解集合プログラミングを用いた配電網問題の解法

山田 健太郎 [1]，湊 真一 [2]，田村 直之 [3]，番原 睦則 [4]

[1] 名古屋大学大学院 情報学研究科
yken66@nagoya-u.jp

[2] 京都大学大学院 情報学研究科
minato@i.kyoto-u.ac.jp

[3] 神戸大学 情報基盤センター
tamura@kobe-u.ac.jp

[4] 名古屋大学大学院 情報学研究科
banbara@nagoya-u.jp

概 要

1 はじめに

2 配電網問題

2.1 配電網問題

本章では，本研究で扱う配電網問題を定式化する．以下の式において，ある配電区間 $i \in \{1, \ldots, n\}$ に対して，配電網構成 X が決まったとする．このとき，各区間 i に対して，変電所と反対側の区間 (木の下流) にある自身を含む区間の集合を C_i^{down} とする．また，各区間 i には，負荷 (電力需要) I_i が与えられる．

区間 i における電流の値 J_i は，次式で与えられる．

$$J_i = \sum_{j \in C_i^{down}} I_j. \tag{1}$$

配電網問題は，以下のように定式化される．

$$subject\ to \quad X \text{ is spanning rooted forest.} \tag{2}$$

$$J_i \leq J^{max},\ i \in \{1, \ldots, n\}. \tag{3}$$

制約 (2)，(3) をそれぞれ，**トポロジ制約**と**電流制約**と呼ぶ．これらの制約を満たす閉じたスイッチの組合せが存在するかどうかの判定問題としてここでは定式化する．なお，文献 [?] では，電圧に関する制約などを含んだ組合せ最適化問題として定式化されている．

次に例として，図 1 で示した配電網問題の解を図 2 に示す．この解は電流制約を $V_{max} = 300$ として求められた解である．閉じたスイッチは，$\{sw_1, sw_2, sw_4, sw_5, sw_7, sw_8, sw_9, sw_{10}, sw_{11}, sw_{12}, sw_{13}, sw_{15}\}$ である．このスイッチの集合から決まる配電経路は，図 2 の通り，トポロジ制約を満たしている．また，最も変電所に近い各区間 $\{s_a, s_b, s_c\}$ について，それぞれ，$s_a^{down} = \{s_a, s_1, s_2, s_3, s_4, s_5, s_{13}, s_{15}, s_{21}\}$，$s_b^{down}\{s_b, s_6, s_7, s_8, s_9, s_{10}, s_{11}, s_{12}, s_{14}, s_{16}, s_{22}\}$，$s_c^{down} = \{s_c, s_{17}, s_{18}, s_{19}, s_{20}\}$ である．したがって，各供給経路に対する電流値の合計は，$J_a = 162$，$J_b = 284$，$J_c = 71$ となり，電流制約も満たしている．

