O-016

配電SCADAネットワークにおける情報収集時間短縮手法
A Data Collection Time Reduction Method for Power Distribution SCADA Networks

小島英春 † 芳中宗一郎 ‡ 土屋達弘 †
Hideharu Kojima Souichiro Yoshinaka Tatsuhiro Tsuchiya

1. はじめに

近年，発電機器を設置する住宅が増加しており，特に太陽光発電については，多くの家庭が導入を行っている [1]．これらの発電機器が発電することにより，配電網では末端側の電圧が上昇することによる逆潮流が発生する恐れがある．逆潮流の発生により，配電網において電圧の変動が生じ適正な電圧を逸脱する可能性があり，そのため，配電網において電圧の監視は重要である．電圧などのセンサー情報を監視するため，配電運用システムにおいては，SCADA（Supervisory Control And Data Acquisition)[2] システムを用いて監視制御を行っている．SCADA システムは，配電系統の監視と開閉器の制御の部分を担い，監視の頻度については，現状の SCADA システムでは，開閉器の状態情報については 10 分に数回程度，センサの計測値については 1 時間に数回程度の周期で情報収集を行っている．SCADA システムはオンラインシステムのサーバと，変電所にある中継装置である RTU(Remote Terminal Unit)，電柱上の配電設備に接続された FTU(Feeder Terminal Unit) とそれらを接続する通信ネットワークから構成される．通信ネットワークは複数の独立したネットワークからなり，RTU はそれぞれのネットワーク上で周期的に，1 対 1 で順番に通信するポーリング方式 [3] により FTU と通信する．

[1] にみられるように太陽光発電の導入がここ数年，特に増加の傾向にあるため，逆潮流の発生により電圧が想定される範囲を逸脱する可能性が高まると考えられる．この電圧の変化を，早期に発見するためには，SCADA システムの監視頻度の向上が必要である．そこで，SCADA システムで利用される配電 SCADA ネットワーク上に，集約点と呼ばれる情報収集を行う端末を配置し，RTU と集約点が並行して情報収集を行うことで情報収集にかかる時間の短縮を行う．集約点は通信線上に配置され，配置場所より末端側にあるすべての FTU から情報を RTU と並行してポーリング方式で収集する．そして，情報収集が完了した後に，RTU からのリクエストに対して収集した情報全てを RTU に送信する．本研究では，効率よく情報収集が可能となる集約点の配置場所の決定と，その配置場所決定に関わる計算時間の短縮を行う手法について提案する．

2. 配電 SCADA ネットワーク

本研究で対象とする配電 SCADA ネットワークについて述べる．配電 SCADA ネットワークは，情報を収集する RTU と，情報を収集される FTU からなる．図 1 のような RTU を根とする根付きグラフ G として扱う．G は次の様に表す．

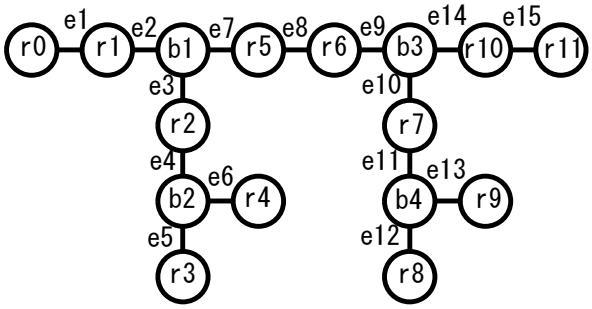

図 1: 配電 SCADA ネットワークのグラフの 1 例

$G = (V, E, root)$
$E = \{e1, e2, \ldots, el\}$
$V = V^R \cup V^B \cup \{root\}$
$V^R = \{r1, r2, \ldots, rn\}$
$V^B = \{b1, b2, \ldots bm\}$

V^R は FTU を表す節点，V^B は分岐点を表す節点，$root$ は RTU を示す節点である．また，根 $root$ と任意の節点 v_i の経路に節点 v_j が存在する場合，v_i は，v_j の下流，v_j は v_i の上流とする．上流，下流の関係は，辺と辺，辺と節点に対しても同様である．

集約点　集約点とは，RTU と並行して FTU から情報を収集する端末である．集約点の配置場所は，グラフの辺である．集約点に情報を収集される FTU は，集約点が配置された場所より，下流の FTU である．集約点の配置の種類として，階層的配置がある場合と階層的配置がない場合の 2 種類存在する．階層的配置があるとは，与えられたグラフ G の根と葉の経路上に集約点が配置された辺が 2 つ以上存在する場合を表す．また，階層的配置がない場合は，G の根と任意の葉の経路上に集約点が配置された辺が 1 つ以下の場合を表す．

情報収集の手順

はじめに通信に関する条件を仮定する．2 つの機器間の情報の授受を通信と呼ぶ．

- 通信は，RTU と FTU，RTU と集約点，集約点と FTU の 2 つの機器間で行われる．
- 通信は分割不可能である．
- 集約点は下流の FTU すべてから情報を得た場合のみ，RTU と通信を行う．
- RTU と集約点による通信によって，RTU はその集約点の下流の FTU の情報を収集できる．

次に，情報収集時間に必要な定義について述べる．システム全体が情報収集を開始する時刻を 0 とする．また，通信に必要な時間を以下に定める．

†大阪大学大学院情報科学研究科, Graduate School of Information Science and Technology, Osaka University
‡大阪大学基礎工学部, School of Engineering Science, Osaka University

d :1 つの情報を収集するために必要な通信時間.

p :機器間通信のオーバーヘッドの通信時間.

t^r :RTU と FTU，集約点と FTU 間の通信に必要な時間. p と FTU の情報分 d が必要であるため，$t^r = p + d$ である.

t^a :RTU と集約点とが通信するために必要な時間．p と集約点が収集した FTU 数 r の情報分 $d \times r$ が必要であるため，t^a=$p + d \times r$ である.

