 構成とした．これは，Topic ごとの既知の DataWriter/DataReader 数と各エンドポイントのマッチ数を一致させることで，エンドポイントのマッチ数から発見状況を計測しやすくするための工夫である．Discovery では，全ノードが全エンドポイントの情報を把握した上でマッチするかを判定するため，各エンドポイントがマッチするかどうかは Discovery でやりとりされるメッセージに影響を与えない．

Discovery 完了時間は，ノード群の起動開始を基準とし，各ノードが持つ 30 個のエンドポイントすべてについて，マッチした相手エンドポイント数が期待値に到達した時刻をログから求める．期待値は，他の 203 ノードが同じ Topic に持つ相手エンドポイントをすべて発見した状態に対応する．DDS 実装が同一ノード内の相手エンドポイントをマッチ数に含める場合は，その 1 個を加えた値をログ上の期待値とする．各ホストの全ノードについて到達時刻を求め，その最大値をその試行の Discovery 完了時間とする．各実装について 3 回ずつ測定を行う．

ⓒ2026 Information Processing Society of Japan

4.2 Discovery 完了時間と発見後の安定性

図 1 に，エンドポイントのマッチとノードごとの発見完了の時間変化を示す．図（a）は，各エンドポイントのマッチ数を期待値で正規化し，両ホストの合計 6,120 エンドポイントについて平均した値である．期待値は Umber DDS では 203，同一ノード内の相手エンドポイントを含める Cyclone DDS では 204 とする．図（b）は，その時点で 30 個のエンドポイントすべてのマッチ数が期待値に達しているノードの割合である．線は 3 試行の平均，帯は最小値から最大値までの範囲を表す．起動後 60 s には提案手法の全試行で全ノードの発見が完了しているのに対し，Cyclone DDS の各試行の平均マッチ数比は 39.4–39.7%であり，発見を完了したノードはなかった．

各試行の Discovery 完了時間を**図 2** に示す．棒は 3 試行の平均，誤差棒は標本標準偏差，黒点は各試行の値を表す．提案手法の平均と標本標準偏差は 49.1 s と 5.4 s，Cyclone DDS では 360.4 s と 0.2 s となり，平均値から求めた短縮率は約 86.4%となった．提案手法は全試行で 55 s 以内に発見を完了したのに対し，Cyclone DDS では全試行で 360 s 以上を要した．

発見後の安定性について，提案手法では全試行を通じてマッチ数の減少がなく，全エンドポイントが期待値へ到達した後も，起動基準時刻から 600 s までその値を維持した．一方，Cyclone DDS では全試行で発見完了前にマッチ数の減少を観測したが，発見完了後は 600 s まで期待値を維持した．この減少を**図 3** に示す．図（a）は，Cyclone DDS の全試行の DataWriter のうち減少イベント数が最多の 1 個について，減少が生じた区間を拡大した図である．この DataWriter のマッチ数（既知の DataReader 数）は，例えば 230.6 s から 233.7 s の間に 165 から 162 へ減少した．図（b）は，2 台のホスト（A，B）における各試行のアンマッチ累積件数を示す．丸印は各試行の Discovery 完了時刻である．両ホストを合計したアンマッチ数は試行順に 1,647 件，1,737 件，1,510 件であり，いずれも発見完了前に発生した．この件数は，複数エンドポイントでのアンマッチをそれぞれカウントしたものであり，ノードの離脱数を表すものではない．

4.3 Discovery 完了時間に差が生じた要因

受信側の負荷を調べるため，**図 4** に NIC 受信帯域と Discovery 用マルチキャストソケットのドロップ数を示す．図（a）は NIC の受信バイトカウンタの差分から求めた 10 s 区間の平均帯域を，2 ホストについて平均した値である．図（b）は，RTPS の Discovery 用ポート 7400 と提案手法の情報交換用ポート 30000 について，ソケットごとの初回観測値からのドロップ数増分を両ホストで合計した値である．ユーザデータ用マルチキャストポート 7401 と，通信の種類をポート番号だけでは判別できないユニキャストソケットは図（b）の集計に含めない．いずれも線は 3 試行の平均，帯は最小値から最大値の範囲を表す．

