A-017

任意の連結グラフにおける 2 頂点対点素パスの構築判定 自己安定アルゴリズム

A Self-stabilizing algorithm to find two node disjoint paths between two senders and targets on an arbitrary connected graph

北岡拓馬 [1] Takuma Kitaoka
金鎔煥 [1] Yonghwan Kim
片山喜章 [1] Yoshiaki Katayama
増澤利光 [2] Toshimitsu Masuzawa

名古屋工業大学 [1] Nagoya Institute of Technology
大阪大学 [2] Osaka University

1 研究背景

近年，WSN(Wireless Sensor Network) の分野において，複数の送受信ノード間を結ぶ経路を構築するルーティングアルゴリズムが考えられている．ここで，ルーティングアルゴリズムで構築された経路上の各ノードは，次のノードの識別子やポート番号などの情報を保持することで経路情報を保持する．しかし，この経路情報はノードの一時故障などにより失われる場合があり，それにより構築された経路が切断される可能性がある．そこで一時故障に耐性を持つシステムの設計が望まれる．そこでルーティングアルゴリズムを自己安定アルゴリズムとして設計することを考える．自己安定アルゴリズムとは，任意の初期状況から有限時間内に解状況に到達する分散アルゴリズムである．この特徴から任意のノードで一時故障が起こってもその状況を初期状況として動作し，有限時間内に解状況に再び収束するため，一時故障に耐性をもつ分散アルゴリズムといえる．

ネットワークにおける複数の送受信ノード間のルーティング自己安定アルゴリズムは数多く提案されている [1]. しかし，これらのアルゴリズムで構築される複数の経路においては共通のノードが存在する可能性がある．この複数の経路に含まれる共通のノードにおいて停止故障が生じた場合，そのノードを経路上に含む複数の経路が利用できなくなることとなる．よって，複数の経路を点素 (複数の経路間で共通のノードがない) にすることが考えられる．[2] では 2-連結グラフ上において 2 つの送信ノード $S = \{s_1, s_2\}$ と受信ノード $T = \{t_1, t_2\}$ 間を結ぶ 2 本の点素パス (2 頂点対点素パス) が提案されている．ここで 2 頂点対点素パスは送受信ノードのペアを考えていないことから，$s_1 - t_1, s_2 - t_2$ パスの場合と $s_1 - t_2, s_2 - t_1$ パスの場合が考えられる (図 1).

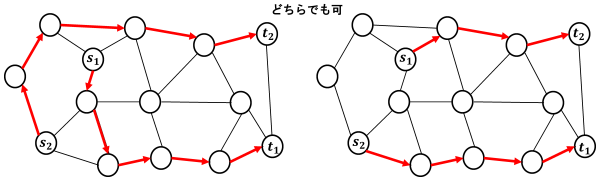

図 1　2 頂点対点素パスの例

[2] は，s_1 と t_1，s_2 と t_2 をそれぞれ結ぶ 2 本の点内素パス (始点と終点以外が異なる複数のパス) を利用することで 2 頂点対点素パスを構築する．そこで本研究では，仮想ノードを用いることによって，[2] を任意の連結グラフ上に拡張したアルゴリズムを提案する．

2 モデルと問題定義

分散システムをグラフ $G = (V, E)$ で表す. グラフ G は連結な単純グラフであり, 各ノードは固有の識別子 (ID) を持たないが，送信ノード s_1,s_2 と受信ノード t_1,t_2 は区別可能であると仮定する．各ノードは自身と隣接している辺集合を局所的なポート番号を用いて区別することが可能である．これらのポート番号は隣接ノードからは把握できない．各ノードは隣接ノードの状態を直接参照することで通信を行う状態通信モデルを仮定する. アルゴリズムの実行は D デーモン (複数のプロセスが同時に動作することを許すスケジューラ) とする.