RTU は自身が収集する全ての FTU との通信が完了し，全ての集約点との通信が完了した時点で，全ての FTU の情報を収集することができる．これらの通信を以下のように定める.

C :通信の集合． $C = C_a \cup C_r$.

C_r :RTU と FTU との通信の集合． $C_r \subset C$.

C_a :RTU と集約点との通信の集合． $C_a \subset C$.

集約点の配置が与えられた場合，収集時間が最短となる通信の方法について述べる．集約点が下流のすべての FTU から情報収集を完了する時刻は，明らかに，時刻 0 から FTU との通信を間を空けずに全 FTU に対して実行した場合に，もっとも早くなる．そこで，通信 $c \in C$ が実行可能となる時刻を $a(c)$，その c の通信に必要な時間を $t(c)$ とすると，それぞれは，式 (1)(2) と表すことができる．また，この通信により送信される FTU の情報の数を，$r(c)$ と表す.

$$a(c) = \begin{cases} 0 & (c \in C_r) \\ t^r \times r(c) & (c \in C_a) \end{cases} \quad (1)$$

$$t(c) = \begin{cases} p + d & (c \in C_r) \\ p + d \times r(c) & (c \in C_a) \end{cases} \quad (2)$$

C に含まれる各通信 c をどの時刻に行えば収集時間が最短になるか，という問題について考える．通信 c の開始時刻を定めたものをスケジュールと呼ぶ．形式的には，スケジュール s を，C に含まれる通信 c への開始時刻 $s(c)$ の割当で式 (3) と式 (4) を満たすものと定義する.

$$s(c) \geq a(c) \quad (c \in C) \quad (3)$$

$$s(c) \geq s(c') + t(c') \text{ または}$$

$$s(c') \geq s(c) + t(c) \quad (c, c' (\neq c) \in C) \quad (4)$$

スケジュール s が与えられたとき，$\max_{c \in C}\{s(c) + t(c)\}$ が全 FTU の情報の収集が完了する時刻である．この時刻がもっとも早いスケジュールを最適と呼ぶ．このとき，$a(c)$ が小さい順に通信を開始する方法が最適となる．つまり，C の要素 c を式 (5) を満たす様に，$a(c)$ の昇順に $c^1, c^2, ...$ と並べて，式 (6) と式 (7) を用いて計算することで，最適なスケジュールが求められる.

$$a(c^i) \leq a(c^{i+1}) \quad (5)$$

$$s(c^1) = a(c^1) \quad (6)$$

$$s(c^{i+1}) = \max\{s(c^i) + t(c^i), a(c^{i+1})\} \quad (7)$$

これは，上記の最適スケジュールを求める問題が，単一マシンへのジョブスケジューリングにおいて，ジョブ集合を C，各ジョブ c のリリース時間を $a(c)$，実行時間を $t(c)$ として，ジョブの完了時刻の最大値を最小にする問題 $(1||C_{max})$ と等価であり，このジョブスケジューリング問題ではリリース時間順にタスクをスケジューリングする方法が最適であることから結論できる [4][5]．このことから，RTU は，まず自身が担当するすべての FTU と通信を実行した上で，下流の FTU の情報収集を完了した集約点から順番に通信した場合に，情報収集時間が最短となる.

3. 提案手法

本節では，収集時間を短くするための集約点配置場所決定方法について述べる．集約点配置場所を求めるにあたり，全ての配置可能場所を総当たりに探索する方法では，計算時間が非常に長くなる可能性がある．そこで，階層的配置を行わない場合に最短の収集時間となる配置を求める提案法 1 と，階層的配置を行う場合の収集時間を短縮する集約点配置方法の提案法 2 の 2 つを提案する．提案法 1 の入力は，与えられた根付きグラフ $G = (V, E, root)$，集約点数 n であり，出力は，収集時間が最短となる階層的配置を行わない集約点の配置である．提案法 2 の入力も，与えられた根付きグラフ $G = (V, E, root)$ 及び，集約点数 n である．出力は，収集時間が提案法 1 で求めた集約点の配置による収集時間以下となるような階層的配置を行う場合の集約点の配置である.

3.1. 提案法 1：階層的配置を行わない集約点の配置

ここでは，集約点の配置を階層的に行わない場合に収集時間が最短となる場所を求める方法を述べる．配置場所を探索木を用いて探索するにあたり，以下の方法を用いて，探索木のサイズを小さくする方法を提案する.

- **収集時間の見積もり**
  最短の収集時間となる集約点の配置を探索しているとき，その配置が n 個の集約点を全て決定していない状況において，収集時間の見積もりを行う．その見積もりした時間を，探索木の探索打ち切りに用いる.

- **はじめの集約点決定**
  見積もりした時間とその時点で最短の収集時間の比較を行うにあたって，最短の収集時間が，実際の最短時間に近い方が，見積もりによる探索打ち切りが行われ，探索木のサイズを小さくすることができる．そのため，探索を開始して最初に選択する集約点を戦略的に求める.

- **探索木の探索方法**
  収集時間の見積とその時点の最短の収集時間を比較することにより，探索木の深さと幅の削減を行う条件を示す.