図 4（a）では，起動後 100–590 s の平均受信帯域は提案手法で約 208 Mbit/s，Cyclone DDS で約 219–220 Mbit/s であり，観測した 10 s 平均の最大値も 2 ホスト平均では両実装で 270 Mbit/s 未満であった．ホスト別に見ても，全試行の送受信帯域の 10 s 平均の最大値は 282 Mbit/s 未満であり，この時間粒度では 1 Gbit/s の回線帯域を使い切っていない．それにもかかわらず，図 4（b）に示す 600 s までのソケットドロップ数増分は，提案手法の約 754–883 万に対し，Cyclone DDS では各試行で約 1.73–1.75 億に達した．なお，同じマルチキャストパケットが複数ノードのソケットへ配信され，そのドロップがソケットごとに別々にカウントされ得るため，図（b）の値はネットワーク上の異なるパケットの損失数を表すものではない．しかし，Cyclone DDS では Discovery が完了する 360 s 付近までドロップが継続して増加しており，この傾向は，先行研究 [7] が Fast DDS で報告した，エンドポイント発見と受信ソケットのドロップが同時に進行する観測結果と整合する．

このように 10 s 平均の回線帯域に余裕がある状況でソケットドロップが多発した要因として，バイト数ではなくパケットの処理頻度の違いを調べるため，**図 5** にホスト A，B それぞれの NIC で観測した受信・送信 PPS（packets per second）を示す．累積受信・送信パケット数の差分を区間の秒数で割り，帯域と同じ 10 s 区間の平均として求めた．時系列図の帯域と PPS の算出では，約 1 s 間隔の観測値を 10 s ごとの区間境界で線形補間し，全ログで観測値が存在する 10–590 s を対象とする．線は各ホストの 3 試行の平均，帯は最小値から最大値の範囲であり，ホスト間の平均はとらない．

図 5 より，提案手法では両ホストの送受信とも全試行・全区間で 20 kpps 未満に収まっているのに対し，Cyclone DDS では Discovery が完了する 360 s 付近まで 30 kpps 前後の高い PPS が継続していることが分かる．特に発見完了直前の 340–360 s には Cyclone DDS の受信・送信 PPS が急増し，各試行の 10 s 平均の最大受信 PPS はホスト A で 68.4–95.4 kpps，ホスト B で 68.1–90.9 kpps に達した．

この Discovery の進行段階ごとの違いを定量的に比較するため，**表 4** に Discovery 完了前後の 2 区間の PPS，帯域，平均パケットサイズを示す．各実装・各試行自身の Discovery 完了時刻を T とし，全ログで観測値が存在する 1–590 s を，区間 I：$1 \le t < T$（Discovery 進行中），区間 II：$T \le t \le 590$（Discovery 完了後）に分割する．境界は 10 s 刻みに丸めず，実装・試行ごとに独立に定める．したがって，同じ区間番号でも実時間の範囲と区間長は異なり，各実装の発見中と発見後の状態をそれぞれ比較する．区間集計では，元の累積 NIC カウンタを区間境界で線形補間し，パケット数の増分を区間長で割って PPS を，バイト数

