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[1]

1. 名古屋大学 大学院情報学研究科
2. 京都大学 大学院情報学研究科
3. 神戸大学 情報基盤センター

第 24 回プログラミングおよびプログラミング言語ワークショップ

配電網問題

求解困難な組合せ最適化問題の一種

- **配電網**とは，変電所と，一般家庭や工場を繋ぐ電力供給経路のネットワークである．
- 配電網の構成技術はスマートグリッドや，災害時の停電復旧などを支える重要な基盤技術として期待されている．
- **配電網問題**とは，
  - **トポロジ制約**と**電気制約**を満たしつつ，
  - 損失電力を最小にするスイッチの開閉状態を求めることが目的．
- これまで，メタヒューリスティクス等の解法が提案されている．
- 厳密解法として，フロンティア法を用いた解法が提案されている．
  - 実用規模の配電網問題 (**fukui-tepco，スイッチ数 468 個**) の最適解を求めることに成功 [井上ほか '12]．

配電網遷移問題

配電網遷移問題

配電網問題とその２つの実行可能解が与えられたとき，一方の解から他方の解へ，**遷移制約**を満たしつつ，実行可能解のみを経由して到達できるかどうかを判定する問題．

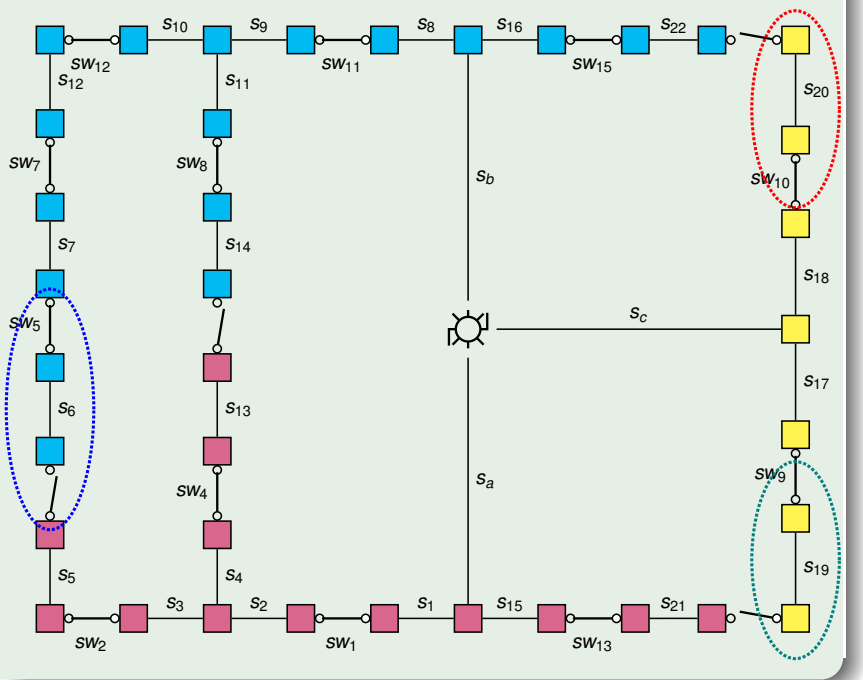
- 各ステップ t で変更可能なスイッチを d 個に制限．(**遷移制約**)
- 本研究では，到達可能であればその最短経路を求めることが目的．