3.1.1. 集約点の選択

集約点の配置をどの様に選択するかについて述べる．本節で利用する記号を次の様に定義する.

A :集約点配置の候補.
a_k :k 番目に選択された集約点を配置する辺.
t_{best} :その時点で最短の収集時間.
A_{best} :収集時間が t_{best} となる集約点の配置.
e_i :集約点を配置可能な辺.
e_{fit} :探索開始時に最初に選択される辺.
$est1(A)$:A の配置を見積 1 で計算した見積時間.
$est2(A)$:A の配置を見積 2 で計算した見積時間.
$est(A)$:$\max\{est1(A), est2(A)\}$.
$r(e_i)$:e_i に集約点を配置した際に収集する FTU 数.
$c_k \in C_a$:RTU と集約点 a_k との通信.

n 個の集約点の配置場所を選択するにあたり，選択された集約点 a_k は，次の条件を満たす.
$r(a_1 = e_j) \geq r(a_2 = e_k) \geq \ldots r(a_n = e_l)(j > k > l)$
集約点の配置が 1 から k まで決定している場合，A は $(a_1, \ldots, a_k, \perp, \ldots)$ となっており，選択された順に並んでいる．また，e_i は $r(e_i)$ の値を元に昇順に並んでいるものとする．つまり，$r(e_i) \leq r(e_{i+1})$ である．

集約点の配置を求めるアルゴリズムを図 2 に示す．図 2 が終了した際の t_{best} が，与えられた SCADA ネットワークのグラフにおいて，集約点が n の時の最短の収集時間を表し，その時の集約点の配置は A_{best} である．

集約点の選択は，収集する FTU 数が多いものから選択する．このとき，a_1 を $r(e_i)$ の値を元に適切な辺を選択する．$r(e_i)$ が大きな値や，小さな値となる辺を選択すると，$est(A)$ で求められる値が最短の収集時間から大きく離れてしまう．最短の収集時間に近しい値を取るように，集約点の配置場所を求めるために，最初に選択される $e_{fit}(n \leq fit \leq |V^R|)$ を選択する必要がある．図 2 の 3 行目から 7 行目までは，a_1 として選択される辺 e_i の $r(e_i)$ が小さくなる様に a_1 を選択していき，8 行目から 12 行目までは，$r(e_i)$ が大きくなる様に a_1 を選択する．a_1 の $est(A)$ がその時点の t_{best} より小さければ，13 行目以降の処理を行う．

収集時間の見積方法である $est1(A)$ と $est2(A)$ は 3.1.2 節に詳細を述べる．また，探索木の探索開始時に最初に選択される辺 e_{fit} の決定方法については，3.1.3 節において詳細を説明する．

3.1.2. 収集時間の見積方法

本節では，図 2 で行っている見積方法について述べる．既に決定された集約点からは次の 2 通りの方法で，収集時間の見積もりを行う．

見積 1($est1(A)$)
見積 1 では，既に決定している集約点に注目して収集時間の見積もりを行う．a_k まで集約点が決定している場合，a_k，a_{k-1}，$\ldots a_1$ という順番で集約点と通信を行う．$s(c_1)$ は，式 (8)，(9)，(10) から求められる．$s(c_1)$ は a_1 との通信の開始時間であるため，見積もりされる収集時間は，$s(c_1) + t(c_1)$ である．この次に配置される a_{k+1} がどこに配置されたとしても，$s(c_1) + t(c_1)$ 以上の時間が必要である．

```
1:  A := (a_1 =⊥, a_2 =⊥, ..., a_n =⊥)
2:  t_best := t^r × |V^R|
3:  for(i := fit;i ≥ n;i − −){
4:    a_1 := e_i;
5:    if(t_best ≤ est(A)){continue;}
6:    else{selectAggregator(A,i − 1,2);}
7:  }
8:  for(i := fit + 1;i ≤ |E|;i + +){
9:    a_1 := e_i;
10:   if(t_best ≤ est(A)){continue;}
11:   else{selectAggregator(A,i − 1,2);}
12: }
13: function selectAggregator(A,l,k){
14: for(i := l;i ≥ n − k;i − −){
15:   a_k := e_i;
16:   if(t_best ≤ est(A)){continue;}
17:   else {
18:     if(k = n){
19:       t_tmp := calcTime(A);
20:       if(t_best > t_tmp){t_best := t_tmp;A_best := A;}
21:     }
22:     else{selectAggregator(A,i − 1,k + 1);}
23:   }
24: }
```

図 2: 集約点の配置を決定するアルゴリズム

$$a(c_i) \leq a(c_{i-1}) \quad (8)$$
$$s(c_k) = a(c_k) \quad (9)$$
$$s(c_{i-1}) = \max\{s(c_i) + t(c_i), a(c_{i-1})\} \quad (10)$$

見積 2($est2(A)$)
a_k まで集約点が決定している場合，RTU が収集する FTU 数を推定し，それから収集時間の見積もりを行う．見積 1 では，既に決定している集約点に注目して収集時間の見積もりを行ったが，ここでは，未決定の集約点に注目して収集時間の見積もりを行う．RTU は自身が収集する FTU から情報を収集したのちに，集約点から情報を収集するため，RTU が収集する FTU から情報を収集する時間と集約点から情報を収集する時間は必要である．ここでは，RTU が収集する FTU 数が最小となるように未決定の集約点が収集する FTU 数を決定し，それらを用いて収集時間の見積もりを行う．