グラフ $G = (V, E)$ と G 上の送信ノード集合 $S = \{s_1, s_2\}$ と受信ノード集合 $T = \{t_1, t_2\}$ が与えられ, 任意の初期状況から,S, T を結ぶ 2 頂点対点素パスを構築できたとき, 問題が解けたという. またここでいうパスの構築とは，2 頂点対点素パスに含まれるノードがパス上の次のノードのポート番号を保持している状態を指す．また，S, T を結ぶ 2 頂点対点素パスが構築できないとき，ネットワーク上の各ノードが NG を出力する．

3 関連研究：2 点間の点内素パス構築アルゴリズム

本節では，連結グラフ上の始点と終点を結ぶ 2 本の点内素パスを構築する自己安定アルゴリズム [3] を紹介する．このアルゴリズムは，任意の連結グラフ上において始点と終点を結ぶ 2 本の点内素パスが構築できるときはパスを構築し，構築できないときはネットワーク上の各ノードにおいて NG を出力する．

2 本の点内素パスの構築には，始点と終点を結ぶ最短パス P と，P のリンクパスを利用する．ここで P のリンクパスとは，始点 o と終点 w 以外が P と異なるパスである．また終点 w は，始点 o から P 以外のパスで到達可能な P 上のノードのうち，P 上において P の終点に一番近いノードである．アルゴリズムにおいては以下を満たすリンクパスの集合 $P_1, P_2, ...P_k$ を構築する．ここで以下の ds_v とは，P 上の始点から v の距離を表す.

(1) P_1 は始点 $o_1 = s$ である.

(2) $1 < i \leq k$ の各リンクパス P_{i-1} は，以下のような o_i

から w_i まで伸びるリンクパス P_i を持つ．

$$\begin{cases} ds_{o_1} < ds_{o_2} < ds_{w_1} \ (i = 2) \\ ds_{w_{i-2}} \leq ds_{o_i} < ds_{w_{i-1}} \ (2 < i \leq k) \\ ds_{o_i} < ds_{o_{i+1}} \ (1 \leq i < k) \end{cases}$$

(3) $1 < i \leq k$ において，P 上の o_i は ds_{w_i} が最大となるように決定される．

(4) P_k は終点 $w_k = t$ である．

[3] は公平な合成を用いている：複数の自己安定アルゴリズム $A_1, A_2, ..., A_k$ において,A_i までの安定状況が A_{i+1} の入力となり, アルゴリズム A_i は全ての $A_j (j < i)$ の変数を読めるが書き込みはできない．

アルゴリズムは以下の 4 つの異なる自己安定アルゴリズムの公平な合成からなる 4 層構成となっている．

[第 1 層] 始点と終点を結ぶ最短パスの構築

[第 2 層] BFS 森の構築

[第 3 層] リンクパスの構築

[第 4 層] 2 本の点内素パスの構築

[3] は第 1 層で始点と終点を結ぶ最短パス P を構築する．その後，第 2 層で P 上の頂点を根とする P の辺以外を用いた BFS 森を構築し，第 3 層で BFS 森からリンクパスを構築する．そして第 4 層で下図のように P とリンクパスを交互に用いることで 2 本の点内素パスを構築する (図 2)．

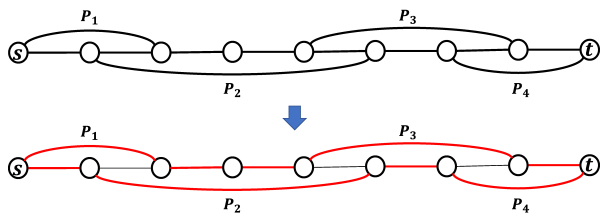

図 2　P とリンクパスを用いた構築の例

4　提案アルゴリズム

アルゴリズムの基本戦略として，まず連結グラフ G に $S = \{s_1, s_2\}$ のみに辺をもつ仮想ノード V_s と $T = \{t_1, t_2\}$ のみに辺をもつ仮想ノード V_t を加えたグラフ G' を考える．このとき，G' 上で V_s と V_t を結ぶ 2 本の点内素パスを構築したとき，G 上の S と T を結ぶ 2 頂点対点素パスが構築できる (図 3)．