- 配電網の構成制御における災害時の停電復旧などへの応用が狙い．
- 近年，理論計算機科学の分野を中心に急速に発展している**組合せ遷移問題**の一種．

しかし現状では，配電網遷移問題を効率的に解くソルバーの
実装技術は確立されていない．

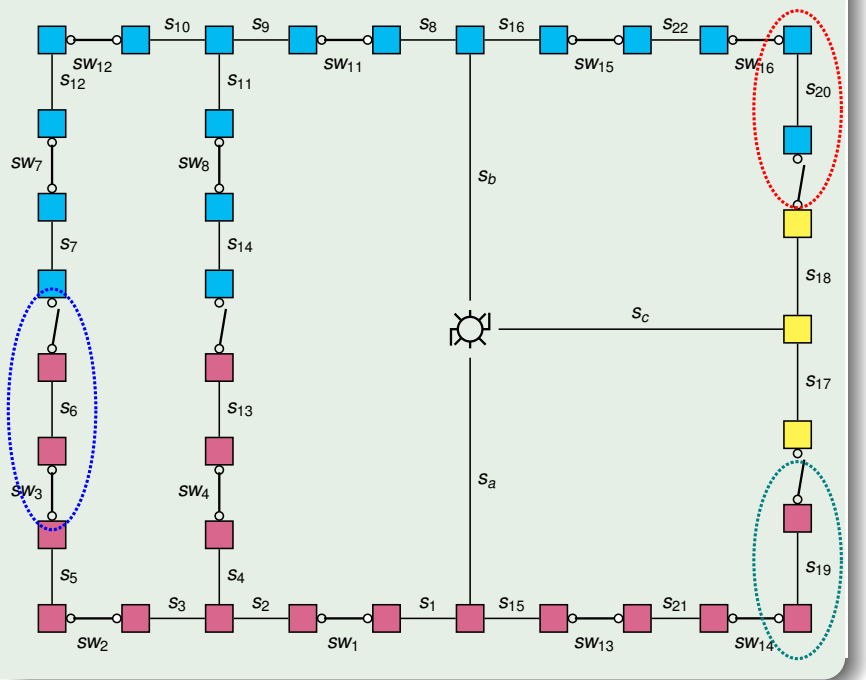
配電網遷移問題の例

スタート状態



$\Rightarrow$

ゴール状態



- セクション数：25 個，スイッチ数：16 個，変電所数：3 個 †
- 各ステップで変更可能なスイッチを 2 個に制限．(**遷移制約**)
- スタート状態からゴール状態へ **3 ステップで到達可能**．

†変電所に直接つながるセクションの数

解集合プログラミング (Answer Set Programming; ASP)

- **ASP の言語**は一階論理に基づく知識表現言語の一種である．
- **ASP システム**は論理プログラムから安定モデル意味論 [Gelfond and Lifschitz '88] に基づく解集合を計算するシステムである．
- 近年，SAT ソルバーの実装技術を応用した高速 ASP システムが実現され，システム検証，プランニング，システム生物学など様々な分野への応用が拡大している．

配電網遷移問題に対して ASP 技術を用いる利点

- ASP 言語の高い表現力を活かし，組合せ問題を**簡潔に記述可能**
  - **組合せ遷移問題への拡張も容易**
- マルチショット ASP 解法により，ステップ長を増やしながら，組合せ遷移問題の**到達可能性を効率的に検査可能**
  - ASP システムを複数回起動するオーバヘッドを回避可能
  - 同様の探索失敗を避けるために獲得した学習節を再利用可能

研究目的

目的

ASP 技術を活用した大規模な配電網遷移問題を効率良く解くシステムを構築する．

研究内容

1. **配電網問題の ASP 符号化の考案**
   - トポロジ制約の ASP 符号化として，**基本符号化**，**改良符号化**，**有向符号化** の 3 種類を考案．
   - 電気制約として，電流制約の ASP 符号化を考案．
2. **配電網遷移問題の ASP 符号化の考案**
   - **シングルショット符号化**と**マルチショット符号化**の 2 種類を考案．
   - 配電網問題の ASP 符号化の自然な拡張である．
3. **実用規模の問題を含むベンチマークによる評価実験**
   - 実用規模の問題である fukui-tepco をもとに，配電網遷移問題ベンチマークを 1000 問作成．
   - **マルチショット符号化**は，**約 3.8 倍の高速化**を実現．