ここで，トポロジ制約を満たす配電網構成は，**根付き全域森**という部分グラフに対応することが知られている [?]．根付き全域森は以下のように定義される．

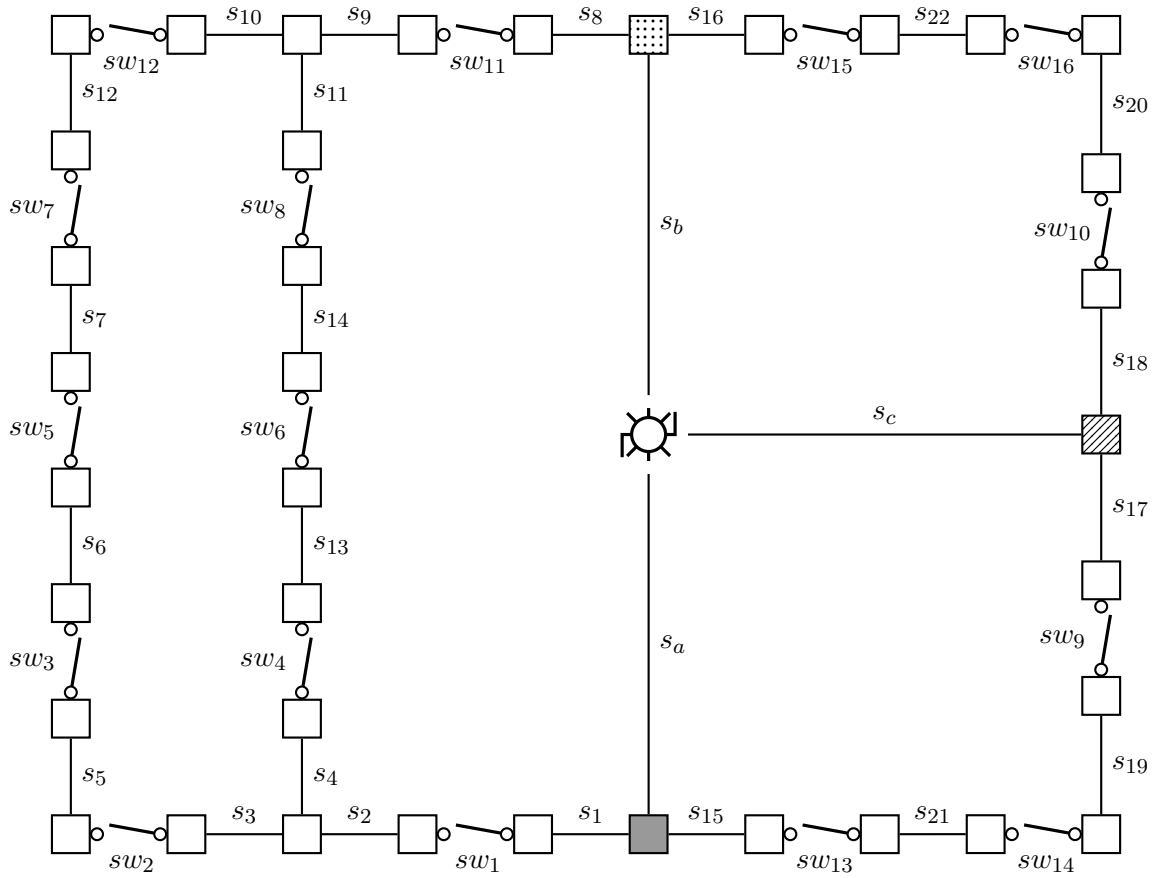

図 1. 配電網の例

定義. グラフ $G = (V, E)$ と，根と呼ばれる V 上のノードの集合が与えられたとする．このとき，G 上の根付き全域森とは，以下の制約を満たす G の部分グラフ $G' = (V, E'), E' \subseteq E$ である．

1. G' はサイクルを持たない．(非閉路制約)
2. G' の各連結成分は，ちょうど 1 つの根を含む．(根付き連結制約)

本稿では，与えられたグラフ G から，根付き全域森 G' を求める部分グラフ探索問題を**根付き全域森問題**と呼ぶ．

根付き全域森問題の入力例となるグラフを図 3 に示す．図 3 は，図 1 で示した配電網に対応しており，配電区間 $\{s_i \mid 1 \leq i \leq 22\}$ は，スイッチで区切られる 1 つのまとまりごとに**ノード**に対応する．例えば，区間 $\{s_2, s_3, s_4\}$ は，ノード 5 に対応している．スイッチ $\{sw_1, \ldots, sw_{16}\}$ は，**辺**に対応する．また，図中の色付きノード $\{r_1, r_2, r_3\}$ は変電所と直接繋がっている区間を含むことを意味しており，**根**に対応している．

根付き全域森の例を図 4 に示す．根付き全域森は，各連結成分が必ずちょうど 1 つの根をもつ木構造を形成することで，非閉路制約と根付き連結制約を満たす．図 4 は，図 2 の配電網問題の解に対応している．

2.2 配電網遷移問題

配電網の構成制御における障害時の復旧予測への応用を狙いとし，ある初期配電網構成 (スタート状態) から目的配電網構成 (ゴール状態) へのスイッチの切替手順を求める組合せ遷移問題を考える．各ステップ t と $t+1$ の間で**遷移制約**を満たしながら，もととなる配電網問題の実行可能解のみを経由し，最短ステップ長での切替手順を求めることが目的である．この組合せ遷移問題を**配電網**