未決定の集約点として選択される可能性のある e_i は，a_k=e_l とすると $e_i(i < l)$ である．そこで，未決定の集約点は $n-k$ 個あるので，e_i から $r(e_i)$ が大きい順に $n-k$ 個を仮の集約点とする．この時選択される仮の集約点を用いて RTU が収集する FTU 数 UA を式 (11) を用いて計算する．UA が負の値の場合は，UA は 0 とする．

$$UA = |V^R| - \sum_{m=1}^{n} r(c_m) \quad (11)$$

全ての FTU の情報を収集するために必要な時間は，集約点から情報を得るために必要なオーバーヘッド p が集約点数分と UA 以外の FTU 数 $|V^R| - UA$ の情報を通信する時間である．これらを用いて，収集時間の見積もりを式 (12) を用いて計算する．

$$t^r \times UA + p \times n + d(|V^R| - UA) \tag{12}$$

3.1.3. 最初の集約点決定方法

図 2 において，探索開始時に選択される集約点を配置する辺を e_{fit} としている．探索木を探索するにあたって，最初に決定する収集時間 t_{best} がなるべく最短の収集時間に近い値になるように a_1 を選択したい．そうすることで，収集時間の見積もりにより，探索対象から削除されるものが多くなる．そこで，本節では，各集約点が収集する FTU 数をどのように配分すれば，最短の収集時間に近しい値となるかについて述べる．ここで得られる FTU 数の配分を行った集約点の配置を $\hat{A}$ と表す．そこで得られた FTU 数を利用して a_1 として最初に選択するべき e_i を示す i の値である fit を決定する．

集約点の配置を $A = (a_1, a_2, \ldots, a_n)$ とする．a_k の通信を $c_k \in C_a$ とする．収集する FTU が多い順に a_k を選択しているため，$r(c_k) \leq r(c_{k-1})$ である．この A は，式 (13) を満たし，$s(c_n)$ を式 (14) とすると，RTU が自身が収集する FTU から収集を終えた後，集約点から間を空けずに情報を収集することになるため，この時の収集時間 t は，式 (2) の $t(c)$ を用いて，式 (15) と示される．

$$a(c_{k-1}) \leq s(c_k) + t(c_k) \tag{13}$$

$$s(c_n) = t^r|C_r| \tag{14}$$

$$t = t^r|C_r| + \sum_{m=n}^{1} t(c_m) \tag{15}$$

式 (15) より，式 (13) を満たすならば，収集時間は，RTU が収集する FTU 数から計算することが可能となる．

次に，集約点の配置 A における RTU の収集する FTU 数 $|C_r|$ が決定された場合，式 (13) を満たす FTU 数の配分を求める．式 (13) を満たすならば，集約点 a_k の通信開始時間 $s(c_k)$ は式 (16) と表すことができる．これを式 (13) に代入すると，式 (17) が得られる．この式を式 (1) と式 (2) を用いて展開すると式 (18) となる．式 (18) より，$r(c_k)$ を求めると式 (19) が得られる．

$$s(c_k) = t^r|C_r| + \sum_{n}^{k+1} t(c_m) \tag{16}$$

$$a(c_k) \leq t^r|C_r| + \sum_{n}^{k+2} t(c_m) + t(c_{k+1}) \tag{17}$$

$$t^r \times r(c_k) \leq t^r|C_r| + \sum_{n}^{k+1} (p + d \times r(c_m)) \tag{18}$$

$$r(c_k) \leq |C_r| + (n-k) + \{\sum_{n}^{k+1} d(r(c_m) - 1)\}/t^r \tag{19}$$

```
1:  Integer tmp := |V^R|/(n+1)
2:  Array Â[],r[];
3:  while(true){
4:    total := tmp × 2,r[0] := tmp,r[n] := tmp;
5:    for(int k := n − 1;k ≥ 1;k − −){
6:      l := 0;
7:      for(int j := n;j > k;k − −){
8:        l := l + r[j];
9:        r[k] := r[0] + n − k + d × (l − 1)/t^r;
10:       total := total + r[k];
11:     }
12:   }
13:   if(total > |V^R|){
14:     Â := r
15:     tmp − −;
16:   }else{break;}
17: }
```

図 3: 式 (13) を満たす配置を求めるアルゴリズム

表 1: $|C_r|$ が 6 から 8 の際に式 (13) を満たす集約点が収集する端末数

$\|C_r\|$	$r(c_5)$	$r(c_4)$	$r(c_3)$	$r(c_2)$	$r(c_1)$	和
6	6	7	8	9	10	46
7	7	8	9	10	11	52
8	8	9	10	11	12	58

式 (19) で求められる集約点の端末数の配分は，図 3 のアルゴリズムを用いて，求めることができる．図 3 に現れる $\hat{A}$ は，$\hat{A}$=$(|C_r|, r(c_1), r(c_2), \ldots, r(c_n))$ である．