ⓒ2026 Information Processing Society of Japan

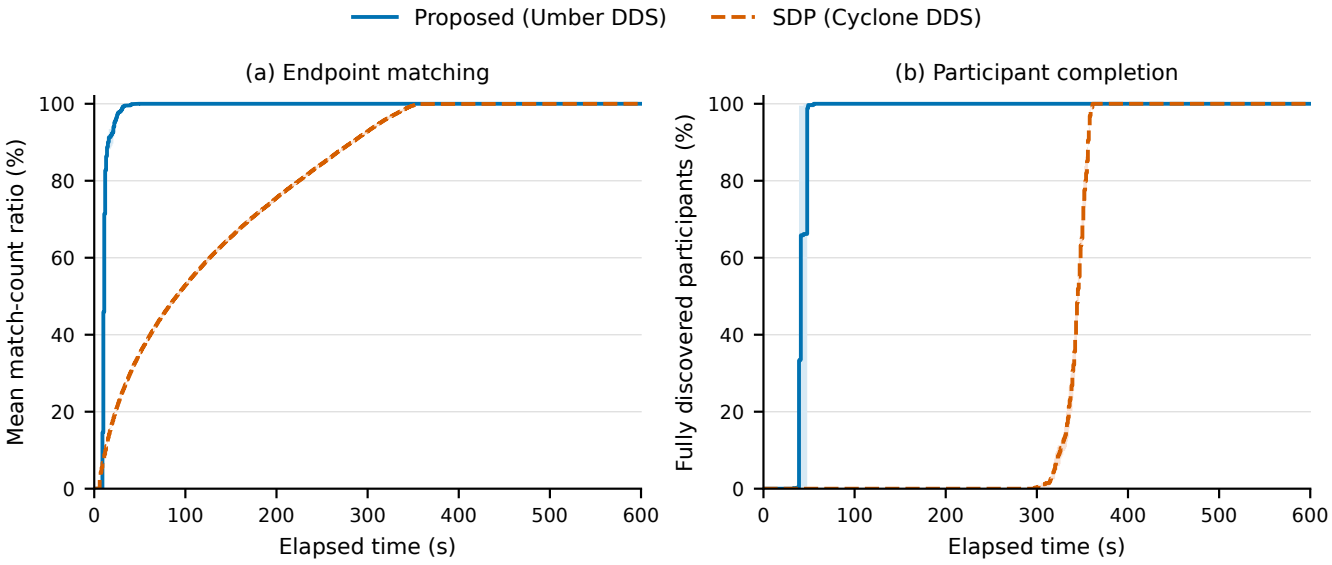

図 1 エンドポイントのマッチ数と発見を完了したノードの割合の時間変化．線は 3 試行の平均，帯は最小値から最大値の範囲を示す．

Fig. 1 Endpoint matching and node completion over time. Lines show means of three trials; shaded bands show their minima and maxima.

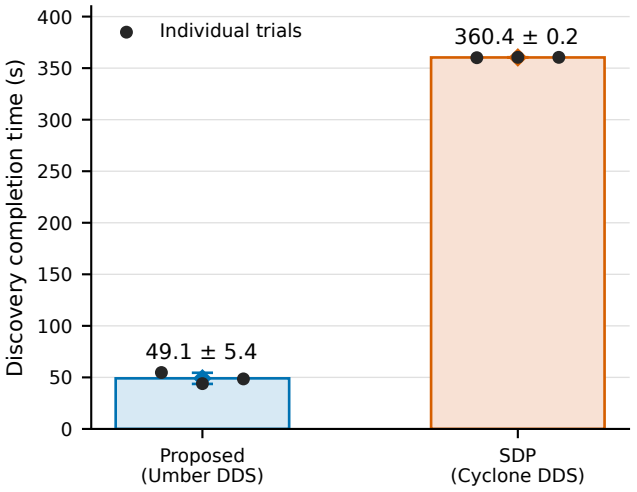

図 2 Discovery 完了時間．棒は平均，誤差棒は標本標準偏差，黒点は各試行の値を示す．

Fig. 2 Discovery completion time. Bars show means, error bars show sample standard deviations, and black dots show individual trials.

の増分をビットへ換算して区間長で割って帯域を求める．平均パケットサイズは，区間のバイト数増分をパケット数増分で割った値とする．表は，これらの値を試行ごとに求めた後の 3 試行の算術平均であり，ホスト A，B と送受信を分けて示す．

Discovery 進行中である区間 I では，提案手法の受信 PPS は約 15.6–15.7 kpps，Cyclone DDS は約 31.9–32.1 kpps であり，提案手法が約 51%小さかった．送信 PPS も提案手法で約 15.7 kpps，Cyclone DDS で約 31.9–32.1 kpps となり，同様の差が見られた．一方，受信帯域は提案手法で約 168 Mbit/s，Cyclone DDS で約 226 Mbit/s であり，帯域の差に比べて PPS の差が大きい．これは，平均受信パケットサイズが提案手法の約 1,341–1,346 byte に対して Cyclone DDS では約 881–889 byte と小さく，Cyclone DDS では小さいパケットが高い頻度で送受信されているためである．なお，本評価では各 DataWriter がマッチ後にユーザデータの送信を開始するため，区間 I にはユーザトラヒックの増加過程も含まれており，区間 I の帯域差は Discovery トラヒックの差のみでは説明できないことに注意が必要である．

一方，Discovery 完了後である区間 II では，受信・送信 PPS は提案手法で約 18.5 kpps，Cyclone DDS で約 18.7 kpps となった．平均パケットサイズも両実装で約 1,402 byte となり，発見中に見られた PPS とパケットサイズの差は完了後にはほぼ解消した．ユーザデータのサイズ，送信周期，マルチキャスト配送の条件は両実装で共通であり，このように区間 II の PPS と平均パケットサイズもほぼ一致することから，IP フラグメンテーションを含むユーザデータ由来の PPS が同程度となる期間では，PPS の差を Discovery メッセージの送受信頻度の差として近似できる．したがって，区間 I で提案手法の PPS が低い結果は，発見情報の交換や欠落修復に伴うパケット送受信頻度が Cyclone DDS より低く抑えられていることを示す．