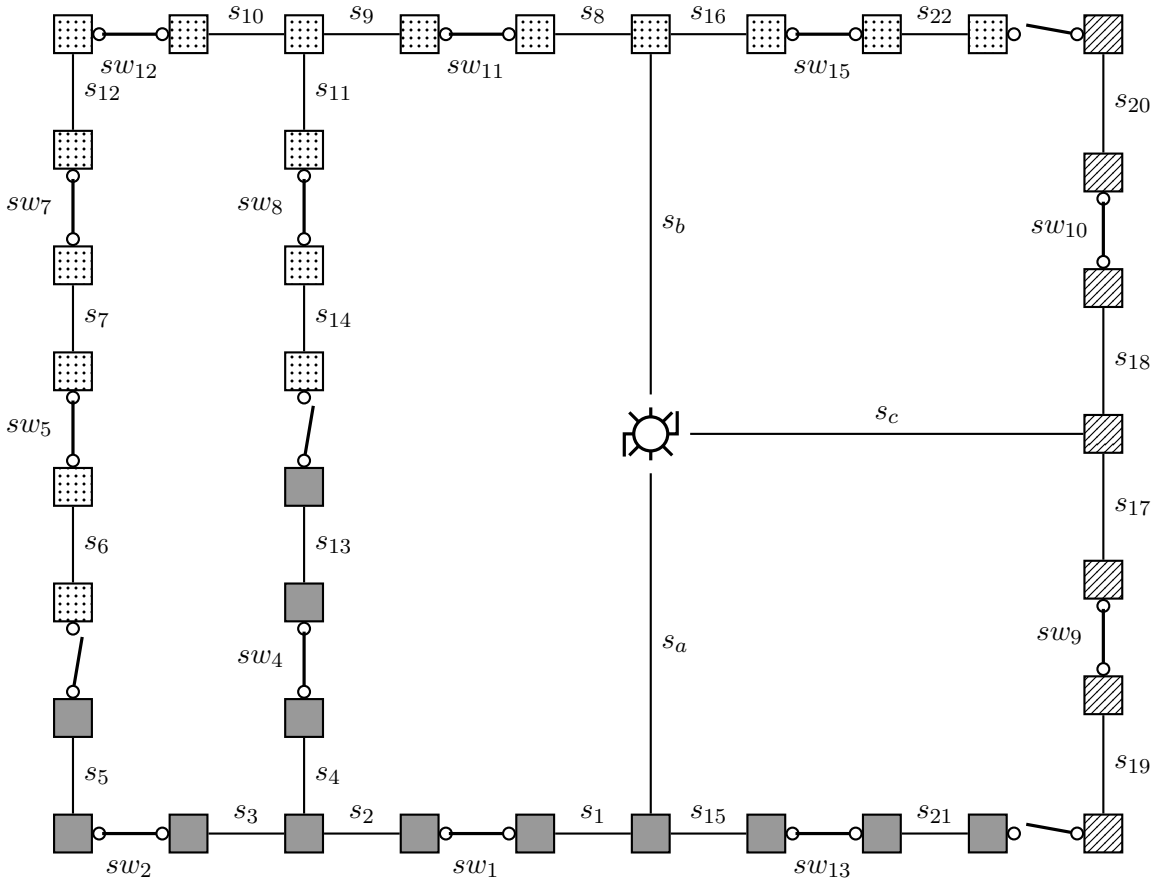

図 2. 解の一例

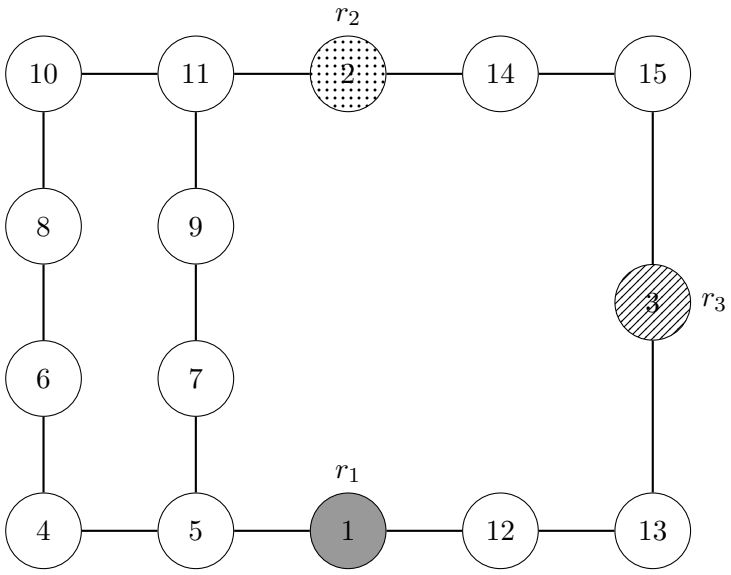

図 3. 根付き全域森問題の入力

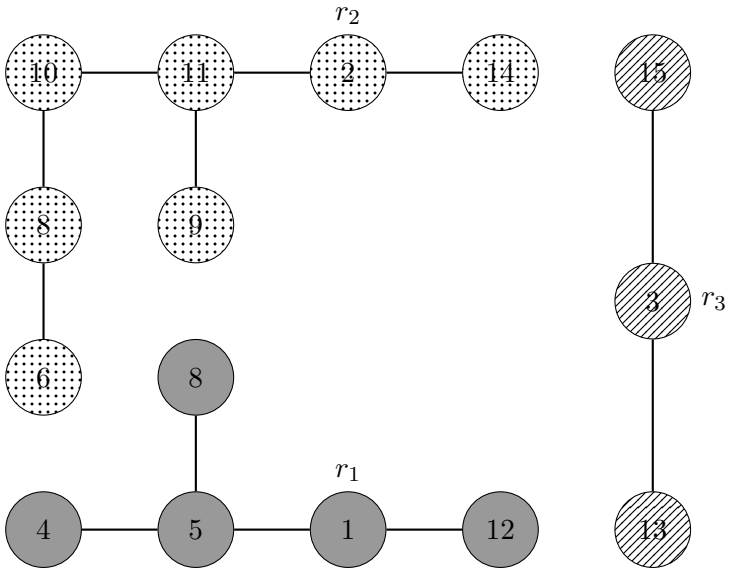

図 4. 根付き全域森問題の出力

遷移問題と呼ぶ．本研究では，遷移制約を各ステップ t で切替可能なスイッチの個数を 2 個に制限する．

図

3　解集合プログラミング

ASP の言語は，一般拡張選言プログラムをベースとしている [**?**]．本稿では，説明の簡略化のため，そのサブクラスである標準論理プログラムについて説明する．以降，標準論理プログラムを単に論理プログラムと呼ぶ．

論理プログラムは，以下の形式の**ルール**の有限集合である．

$$a_0 \texttt{ :- } a_1\texttt{ , . . . ,}a_m\texttt{ , not } a_{m+1}\texttt{ , . . . ,not } a_n\texttt{ .}$$

ここで，$0 \leq m \leq n$ であり，各 a_i はアトム，`not` は**デフォルトの否定**[1]，“`,`” は連言を表す．“`:-`” の左側を**ヘッド**，右側を**ボディ**と呼ぶ．“`.