配電網問題

配電網問題のトポロジ制約

トポロジ制約を満たす配電網構成は，グラフと根と呼ばれる特別なノードから，**根付き全域森**を求める部分グラフ探索問題に帰着できる．

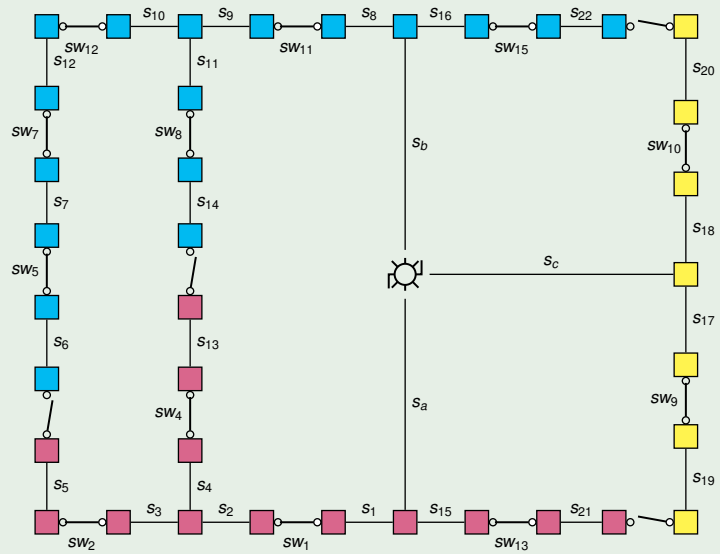
根付き全域森 (Spanning Rooted Forest) [川原・湊 '12]

グラフ $G = (V, E)$ と，**根**と呼ばれる V 上のノードが与えられたとき，G 上の根付き全域森とは，以下の条件を満たす G の部分グラフ $G' = (V, E')$, $E' \subseteq E$ である．

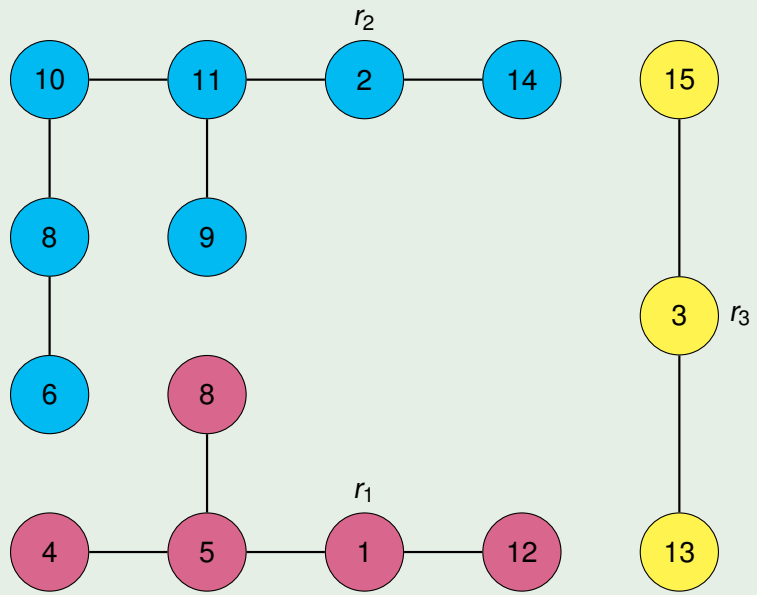
1. G' はサイクルを持たない． (**非閉路制約**)
2. G' の各連結成分は，ちょうど 1 つの根を含む． (**根付き連結制約**)