以上の観測結果から，Discovery 完了時間に差が生じた要因は次のように考察できる．ただし，図の PPS は 10 s 区間，表の PPS は各進行段階の区間平均であり，CPU 使用率や受信キュー長との対応も解析していないため，短時間の帯域飽和の有無や，パケット処理能力の超過そのもの

ⓒ2026 Information Processing Society of Japan

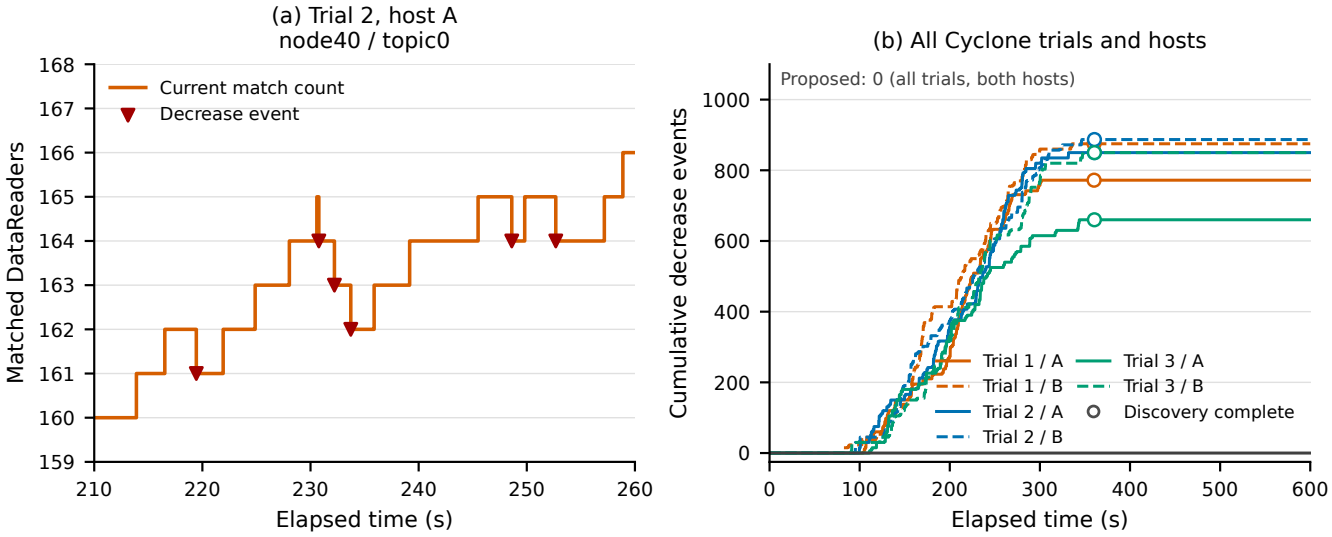

図 3 Cyclone DDS のアンマッチ発生数．(a) はアンマッチ数が最多の DataWriter の拡大例，(b) は各試行・各ホストのアンマッチ累積件数を示す．丸印は Discovery 完了時刻である．

Fig. 3 Match-count decreases in Cyclone DDS. (a) shows the writer with the most decrease events. (b) shows cumulative events by trial and host, with circles marking discovery completion.