`” はルールの終わりを表す終端記号である．ルールの直観的な意味は,「$a_1, \ldots, a_m$ がすべて成り立ち，$a_{m+1}, \ldots, a_n$ のそれぞれが成り立たないならば，a_0 が成り立つ」である．ボディが空のルール (すなわち a_0 `:-` .) を**ファクト**と呼び，“`:-`” を省略することができる．

ヘッドが空のルールを**一貫性制約**と呼ぶ．

$$\texttt{:- } a_1\texttt{ , . . . ,}a_m\texttt{ , not } a_{m+1}\texttt{ , . . . ,not } a_n\texttt{ .}$$

例えば，一貫性制約 “`:-` a_1`,`a_2`.`” は,「a_1 と a_2 が両方同時に成り立つことはない」を意味し，“`:-` a_1`, not` a_2`.`” は,「a_1 が成り立つならば，a_2 も成り立つ」を意味する．

ASP 言語には，組合せ問題を解くために便利な拡張構文が用意されている．その代表的なものが**選択子**と**個数制約**である．例えば，選択子 “`{`a_1`;` . . . `;`a_n`}.`” をファクトとして書くと,「アトム集合 $\{a_1, \ldots, a_n\}$ の任意の部分集合が成り立つ」を意味する．個数制約は選択子の両端に選択可能な