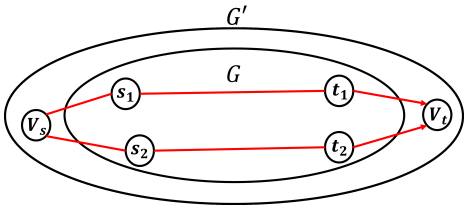

図 3　V_s, V_t を結ぶ 2 本の点内素パス

つまり，s_1, s_2 において共通の仮想ノード V_s を，t_1, t_2 において共通の仮想ノード V_t をそれぞれシミュレートし，その上で第 3 節で紹介したアルゴリズム [3] を適用することによって 2 頂点対点素パスを構築できる．また，[3] は任意の連結グラフにおいて 2 本の点内素パスが構築できない時はネットワーク上の全てのノードにおいて NG を出力するような構築判定も行うことから，仮想ノードのシミュレートによって 2 頂点対点素パスの構築判定も行うことができる．

提案アルゴリズムは，[3] の各層において仮想ノードをシミュレートすることによって実現する．

まず第 1 層は仮想ノード V_s, V_t 間の最短パス P を求める．これはつまり始点 s_1, s_2 と終点 t_1, t_2 の各始点と各終点のペア間の最短パスのうち，最短のパスを求めることである．そのために，s_1, s_2 の各ノードを根とする 2 つの BFS 木を構築することで各始点と各終点のペア間の最短パスを構築したのちにそれぞれのパスの長さを求め，そのそれぞれの長さを s_1, s_2 間で共有することで最短のパスを求める．ここで s_1, s_2 間でのパスの長さの共有のために，$s_1 - s_2$ 最短パスを s_1 を根とする BFS 木から構築する必要がある．この $s_1 - s2$ 最短パスに沿って s_1, s_2 がそれぞれのパスの長さを伝搬することによって，パスの長さを共有できる．

次に第 2 層は，P 上の頂点を根とする P の辺以外を用いた BFS 森を構築する。この時，必ず V_s を根とする BFS 木を構築する必要がある。これは s_1, s_2 のうち P 上でない方のノードを根とする BFS 木を構築することで，V_s を根とする BFS 木をシミュレートできる．

次に第 3 層は，BFS 森を利用して各リンクパス P_i の始点 o_i と終点 w_i を求めることで，P 上のリンクパスを求める．このとき [3] では，P 上における各ノード v に V_s からの距離値 (P 上における V_s から v までのホップ数) を割り当て，また P 上の各ノードにおいて BFS 木を用いて到達できるノードの距離値のうち最大のもの (最大距離値) を計算する．この計算の際に，s_1, s_2 において共通の仮想ノード V_s をシミュレートするために，V_s の最大距離値を共有する必要がある．この最大距離値は，s_1, s_2 のうち P 上でない方のノードが BFS 木から得ることができ，また得た最大距離値を $s_1 - s_2$ 最短パスに沿って伝搬することによって共有できる．

最後に第 4 層は，[3] と同様に P とリンクパスを交互に用いることで 2 本の点内素パスを構築する．

5　今後の課題

提案アルゴリズムの実行時間の解析，メモリ量の解析などが課題としてあげられる．

参考文献

[1] Y. Kim et al., "A Self-Stabilizing Algorithm for Constructing an ST - Reachable Directed Acyclic Graph When $|S| \leq 2$ and $|T| \leq 2$," in Proceedings of the 39th IEEE International Conference on Distributed Computing Systems (ICDCS), 2019, pp. 2228–2237.

[2] 北岡拓馬, 金 鎔煥, 片山喜章,2-連結グラフ上の 2 頂点対点素パスを構築する自己安定アルゴリズム, 信学技報, vol. 122, no. 33, COMP2022-4, pp. 25-32, 2022 年 5 月.

[3] Rachid Hadid,et al. "A stabilizing algorithm for finding two node-disjoint paths in arbitrary networks" International Journal of Foundations of Computer Science, 28(04):411-435, 2017.