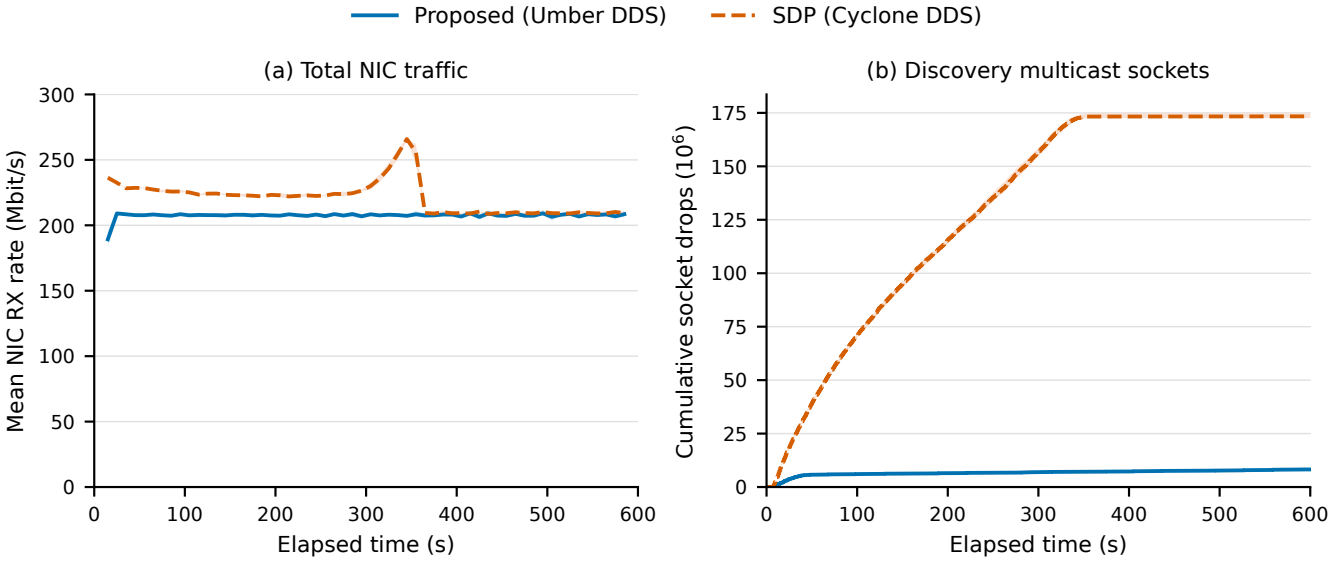

図 4 NIC 受信帯域と Discovery 用マルチキャストソケットのドロップ数増分．帯域は 2 ホスト平均，ドロップ数は 2 ホストの全対象ソケットの合計である．

Fig. 4 NIC receive rate and cumulative drop-counter increments on discovery multicast sockets. Rates are averaged over two hosts; drops are summed over all monitored sockets on both hosts.

は断定できない．その上でプロトコルの構成を比較すると，エンドポイント情報の交換と欠落修復の方法の違いが，Discovery に伴うパケット処理負荷の差を生むと考えられる．SDP では，ノードごとに SEDP でエンドポイント情報を交換し，HEARTBEAT と ACKNACK を用いて未達の情報を再送する．本条件では各ノードが 30 個のエンドポイントを持つため，多数の相手との情報交換に，確認応答や再送の小さな制御パケットが多数加わる．これに対し，提案手法ではホストごとのリーダーを通じてエンドポイント情報を集約し，マルチキャストで複数の受信者へ同じ情報を一括して配布する．欠落修復でも情報の件数とダイジェストを比較し，不一致があるエンドポイントレコード群を，そのエンドポイントが属するノード単位で再配布するため，再配布した情報を複数の受信者が利用できる．