配電網問題のトポロジ制約

配電網問題の解



根付き全域森



- **停電** (変電所と結ばれないセクション)
- **短絡** (供給経路上のループ，複数の変電所と結ばれるセクション)
- **配電網とグラフの対応**

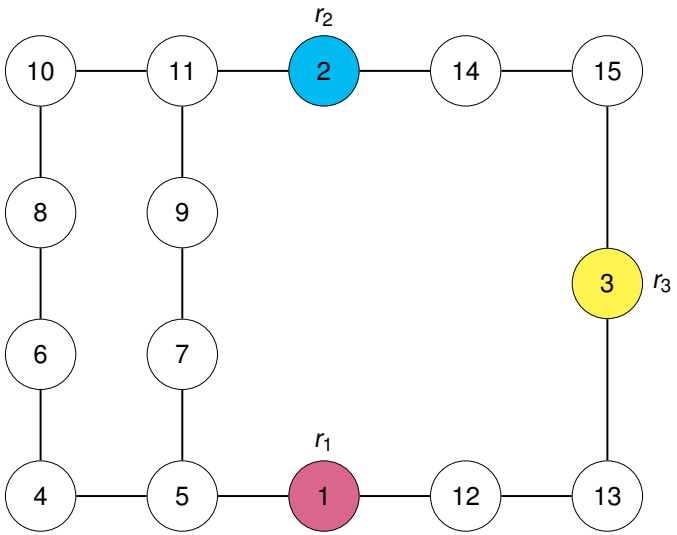
配電網	セクション	スイッチ	変電所
グラフ	ノード	辺	根

トポロジ制約の ASP 符号化

トポロジ制約に関して，
基本符号化，改良符号化，有向符号化の 3 種類の ASP 符号化を考案

- **基本符号化**は，根付き連結制約を *at-least-one* 制約と，*at-most-one* 制約を用いて表現する基本的な符号化である．
- **改良符号化**は，根付き連結制約を ASP の個数制約を用いることで，基礎化後のルール数を少なく抑えるように工夫した符号化である．
- **有向符号化**は，無向グラフの各辺 $u - v$ に対して，2 つの弧 $u \to v$ と $v \to u$ を対応させることで有向グラフ化して解く符号化であり，非閉路制約をノードの入次数の制約で簡潔に表現できる．

グラフ表現の ASP ファクト形式




```
node(1..15).

edge(1,5). edge(1,12). edge(2,11). edge(2,14).
edge(3,15). edge(3,13). edge(4,5). edge(4,6).
edge(5,7). edge(6,8). edge(7,9). edge(8,10).
edge(9,11). edge(10,11). edge(12,13).

root(1). root(2). root(3).
```

有向符号化の ASP コード

```
(1) { inForest(X,Y); inForest(Y,X) } 1 :- edge(X,Y).

(2) :- root(R), inForest(_,R).
(3) :- node(X), not root(X), not 1 { inForest(_,X) } 1.

(4) reached(R,R) :- root(R).
(5) reached(X,R) :- reached(Y,R), inForest(Y,X).

(6) :- node(X), not 1 { reached(X,R) } 1.
```

- (1) のルールで，与えられた無向グラフを有向グラフ化する．
- アトム 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．

有向符号化の ASP コード

```
(1) { inForest(X,Y); inForest(Y,X) } 1 :- edge(X,Y).

(2) :- root(R), inForest(_,R).
(3) :- node(X), not root(X), not 1 { inForest(_,X) } 1.

(4) reached(R,R) :- root(R).
(5) reached(X,R) :- reached(Y,R), inForest(Y,X).

(6) :- node(X), not 1 { reached(X,R) } 1.
```