[1] **失敗による否定**とも呼ばれる．述語論理で定義される否定 ($\neg$) とは意味が異なる．

コード 1. 配電網問題 (図 1) のファクト表現

個数の上下限を付けたものである．例えば，“`:-` a_0`,` `not` lb `{`a_1`;` $\ldots$ `;`a_n`}` ub.” と書くと,「a_0 が成り立つならば，$a_1, \ldots, a_n$ のうち，lb 個以上 ub 個以下が成り立つ」を意味する．また，**重み付き個数制約** (`#sum`) も用意されている．例えば，“`:-` a_0`,` `not` lb `#sum` `{`w_1`:`a_1`;` $\ldots$ `;`w_n`:`a_n`}` ub.” と書くと,「a_0 が成り立つならば，$a_1, \ldots, a_n$ のうち真となるアトムの重み和が lb 個以上 ub 個以下である」を意味する．項 w_i は重みを表す．

ASP システムは，与えられた論理プログラムから，安定モデル意味論 [**?**] に基づく解集合を計算するシステムである．本稿では，高性能かつ高機能な ASP システムとして世界中で広く使われている *clingo* [2]を使用する．*clingo* を含め最新の高速 ASP システムは，グラウンダーを用いて変数を含む論理プログラムを変数を含まない論理プログラムに変換 (**基礎化**) したのち，ASP ソルバーを用いて解集合を計算する方式が主流となっている．

4 配電網問題の ASP 符号化

本節では，配電網問題の入力と制約を論理プログラムとして表現する方法について述べる．

要相談

ASP ファクト形式. 配電網問題の入力 (図 1) のファクト表現をコード 1 に示す．この問題は，区間 25 個，スイッチ 16 個から構成されている．各区間とスイッチは，隣接関係をアトム `dnet_node/2` によって表される．例えば，ファクト `dnet_node(1,2).` は，ノード 1 とノード 2 が隣接していることを表す．

前処理. 配電網問題を解くための前処理となる ASP 符号化をコード**??**に示す．1 行目のルールは，スイッチノードの番号を表すアトム `swt_node(X)` を導入する．この `swt_node(X)` は，ノード番号 `X` はスイッチを含むことを意味する．反対に，2 行目のルールは，スイッチを含まないノード (ジャンクションノード) 番号を表すアトム `jct_node(X)` を導入する．この `jct_node(X)` は，ノード `X` には，スイッチが含まれないことを意味する．

4 行目のルールは，各区間がどのノード番号に属するかを表すアトム `section(S,X)` を導入する．また，5 行目で区間を意味するアトム `section(S)` を導入する．

7,8 行目のルールでは，各区間について，スイッチノードまたはジャンクションノードに含まれていることを表すアトム `swt_node(S)`,`jct_node(S)` を導入する．

10 行目のルールで，各スイッチがどの 2 つの区間を接続されているかを表すアトム `switch(SW,S,T)` を導入する．このアトム `switch(SW,S,T)` は，区間 `(S,T)` はスイッチ `SW` で接続されていることを意味する．

14 行目からのルールは根付き全域森問題の入力を生成するルールである．14,15 行目のルールは，根付き全域森問題のノードと区間を対応させるアトム `node/2` を導入する．アトム `node(X,S)` は，各区間 `S` は根付き全域森のノード `X` に含まれることを意味する．14 行目のルールで，ジャンクションノードはそのまま根付き全域森のノードとして定義され，ジャンクションノードに属する区間は，そのまま対応づけられる．15 行目のルールでは，スイッチノードに含まれ，ジャンクションノー

[2] `https://potassco.org/`

コード 2. 根付き全域森問題の基本符号化

コード 3. 根付き全域森問題の改良符号化

ドに含まれない各区間について，属するノード番号のうち，小さい方の番号を根付き全域森問題のノードとして定義する．

18 行目は，根ノードを表すアトム `root(X)` を導入する．この `root(X)` は，`node(X,S)` のうち，区間 `S` が変電所と直接つながっているならば，根ノードであることを意味する．また，19 行目で各 `node/2` を，根付き全域森問題の入力のノードを表すアトム `node/1` とする．20 行目で，辺を表すアトム `edge(X,Y)` を導入する．この `edge(X,Y)` は任意の 2 つのノード `(X,Y)` が辺で結ばれていることを意味する．