ⓒ2026 Information Processing Society of Japan

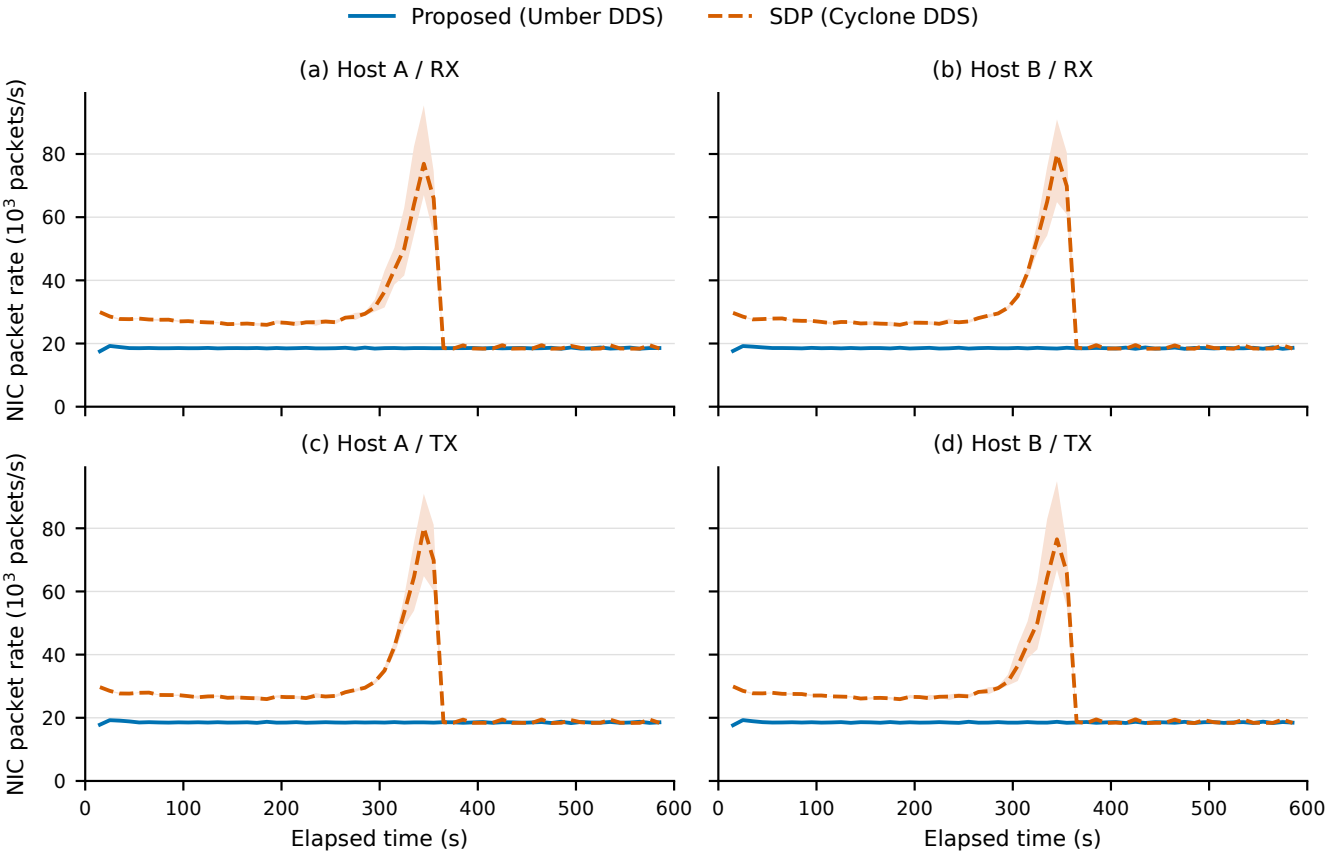

図 5 ホスト別の NIC 受信・送信 PPS の時間変化．(a)，(b) はホスト A，B の受信，(c)，(d) はホスト A，B の送信である．10 s 区間の平均値について，線は 3 試行の平均，帯は最小値から最大値を示す．

Fig. 5 NIC receive and transmit packet rates by host. Panels (a) and (b) show RX on hosts A and B; (c) and (d) show TX. Lines are three-trial means of 10-second rates, and shaded bands show their minima and maxima.

表 4 Discovery 完了前後の NIC 統計 (3 試行の平均)．区間 I：各実装自身の Discovery 完了まで，区間 II：完了後．区間境界は実装・試行ごとに異なる．

Table 4 NIC statistics before and after discovery completion, averaged over three trials. Phase I: discovery in progress; phase II: discovery complete. Each implementation and trial uses its own completion time as the boundary.

区間	方式	ホスト	受信			送信		
			PPS [kpps]	帯域 [Mbit/s]	平均サイズ [byte]	PPS [kpps]	帯域 [Mbit/s]	平均サイズ [byte]
I	CDP	A	15.62	168.19	1346.1	15.75	168.59	1338.6
I	CDP	B	15.67	168.06	1341.2	15.68	168.57	1344.2
I	SDP	A	31.86	225.99	888.6	32.10	226.13	881.3
I	SDP	B	32.10	226.11	881.2	31.85	225.97	888.6
II	CDP	A	18.53	207.82	1401.7	18.53	207.83	1401.7
II	CDP	B	18.53	207.82	1401.7	18.53	207.82	1401.7
II	SDP	A	18.68	209.43	1401.7	18.68	209.42	1401.7
II	SDP	B	18.68	209.44	1401.7	18.68	209.44	1401.7

この違いにより，相手ごとの確認応答や再送に伴う多数のパケット送受信を抑制し，パケットの受信処理や各ノードのソケットへの配信処理の負荷を減らすことで，Discovery 用ソケットでのドロップと発見遅延を抑制した可能性がある．