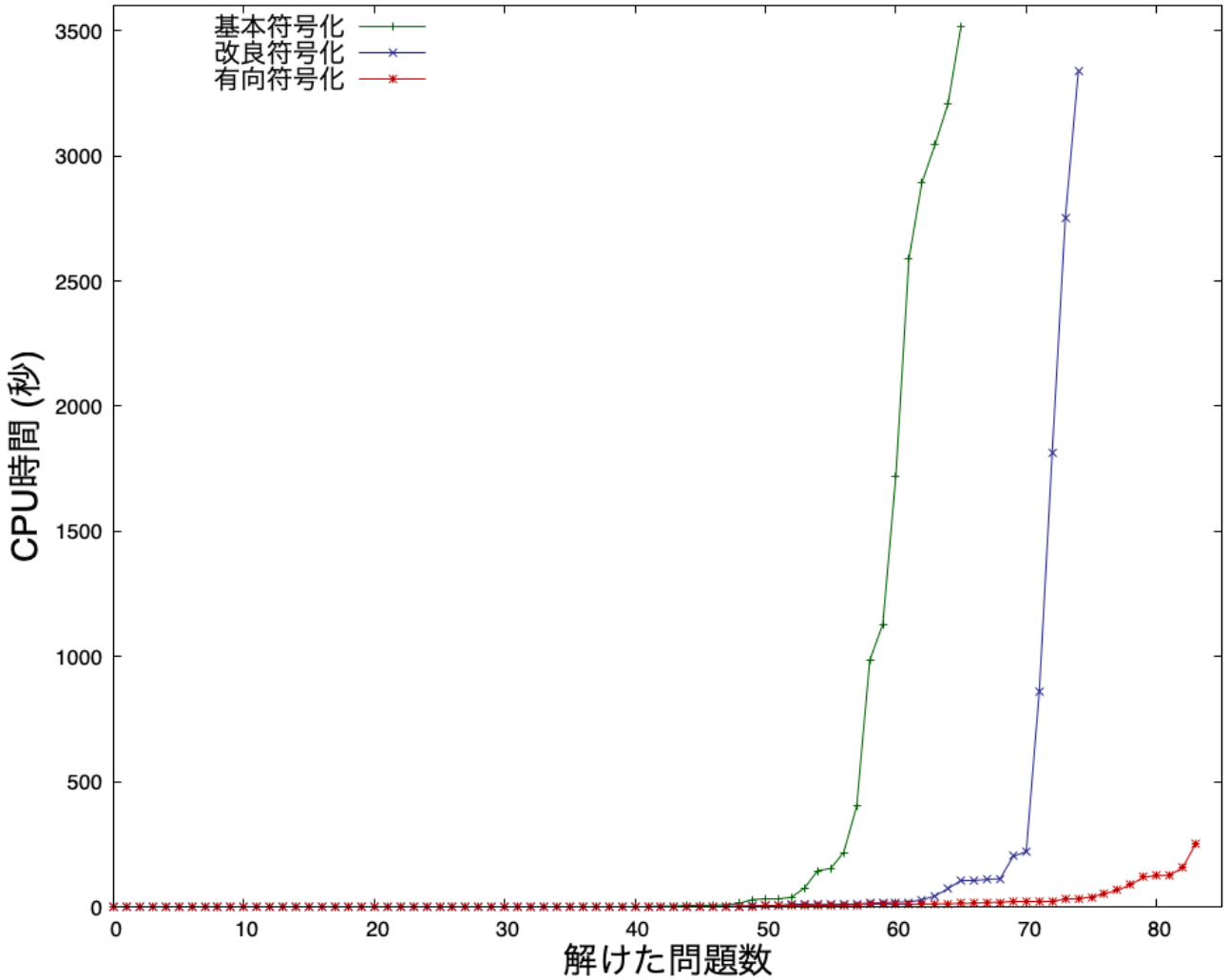
- (2)–(3) のルールは，非閉路制約を表す．
- (2) は，各根 R について，入次数が 0 であることを表す．
- (3) は，根ではない各ノード X について，入次数がちょうど 1 であることを表す．

有向符号化の ASP コード

```
(1) { inForest(X,Y); inForest(Y,X) } 1 :- edge(X,Y).

(2) :- root(R), inForest(_,R).
(3) :- node(X), not root(X), not 1 { inForest(_,X) } 1.

(4) reached(R,R) :- root(R).
(5) reached(X,R) :- reached(Y,R), inForest(Y,X).

(6) :- node(X), not 1 { reached(X,R) } 1.
```

- (4)–(5) は，到達可能性を表す．
- アトム reached(X,R) は，ノード X が根 R から到達可能であることを意味する．
- (6) は，根付き連結制約を表す．