$p = 50$，$d = 1$，$n = 5$ の場合を例に挙げる．この場合に $\hat{A}$ として計算される値を表 1 に示す．表 1 の列 $|C_r|$ は RTU が収集する FTU 数を表しており，列 $r(c_5)$ から $r(c_1)$ は，式 (19) を満たす最大の数値を示している．また，和の列は，行の和を示す．この表は，$|C_r|$ が 6 から 8 まで求めている．図 3 では，1 行目において，$tmp = 50/(5+1)$ を行っており，tmp が 8 から始まり，表 1 の $|C_r|$ が 8 の場合を計算し，15 行目で tmp から 1 引いて $|C_r|$ が 7 の場合を計算，そののちに，$|C_r|$ が 6 の場合を計算した際に 16 行目で，ループを抜ける．その結果，得られる $\hat{A}$ は $|C_r|$ が 7 となる．$|V^R| = 50$ の場合，$|C_r| = 7$ の時の FTU 数の和が $|V^R|$ を超えるもっとも少ない数値となるため，$|C_r| = 7$ が式 (13) を満たす最短の収集時間となる．このように，FTU 数，p，d，n，を用いることにより，式 (13) を満たす場合の最短の収集時間となる FTU 数の分配を計算することができる．

3.1.4. 探索木の探索について

提案アルゴリズムでは，探索木を深さ優先探索で探索している．深さについては，収集時間の下限の見積もりを行い，見積もりした時間がその時点の最短時間より

遅くなる場合はそれ以上深く探索しないことにより，探索を打ち切る．例えば，$A = (a_1 = e_y, a_2 = e_z, \perp, \perp)$ のとき，$est(A) > t_{best}$ であるならば，$(a_1 = e_y, a_2 = e_z, *, *)$ の集約点の配置は，a_3 になにを当てはめても，$est(A) > t_{best}$ となるため探索されない．しかし，探索木の幅方向については，探索が続けられる．例えば，$A = (a_1 = e_y, a_2 = e_z, \perp, \perp)$ のとき，$est(A) > t_{best}$ であるならば，$e_x(x < z)$ が選択され，A は，$A = (a_1 = e_y, a_2 = e_x, \perp, \perp)$ となる．その次は，$e_w(w < x)$ が選択され，A は，$A = (a_1 = e_y, a_2 = e_w, \perp, \perp)$ となる．このように，a_2 に選択可能な e_i が存在する限り行われる．

そこで，幅方向にも探索を打ち切る条件を以下のように決定する．また，以下の条件を図 2 に加えた集約点の配置を決定するアルゴリズムを図 4 に示す．

条件 1

$k = 1$ かつ $t_{best} \leq est1(A)$ かつ $fit < x$ かつ $a_k = e_x$ であるならば，$a_k = e_y(x < y)$ の探索を行わない．

理由

e_i は，収集する FTU 数 $r(e_i)$ を元に昇順に並んでいるため，e_x と e_y に集約点を配置した際の収集する FTU 数は，$r(e_x) \leq r(e_y)$ となる．$a_k = e_y(x < y)$ に配置した場合の収集時間の見積 $est1(A)$ は，e_x に配置したとき以上であるため，$x < y$ の e_y については，探索を行わない．

条件 2

$k = 1$ かつ $t_{best} \leq est2(A)$ かつ $x \leq fit$ かつ $a_k = e_x$ であるならば，$a_k = e_y(y < x)$ の探索を行わない．

理由

$est2(A)$ の見積時間は RTU が収集する FTU を元に計算を行っている．e_i は，収集する FTU 数 $r(e_i)$ を元に昇順に並んでいるため，e_x と e_y に集約点を配置した際の収集する FTU 数は，$r(e_x) \geq r(e_y)$ となる．e_y に集約点を配置した場合，式 (11) における，集約点が収集する FTU 数の和が，e_x に配置した場合以下になる．つまり，RTU が収集する FTU 数 UA と式 (12) が，e_x に配置したとき以上となる．よって $est2(A)$ によって得られる見積時間は，e_x に配置した場合以上となるため，$a_k = e_y(y < x)$ の探索を行わない．

条件 3

$k \neq n$ かつ $t_{best} \leq est2(A)$ かつ $a_k = e_x$ であるならば，$a_k = e_y(y < x)$ の探索を行わない．

理由

条件 2 の場合と同じ理由である．$t_{best} \leq est2(A)$ となる $a_k = e_x$ を選択した場合，$a_k = e_y(y < x)$ の集約点の配置は，RTU が収集する FTU 数が $a_k = e_x$ のとき以上となり，$est2(A)$ は $a_k = e_x$ に配置したとき以上となる．よって，$a_k = e_y(y < x)$ の探索を行わない．

3.1.5. 提案法 1 の事例

本節では，提案法の事例について述べる．図 5 は，ある SCADA ネットワークを表すグラフである．FTU 数 $|V^R| = 13$，配置する集約点数 n は 3 とする．また，$d = 1$，$p = 20$ とする．図 4 のアルゴリズムを用いるにあたって，辺を $r(e_i)$ を元に昇順に並べる．選択対象となる全ての辺を列挙すると，e5(1)，e6(1)，e18(1)，e15(1)，e12(1)，e14(1)，e4(2)，e17(2)，e13(2)，e3(3)，e16(3)，e11(3)，e10(4)，e9(5)，e8(6)，e7(9)，e2(12)，e1(13)，である．括弧内は，そこに集約点を配置した場合に収集する FTU 数である．よって (e_1=e5,e_2=e6,... e_{13}=e1) である．次に，fit の値を求める．図 3 のアルゴリズムを用いて $\hat{A}$ を求めると，