本評価では，204 ノードと 6,120 エンドポイントがユーザデータを送受信する条件で，提案手法を実装した Umber DDS が全エンドポイントの発見を完了してマッチ数を維

ⓒ2026 Information Processing Society of Japan

持し，標準 SDP を用いる Cyclone DDS より Discovery 完了時間を短縮したことを確認した．

5. まとめ

本研究では，大規模な DDS ネットワークにおける Discovery 完了時間の短縮を目的として，リーダー選出を用いた Centralized Discovery Protocol（CDP）を提案した．提案手法では，ホストごとに選出したリーダーへ同一ホスト内のエンドポイント情報を集約し，マルチキャストでドメイン内の各ノードへ配布するとともに，ダイジェスト照合と Snapshot 再配布によって欠落情報を修復する．204 ノード，6,120 エンドポイントがユーザデータを送受信する環境で評価した結果，標準 SDP を用いる Cyclone DDS では完了に 360 s 以上を要した Discovery を，CDP を実装した Umber DDS では 55 s 以内（平均 49.1 s）で完了し，完了時間を約 86.4%短縮できることを確認した．

今後の課題は 3 点ある．第一に，同一の DDS 実装上で Discovery 方式のみを切り替えた比較評価を行い，プロトコルの変更単体による寄与を定量化する必要がある．第二に，リーダーの離脱や障害が発生した際の再選出と情報再同期にかかる回復性能を評価する必要がある．第三に，CDP に対応したノードと標準 SDP のみに対応したノードが混在する環境での相互運用に向け，標準 SPDP を維持しつつ標準 SEDP と連携してエンドポイント情報を交換する方法を検討し，互換性と Discovery 性能を検証する必要がある．

謝辞　この成果は，国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の助成事業（JPNP21027）の結果得られたものである．

参考文献

[1] Object Management Group: Data Distribution Service (DDS), (online), available from ⟨https://www.omg.org/omg-dds-portal/⟩ (accessed 2026-10-19).
[2] Autoware Foundation: Autoware, (online), available from ⟨https://autoware.org/⟩ (accessed 2026-10-19).
[3] Object Management Group: DDS Interoperability Wire Protocol Version 2.5, Technical report, Object Management Group (OMG) (2022).
[4] eProsima: Fast DDS, (online), available from ⟨https://fast-dds.docs.eprosima.com/en/latest/⟩ (accessed 2026-10-19).
[5] Eclipse Foundation: Cyclone DDS, (online), available from ⟨https://cyclonedds.io/⟩ (accessed 2026-10-19).
[6] TIER IV, Inc.: Umber DDS, (online), available from ⟨https://github.com/tier4/umber_dds⟩ (accessed 2026-10-19).
[7] 富重亮佑，樽谷優弥，横平徳美：DDS におけるノード・モジュール発見プロトコルの性能解析，電子情報通信学会ソサイエティ大会，No. B-6-22, 岡山県 (2025).
[8] Park, J., Joe, I. and Kim, W.-T.: An efficient discovery protocol of large-scale CPS middleware for real-time control system, *Studies in Informatics and Control*, Vol. 23, No. 1, p. 24 (2014).
[9] eProsima: Discovery Server Settings (Fast DDS 3.4.2), (online), available from ⟨https://fast-dds.docs.eprosima.com/en/v3.4.2/fastdds/discovery/discovery_server.html⟩ (accessed 2026-10-28).
[10] eProsima: Environment Variables (Fast DDS 3.x), (online), available from ⟨https://fast-dds.docs.eprosima.com/en/3.x/fastdds/env_vars/env_vars.html⟩ (accessed 2026-10-28).
[11] eProsima: Use ROS 2 with Fast-DDS Discovery Server (Fast DDS 2.6.11), (online), available from ⟨https://fast-dds.docs.eprosima.com/en/v2.6.11/fastdds/ros2/discovery_server/ros2_discovery_server.html⟩ (accessed 2026-10-28).
[12] Eclipse Foundation: Eclipse Cyclone DDS 0.10.5 Source Code, (online), available from ⟨https://github.com/eclipse-cyclonedds/cyclonedds/tree/0.10.5⟩ (accessed 2026-10-28).
[13] Linux man-pages project: socket(7) — Linux Manual Page, (online), available from ⟨https://man7.org/linux/man-pages/man7/socket.7.html⟩ (accessed 2026-10-28).

ⓒ2026 Information Processing Society of Japan