4.1 根付き全域森問題の ASP 符号化

基本符号化. 根付き全域森問題の ASP 符号化をコード 2 に示す．2 行目のルールは，各辺 (X,Y) に対して，解の候補となるアトム `inForest(X,Y)` を導入する．この `inForest(X,Y)` は，辺 (X,Y) が根付き全域森に含まれることを意味する．各ノードの到達可能性は 5〜7 行目のルールで表される．アトム `reached(X,R)` は，ノード `X` は根 `R` から到達可能であることを意味する．5 行目のルールは，各根ノードは自分自身から到達可能であることを表す．6〜7 行目のルールは，ノード `Y` が根ノード `R` から到達可能であり，かつ，辺 (X,Y) が全域森に含まれるならば，ノード `X` も `R` から到達可能であることを表す．

非閉路制約は 10〜13 行目のルールで表される．このルールは，各連結成分のノード数と辺数の差が 1 になること (木の性質) を，ASP の重み付き個数制約を使って表している．根付き連結制約は 16〜17 行目のルールで表される．16 行目のルールは，各ノードは少なくとも 1 つの根から到達可能であることを表している (*at-least-one* 制約)．17 行目のルールは，各ノードは高々1 つの根から到達可能であることを表している (*at-most-one* 制約)．これら 2 つの一貫性制約により，各ノードはちょうど 1 つの根から到達可能であることが強制される．

改良符号化. 基本符号化は，根付き全域森問題の制約を ASP のルール 7 個で簡潔に表現できる．しかし，根付き連結制約を表す *at-most-one* 制約の基礎化後のルール数は，根ノード数の 2 乗に比例するため，大規模な問題に対する求解性能が低下する可能性がある．この問題を解決するために考案した改良符号化をコード 3 に示す．基本符号化との違いは，根付き連結制約を ASP の個数制約で表している点である (16 行目)．グラフのノード数を n，根ノード数を r として，根付き連結制約の基礎化後のルール数を比較すると，基本符号化が $n(1+{}_rC_2)$ 個なのに対し，改良符号化は n 個と少なく抑えることができる．これにより，大規模な問題に対する有効性が期待できる．

発展符号化. 非閉路制約を別の方法で符号化するために，根付き全域森問題を有向グラフに拡張し，各ノードの入次数に関する制約を用いて符号化を行った．考案した発展符号化をコード 4 に示す．9 行目のルールでは，各根ノードについて，入次数は 0 でなければならないことを表している．10 行目のルールでは，根ノード以外の各ノードについて，入次数が 1 であることを表している．その他のルールは改良符号化と同じである．

コード 4. 根付き全域森問題の発展符号化

コード 5. 電流制約の符号化

4.2 電流制約の ASP 符号化

電流制約の ASP 符号化をコード 5 に示す．3 行目のルールで，根付き全域森問題の解となるアトム `inForest(X,Y)` をもとに，配電網において，スイッチが閉じて区間が接続されていることを表すアトム `connected(SW,S,T)` を導入する．このアトムは，区間 S からスイッチ SW によって，もう 1 つの区間 T に接続していることを表す．6 行目のルールで，配電網問題の解を表すアトム `closed_switch(SW)` を導入する．この `closed_switch(SW)` は，スイッチ SW が閉じたスイッチであることを意味する．

ジャンクションノード内での上下流の関係は，8,9 行目のルールで表される．アトム `entrance_section(S,X)` は，区間 S がノード X 内で最も上流にあることを意味する．8 行目のルールで変電所と直接つながっている区間が上流にあることを表している．9 行目のルールで，ジャンクションノード内の区間が，スイッチによって外部のノードから接続されているならば上流であることを意味している．

また供給経路内の上下関係は，アトム `downstream(S,T)` で表される．この `downstream(S,T)` は，区間 S の下流に区間 T が接続されていることを意味する．12 行目のルールで，接続されている 2 つの区間 `(S,T)` がそのまま上下関係であることを意味している．13 行目のルールでは，`entrance_section(S,X)` が属するノード X 内にある他の区間 T が下流にあることを意味している．

15,16 行目は，変電所と区間の供給関係を表している．アトム `suppliable(T,R)` は，区間 T が変電所 R から供給を受けていることを意味する．15 行目のルールでは，各変電所は自身から供給を受けることを表す．16 行目のルールでは，区間 S が変電所 R から供給され，かつ，区間 T が S の下流にあるならば，T も R から供給されることを表す．

18 行目のルールで，各変電所 R について，供給する各区間 S の負荷 I の総和が入力として与えられる `max_current` 以下でなければならないことを表す．

5 配電網遷移問題の ASP 符号化

6 実験