実験結果：カクタスプロット



- 有向符号化は，他の符号化と比較して，より多くの問題 (84/85 問) を高速に解いている．

配電網遷移問題

配電網遷移問題の定式化

- 配電網問題の変数集合 $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$ に対して，ステップ $t \geq 0$ での各変数の値を表す変数集合 $\boldsymbol{x}^t = \{x_1^t, x_2^t, \ldots, x_n^t\}$ を導入．
- スタート状態から ℓ ステップ遷移した後の各変数の値 $\boldsymbol{x}^\ell$ が，ゴール状態を満足するかを判定するため，論理式 φ_ℓ を構成する．

$$\varphi_\ell = S(\boldsymbol{x}^0) \qquad S：スタート状態を表す論理式$$

$$\wedge \bigwedge_{t=0}^{\ell} C(\boldsymbol{x}^t) \qquad C：トポロジ制約，電流制約を表す論理式$$

$$\wedge \bigwedge_{t=1}^{\ell} T(\boldsymbol{x}^{t-1}, \boldsymbol{x}^t) \qquad T：遷移制約を表す論理式$$

$$\wedge G(\boldsymbol{x}^\ell) \qquad G：ゴール状態を表す論理式$$

- φ_ℓ が充足可能の場合，ステップ長 ℓ の到達可能な遷移系列が存在することを意味する．

ASP を用いた配電網遷移問題の解法

配電網遷移問題に対して，制限された長さの遷移系列

$$\varphi_\ell = S(\boldsymbol{x}^0) \wedge \bigwedge_{t=0}^{\ell} C(\boldsymbol{x}^t) \wedge \bigwedge_{t=1}^{\ell} T(\boldsymbol{x}^{t-1}, \boldsymbol{x}^t) \wedge G(\boldsymbol{x}^\ell)$$

を論理プログラムとして表現し，ASP システムを用いて
実行することにより，到達可能性の検査を行う．

- φ_ℓ が**充足可能**の場合，ステップ長 ℓ の**到達可能**な遷移系列が存在．
- φ_ℓ が**充足不能**の場合，ステップ長 ℓ では**到達不能**．
- 充足不能の場合，ℓ を増加させた論理プログラムを再構成し，繰り返し ASP システムを実行．

配電網遷移問題の ASP 符号化

シングルショット符号化

- φ_ℓ をそのまま 1 つの論理プログラムとして記述.
- 配電網問題の ASP 符号化の自然な拡張.

- ステップ長 ℓ を増加させながら，φ_ℓ を繰り返し構成し解く.
- 短所: 学習節が再利用できない.
- 短所: ASP システムを毎回起動するオーバーヘッドが大きい.

マルチショット符号化

- φ_ℓ を，$S(\boldsymbol{x}^0)$ を表す `base` 部，$C(\boldsymbol{x}^t), T(\boldsymbol{x}^{t-1}, \boldsymbol{x}^t)$ を表す `step(t)` 部，$G(\boldsymbol{x}^t)$ を表す `check(t)` 部に分けて記述.

- φ_ℓ をインクリメンタルに構成しながら解くことが可能.
- 長所: 学習節の再利用が可能．ASP システムの起動は 1 回のみ.
- 短所: 現状では，デバックしにくい.

実験概要

提案する ASP 符号化の性能の評価実験を行った．

- **比較する ASP 符号化:**
  - シングルショット符号化
  - マルチショット符号化
- **ベンチマーク問題:** 全 1000 問
  - DNET † で公開されている実用規模の配電網問題 (**fukui-tepco**，スイッチ数 468，変電所の数 72，許容電流 300A) をベース
  - 実行可能解の中から，スタート状態を 10 個，ゴール状態を 100 個をランダムに抽出し，それらを組み合わせて生成
- **ASP システム:** *clingo-5.4.0* + *trendy*
- **制限時間:** 10 分/問
- **実験環境:** Mac mini，3.2GHz Intel Core i7，64GB メモリ

†https://github.com/takemaru/dnet

実験結果：平均CPU時間の比較

最短ステップ長	問題数	シングルショット	マルチショット	シングル/マルチ
1	6	1.677	1.035	1.620
2	62	3.507	1.608	2.180
3	189	6.089	2.155	2.826
4	312	9.294	2.734	3.399
5	280	13.338	3.361	3.968
6	130	18.303	4.165	4.394
7	21	24.483	5.086	4.814
計	1000	76.691	20.114	3.807