$\hat{A} = (|C_r|, r(c_3), r(c_2), r(c_1)) = (3, 3, 4, 5)$

を得る．これより，$r(e_{fit}) = 5$ となる辺は e_{14}=e9 であるため fit=14 となる．収集時間 t_{best} の初期値は集約点配置を行わない場合の収集時間 273 である．これらの値を用いて，図 4 のアルゴリズムを実行すると，t_{best} を求めるための探索木は，図 6 に示すものになる．図 6 内の $(e, \perp, \perp)$ は集約点の配置候補 A を示しており，上部左側の数値は，$est2(A)$ の値，上部右側の数値は $est1(A)$ の値を示している．図 6 の (1) から (8) は，探索木のノードがどのように辿られているかを表している．(4) で得られる収集時間が最短の 133 である．探索の打ち切りは，(8)，(7) がそれぞれ打ち切り条件 1，条件 2 を満たしており，(6) が条件 3 を満たしている．

```
1:  A := (a_1 =⊥, a_2 =⊥, ..., a_n =⊥)
2:  t_best := t^r × |V^R|
3:  for(i := fit;i ≥ n;i − −){
4:    a_1 := e_i;
5:    if(t_best ≤ est2(A)){return;}
    //探索打ち切り条件 2
6:    else{selectAggregator(A,i − 1,2);}
7:  }
8:  for(i := fit + 1;i ≤ |E|;i + +){
9:    a_1 := e_i;
10:   if(t_best ≤ est1(A)){return;}
    //探索打ち切り条件 1
11:   else{selectAggregator(A,i − 1,2);}
12: }
13: function selectAggregator(A,l,k){
14: for(i := l;i ≥ k;i − −){
15:   a_k := e_i;
16:   if(t_best ≤ est2(A)){return;}
    //探索打ち切り条件 3
17:   else if(t_best ≤ est1(A)){continue;}
    //深さ方向の探索打ち切り
18:   else if(est1(A) < t_best&&est2(A) < t_best){
19:     if(k = n){
20:       t_tmp := calcTime(A);
21:       if(t_best > t_tmp){t_best := t_tmp;A_best := A;}
22:     }
23:     else{selectAggregator(A,i − 1,k + 1);}
24:   }
25: }
```

図 4: 探索打ち切りを含む集約点配置を決定するアルゴリズム

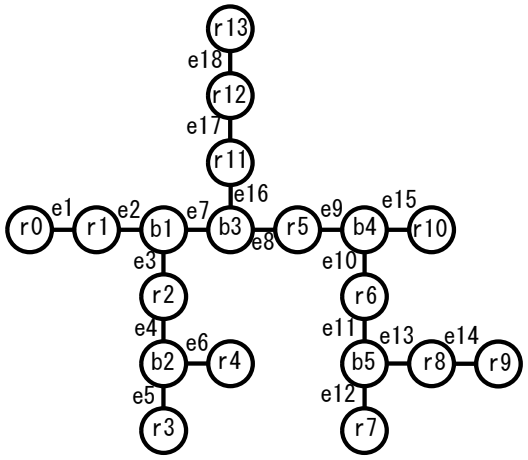

図 5: 事例に用いる SCADA ネットワークのグラフ

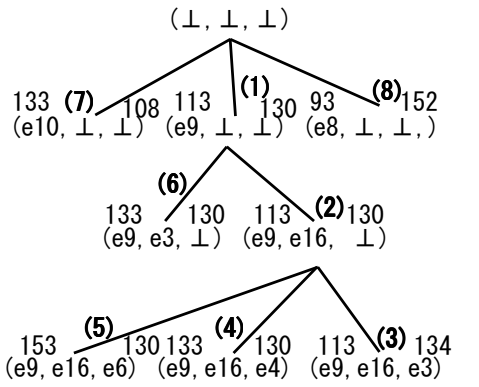

図 6: 図 5 に提案法 1 を適用した場合の探索木

3.2. 提案法 2 : ヒルクライム法を用いた集約点の配置

提案法 2 では，ヒルクライム法を用いて，収集時間を短くする集約点の階層的配置を許す配置場所を決定する．初期配置を求めるために利用される提案法 1 は，$A^n = NC(G, n)$ と表す．G は与えられた配電 SCADA ネットワークのグラフ，n は集約点数，A^n は集約点数 n の集約点の配置である．提案法 2 の入力として，グラフ G と集約点数 n が与えられ，出力は集約点の配置と収集時間である．手順は次の通りである．

1 ：$NC(G, n)$ を利用して，n 通りの集約点の初期配置 $A^{h1}, A^{h2} \ldots A^{hn}$ を求める．

2 ：n 通りの初期配置それぞれに対してヒルクライム法を行い，その最良の収集時間となる集約点の配置 A^{hk}_{best} を得る．

3 ：n 通りの初期配置から得られた A^{hk}_{best} の中から最も収集時間が短い A^{hk}_{best} と収集時間を出力する．

n 通りの初期配置，ヒルクライム法の手順については，以降に詳細を述べる．

ヒルクライム法の初期配置決定方法

n 回行われるヒルクライム法の初期配置 A^{h1} から A^{hn} を次の手順で求める．はじめに，$NC(G, n)$ を用いて A^1 から A^n を求める．次に，$A^m (1 \leq m < n)$ の集約点 $(a^m_1, a^m_2, \ldots a^m_m)$ それぞれの下流に配置する集約点数を割当てる．m 個配置済みであるので，$n - m$ 個の集約点数を均等に分配する．均等に分配できない場合は，収集する FTU 数が多い集約点に優先的に分配する．最後に，集約点 a^m_k の下流に分配された集約点数が l 個の場合，l 個の集約点の配置は，a^m_k を根とするサブグラフ G' と l を $NC(G', l)$ を用いて求める．全ての a^m_k に対して，$NC(G', l)$ を行い A^{hm} を求める．

ヒルクライム法の実行手順

それぞれの初期配置に対して，ヒルクライム法を実行する．ヒルクライム法は，現時点での解と近傍の解を比較し，近傍の解が良い場合，それを現時点の解と入れ替える．そして，また現時点の解と近傍の解を比較することを繰り返し，良い解が得られなくなれば探索を終了する．提案法 2 におけるヒルクライム法の手順は次の通りである．現時点の解を収集時間 T とし，その配置を A とする．近傍の解を T' とし，その配置を A' とする．

手順 1 ：集約点の配置 A の収集時間 T を計算する．

手順 2 ：近傍の配置 A' の収集時間 T' を計算する．

手順 3 ：$T \geq T'$ となる T' が最も小さい A' を A として手順 2 を行う．

手順 4 ：どの A' も $T < T'$ となる場合 A を出力する．また，$T \geq T'$ となる最小の T' の A' が，それまでの T の配置として用いられていた場合，同じ配置を繰り返すことになるため，ヒルクライム法を終了し A を出力する．

近傍の探索方法

近傍の解 T' を得る A' の配置は次の様に行う．集約点は，与えられたグラフの辺である．集約点として選択された辺 e に接続する頂点を i，j とする場合，辺 e を $e_{(i,j)}$ と表す．ヒルクライム法の実行手順 2 において，近傍となる A' の集約点 $e_{(i,j)}$ の変更場所は，頂点 i，j を端点とする辺 $e_{(a,i)}$，$e_{(j,b)}$ である．A' の集約点の変更場所は，同時に 1 つだけである．

4. 実験

本節では，提案法 1 を用いて集約点が階層的な配置を行わない場合の最短の収集時間となる集約点の配置場所を求める．提案法 1 を実行した際の，探索木の大きさ，計算時間，収集時間を比較する対象として，配置可能な集約点の組合せを全て網羅する総当たり方法を対象とする．また，ヒルクライム法を用いた提案法 2 を実行し，階層的な配置を行う場合の集約点の配置を求め，収集時間と計算時間を示す．実験環境は以下の通りである．

- OS CentOS6.5
- CPU Xeon E5-2665
- Memory 128GB
- 実装言語 JAVA

表 2: 集約点数に対する収集時間

FTU 数 127											
集約点数	0	1	2	3	4	5	6	7	8	9	10
階層配置あり	2667	1427	1384	827	780	730	680	547	502	496	462
階層配置なし	2667	1427	1384	867	780	730	687	680	567	536	525
FTU 数 164											
集約点数	0	1	2	3	4	5	6	7	8	9	10
階層配置あり	3444	2198	1340	1008	972	914	784	664	610	592	567
階層配置なし	3444	2198	1340	1008	972	914	900	884	878	878	878

表 3: 提案法 1 と総当たりの探索木の大きさ

FTU 数 127					
集約点数	1	2	3	4	5
総当たり	190	18145	990682	38199106	1103993223
提案法 1	4	5	12	96	34
FTU 数 164					
集約点数	1	2	3	4	5
総当たり	221	24531	1370921	49777045	1241937897
提案法 1	3	9	45	221	60

表 4: ヒルクライム法にかかる計算時間 [ms]

集約点数	2	3	4	5	6	7	8	9	10
FTU 数 127	1	1	1	2	3	5	6	33	266
FTU 数 164	1	2	4	5	34	126	266	519	936

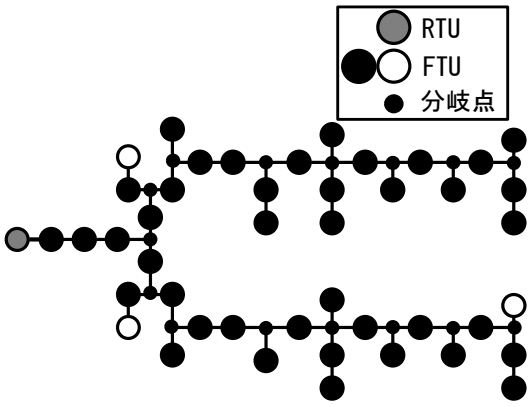

図 7: [6] に示される SCADA ネットワークトポロジ

通信にかかる時間については，FTU から収集する情報を通信する時間 d を 1 単位時間とし，通信にかかるオーバーヘッドの時間 p を 20 単位時間としている．実験には，２つのトポロジを用いた．FTU 数が 127 個のものは，2 分木となっている．また，FTU 数 164 個のものは，[6] に示される，配電 SCADA ネットワーク (図 7) を 4 つ接続したものを利用した．図 7 の 3 つの白丸に，RTU 直下の FTU を接続し FTU 数を 164 個としている．

表 5: 階層的配置を行わない場合の計算時間 [ms]

FTU 数 127					
集約点数	1	2	3	4	5
総当たり	3	104	341	12295	383718
提案法 1	3	3	3	13	3
FTU 数 164					
集約点数	1	2	3	4	5
総当たり	1 以下	13	572	21735	583322
提案法 1	2	3	4	14	4

実験結果：集約点数と収集時間

表 2 は，集約点の数に対する収集時間の変化を表している．FTU 数 164 の階層配置なしの集約点数 8 から 10 以外では，どちらのトポロジも，集約点数が増加すると収集時間の短縮が図れている．階層配置の有無について比較すると，集約点数 5 までは，収集時間にあまり変化は見られないが，集約点数 6 以上となると，階層配置がある場合の方が収集時間が短縮できていることがわかる．階層的配置を行うことで，収集時間は階層配置を行わない場合以下の値となっている．提案法 1 の集約点配置を元にヒルクライム法の初期配置を決定することにより，より収集時間が短くなる配置を求めることができた．