- 1000 問全ての到達可能性を判定でき，全て到達可能であった．
- 今回生成した問題のうち，最長で最短ステップ数は 7 であった．
- マルチショットは，シングルショットと比較して，全ての問題をより高速に解いており，**平均で 3.8 倍の高速化**を実現している．

まとめと今後の課題

配電網遷移問題に対して，ASP を用いた解法を提案した．

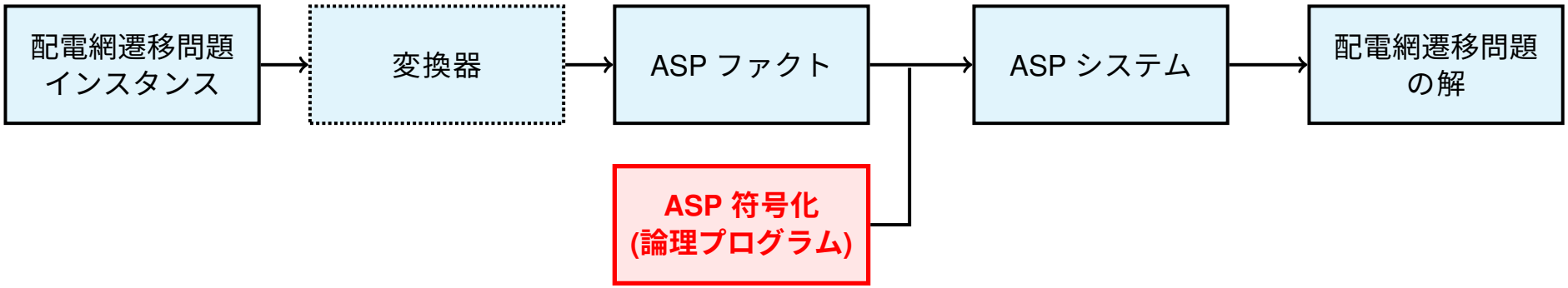
1. **配電網問題の ASP 符号化の考案**
   - トポロジ制約の ASP 符号化として，**基本符号化**，**改良符号化**，**有向符号化** の 3 種類を考案．
   - 電気制約として，電流制約の ASP 符号化を考案．
2. **配電網遷移問題の ASP 符号化の考案**
   - **シングルショット符号化**と**マルチショット符号化**の 2 種類を考案．
   - 配電網問題の ASP 符号化の自然な拡張である．
3. **実用規模の問題を含むベンチマークによる評価実験**
   - 実用規模の問題である fukui-tepco をもとに，配電網遷移問題ベンチマークを 1000 問作成．
   - **マルチショット符号化**は，**約 3.8 倍の高速化**を実現．

今後の課題

- 電流制約だけでなく電圧制約も含む配電網遷移問題への拡張
- 完全な問題は非線形な制約を含むため，ASP Modulo Theories を用いた解法を検討

- 補足用 -

ASP を用いた配電網遷移問題の解法の流れ



1. 問題インスタンスを ASP のファクト形式に変換する.
2. ASP ファクトと配電網遷移問題を解く ASP 符号化を入力として, ASP システムを用いて解集合を計算する.
3. 解集合を解釈して配電網遷移問題の解を得る.

ASP の構文

ASP の言語は論理プログラムをベースとしている [†].

- **論理プログラム**とは，以下の**ルール**の有限集合である．

$$\underbrace{a_0}_{\text{ヘッド}} \quad :- \quad \underbrace{a_1, \ldots, a_m, \texttt{not}\ a_{m+1}, \ldots, \texttt{not}\ a_n}_{\text{ボディ}}.$$

  $0 \leq m \leq n$ であり，各 a_i はアトム，not は**デフォルトの否定**，“,” は連言 (AND) を表す．
- **直感的な意味**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．

[†]本発表では標準論理プログラムを単に論理プログラムと呼ぶ．

ASP の構文

- ボディが空のルールを**ファクト**と呼び，“:-” は省略できる．

$$\underbrace{a_0}_{\text{ヘッド}}\ .$$

- ヘッドが空のルールを**一貫性制約**と呼ぶ．

$$\texttt{:-}\quad \underbrace{a_1\texttt{,}\ldots\texttt{,}a_m\texttt{, not }a_{m+1}\texttt{,}\ldots\texttt{,not }a_n\texttt{.}}_{\text{ボディ}}$$

ボディのリテラル (a_i あるいは $\texttt{not}\ a_i$) の連言 (AND) が成り立たないことを表す．

ASP の拡張構文

組合せ問題を解くための便利な構文が用意されている．

- **選択子**

$$\{a_1 ; \ldots ; a_n\}$$

アトム集合 $\{a_1, \ldots, a_n\}$ の任意の部分集合が成り立つことを意味する．

- **個数制約**

$$lb\ \{a_1 ; \ldots ; a_n\}\ ub$$

$a_1, \ldots, a_n$ のうち，lb 個以上，ub 個以下が成り立つことを意味する．

実験概要

- **比較する ASP 符号化:**
  - 基本符号化
  - 改良符号化
  - 有向符号化
- **ベンチマーク問題:** 全 85 問
  - DNET[†] で公開されている配電網問題 3 問
    (トポロジ制約のみ，スイッチ数: 16 個，36 個，468 個)
  - *Graph Coloring and its Generalizations* [‡] で公開されている
    グラフ彩色問題をベースに，独自に生成した 82 問 [§]
    ($20 \leq$ 辺数 $\leq 49{,}629$)
- **ASP システム:** *clingo-5.4.0* + *trendy*
- **制限時間:** 3600 秒/問
- **実験環境:** Mac mini，3.2GHz Intel Core i7，64GB メモリ

[†]https://github.com/takemaru/dnet
[‡]https://mat.tepper.cmu.edu/COLOR04/
[§]各問題に対し，全ノードのうち 1/5 個をランダムに変電所として与えた．

実験結果: 解けた問題数による比較

辺の範囲	問題数	基本符号化	改良符号化	有向符号化
1 ～ 1,000	30	30	30	30
1,001 ～ 4,000	20	20	20	20
4,001 ～ 7,000	11	9	10	11
7,001 ～ 10,000	8	4	6	7
10,001 ～ 20,000	9	2	5	9
20,001 ～ 30,000	2	1	2	2
30,001 ～ 40,000	1	0	0	1
40,001 ～ 50,000	4	0	2	4
計	85	66	75	84

- 有向符号化は，ベンチマーク問題 85 問中，**84 問**を解いている．
- 大規模な問題に対しても有向符号化は，優位性を示した．
- 有向符号化で解けなかったグラフは，*leighton graph* と呼ばれるグラフから生成したものであった．

電流制約

電流制約

$$J_i = \sum_{j \in S_i^{down}} I_j, \quad J_i \leq J^{max} \quad (\forall s_i \in S)$$

S	セクションの集合
s_i^{down}	セクション i より下流にあるセクション
I_i	セクション i の負荷電流
J_i	セクション i に流れる電流
J^{max}	電流の許容範囲 (入力)

電流の計算例

- 電流が許容範囲を超えると電線が焼き切れる事故につながる.

電圧制約

電圧制約

$$V_i = V_0 - \sum_{s_j \in S_i^{up} \cup \{s_i\}} Z_j \left[\sum_{s_k \in S_j^{down}} I_k + \frac{I_j}{2} \right], \quad V_i, \geq V^{min} \quad (\forall s_i \in S)$$

s_i^{up}	セクション i より上流にあるセクション
s_i^{down}	セクション i より下流にあるセクション
I_i	セクション i の電流
Z_i	セクション i のインピーダンス
V_i	セクション i における電圧
V^{min}	電圧の許容範囲 (入力)

- 電圧が許容範囲を下回ると，電化製品などが適切に動作できない恐れがある．
- 純粋な ASP のみで扱うことが困難な制約である．