実験結果：探索木の大きさ

表 3 は，提案法 1 と総当たり方法を用いて，階層的配置を行わない場合の最短の収集時間を求める探索木のノード数を示している．提案法 1 では，集約点配置の途中において，収集時間の見積もりを用いた探索木の刈込が効果的に働いていると考えられる．これは，3.1.3 節において，最初に探索する a_1 を，式 (13) を満たす最短の FTU 数分配に基づいて，決定したことが寄与していると考えられる．

どちらのトポロジでも，集約点数が 4 のとき，他の場合と比べてノード数が増加している．FTU 数 164 の場合を例に挙げると，3.1.3 節の方法を用いて最初に決定した a_1 が収集する FTU 数が 38 である．FTU 数 164 で用いたトポロジーでは，収集する FTU 数が 38 となる集約点の配置場所が 3 か所，39 が 3 か所，40 が 3 か所，41 が 3 か所となっている．FTU 数のみを用いて A を記述すると，$(41, 40, 40, \perp)$ や $(41, 40, 39, \perp)$ などの A の見積もり時間がその時点の最短の時間より短くなるものが多く存在するため，探索木のノード数が増加したと考えられる．しかし，総当たりと比較すると，大きく削減されているため，最初に a_1 に選択される辺を適切に選択し，探索の打ち切りを行うことで，探索木のノード数を削減することができている．

実験結果：計算時間

表 4 は，ヒルクライム法を用いて，収集時間を短くする配置を求めるための計算時間を表している．表 5 は，階層的配置を行わない場合に最短の収集時間となる配置を求めるためにかかった計算時間を表している．表 4 の実行時間は，ヒルクライム法を実行するために必要な時間を表しており，初期値を求めるための時間は含んでいない．集約点数が多くなれば計算時間も増加している．これは，集約点数 n が増加すると，1 個から n-1 までの $NC(G, k)(1 \leq k < n)$ を求め，求められた配置それぞれに対して，改めて集約点の配置を行う回数が増加するためと考えられる．しかし，ヒルクライム法を用いた場合に集約点数が増加したとしても，表 5 の総当たりの計算時間と比べると，計算時間は十分に小さいといえる．

表 5 より，提案法 1 は総当たり方法と比較して，非常に短い時間で集約点配置場所を求めることができることがわかる．これは，表 3 に示す通り，探索木の刈込が効果的に働き，探索するノードが非常に少なくなっており，計算回数が削減されているためと考えられる．

5. まとめ

本論文では，SCADA ネットワーク内の FTU から情報を収集する時間を削減する方法とその配置場所を求める計算において，計算回数の削減について述べた．集約点と呼ばれる，情報を収集する端末を設置することにより，RTU と並行して情報を収集することが可能となり，収集時間の短縮を可能とした．集約点を配置した場合，RTU が直接通信を行う端末は，集約点と FTU である．RTU は，どのような順序で直接通信可能な端末から情報を収集すれば，最短の時間で情報を収集することが可能かを示した．提案法 1 において，集約点の階層的配置を行わない場合における収集時間が最短となる集約点配置場所の決定方法について提案した．それは，配置場所の探索木の探索において，集約点がすべて決定していない場合でも，収集時間の見積もりを行い，その見積もりによって探索を打ち切ることにより，探索木の探索する範囲を削減することにより，計算回数の削減を行った．提案法 1 を用いて，集約点の配置を求める実験を行い，階層的配置を行わない配置全てを総当たりする方法と比べ，探索木の探索範囲が削減されていることと，計算時間の削減できることを示した．

より収集時間を短くするために，ヒルクライム法の初期値に提案法 1 で求めた配置を用いて，集約点の階層的配置を行う場合の集約点の配置を求めた．その結果，階層的配置を行うことにより，より収集時間が短くなることを示した．

ヒルクライム法は，局所最適解を求める手法であるため，最短の収集時間であるとは言えない．初期配置により得られる収集時間に差がある可能性があるため，今後の研究では，より短い収集時間となる初期配置を求める必要がある．さらに，最短の収集時間を求める集約点の配置方法について検討を行うことも必要である．

参考文献

[1] 経済産業省，資源エネルギー庁，“太陽光発電システム等の普及動向に関する調査,” 参照 2015-04-05. $http://www.meti.go.jp/meti_lib/report/2013fy/E002502.pdf$.

[2] S.A. Boyer, Scada: Supervisory Control And Data Acquisition, 4th edition, International Society of Automation, USA, 2009.

[3] J. Luque, I. Gomez, and J.I. Escudero, “Determining the channel capacity in scada systems using polling protocols [power system telecontrol],” IEEE Transactions on Power Systems, vol.11, no.2, pp.917–922, May 1996.

[4] J.R. Jackson, “Scheduling a production line to minimize maximum tardiness,” Technical report, DTIC Document, 1955.

[5] E. Nowicki and S. Zdrzałka, “A survey of results for sequencing problems with controllable processing times,” Discrete Applied Mathematics, vol.26, no.2, pp.271–287, 1990.

[6] H. Terada, T. Onishi, and T. Tsuchiya, “A monitoring point selection approach for power distribution systems,” 2013 8th International Conference on System of Systems Engineering (SoSE), pp.190–195, June 2013.