SAT 符号化を用いた命題論理式の極小・極大モデル計算の実装と評価

102230349 前田 悠士朗

番原・宋研究室

2025 年度卒業研究発表会
2026 年 2 月 20 日

命題論理式のモデル

- 命題論理式の充足可能性判定問題を SAT 問題といい，通常 CNF という形式で記述される.
- CNF の解 (モデル) が持つ良い性質の例として，極小性，極大性があり，それらを満たすモデルを極小モデル，極大モデルと呼ぶ.

- 極小モデル: 命題論理式 Φ のモデル M について，$M' \subset M$ かつ M' も Φ のモデル，という M' が存在しないもの.
- 極大モデル: 命題論理式 Φ のモデル M について，$M \subset M''$ かつ M'' も Φ のモデル，という M'' が存在しないもの.

- 極小・極大モデルの応用例
  - 多目的最適化ナップサック問題を順序符号化し，極小・極大モデルを列挙することで，パレートフロントを求めることができる.
  - 整数ナップサック問題を多値符号化し，極大モデルを列挙することで，圧縮解を求めることができる.

先行研究

[Koshimura+,'09] では，SAT ソルバーを複数回起動し，1 つの極小モデルを求める方法が提案された.

ここで SAT 問題を符号化した P が与えられたとき，極小モデルは以下の手順で求めることができる.

1. P に SAT ソルバーを起動し，モデル M を求める
2. 1. で求めたモデル M のうち，真として含まれるリテラルを $x_1, ..., x_m$ とし，偽として含まれるリテラルを $y_1, ..., y_n$ とし，次の F1,F2 を作る

$$\text{F1} = \neg(x_1 \wedge ... \wedge x_m)$$
$$\text{F2} = \neg y_1 \wedge ... \wedge \neg y_n$$

3. $P := P \wedge F1 \wedge F2$ として，SAT ソルバーを起動する
   - UNSAT ならば，モデル M が極小モデル
   - SAT ならば，1. に戻る

この方法ではの方法では SAT ソルバーの起動回数が増加し，計算コストが大きくなるという課題がある.

研究概要

SAT 符号化を用いた極小・極大モデルを効率的に計算するシステムの実現

- [足立,'23] では CNF に変換を施し，SAT ソルバー 1 回起動で，1 つの極小モデルを計算する方法が提案されたが，極大モデルを計算する方法の提案や実装，評価はされていなかった.

研究内容

1. 極大モデル計算のための CNF 変換の提案
   - 変換後の CNF のモデルの集合が，変換前の CNF の極大モデルの集合と一致するような変換.
2. CNF 変換による極小・極大モデル計算の実装
   - SAT ソルバー 1 回起動につき，1 つの極小・極大モデルを計算する
3. CNF 変換による極小・極大モデル計算のグラフ問題を用いた実験

提案変換

アイデア

ある変数が負リテラルとして含まれる全ての節において，節中の他の変数で節が満たされている場合，その変数を真にすることで，できるだけ多くの変数を真にする.

与えられた CNF を Φ，変換後の CNF を Ω とすると，提案変換は，Φ に以下のような制約 $M(\Phi)$ を追加し，$\Omega = \Phi \wedge M(\Phi)$ とするものである.

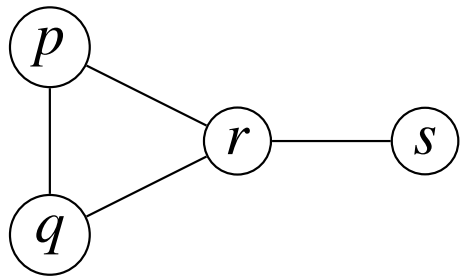
$$M(\Phi) = \bigwedge_{p \in Var(\Phi)} Cl(\Phi, \neg p) \rightarrow p$$

$$Cl(\Phi, \neg p) \equiv \bigwedge_{c \in \Phi, \neg p \in c} c \backslash \{\neg p\}$$

- $Cl(\Phi, \neg p)$: $\neg p$ を含む全ての節が，$\neg p$ 以外のリテラルで満たされている状態.
- $M(\Phi)$: $Cl(\Phi, \neg p)$ が真ならば，p を真にする制約.

提案変換の説明 (例)

- 次のグラフ上の極大独立集合問題を考える.
- 独立集合条件：隣接する 2 頂点を同時に選ばない.



CNF Φ (独立集合条件)

$$\neg p \vee \neg q$$
$$\neg q \vee \neg r$$
$$\neg r \vee \neg p$$
$$\neg r \vee \neg s$$

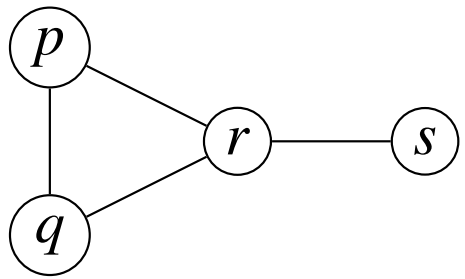
追加する節集合 $M(\Phi)$

- Φ のモデルは以下の 7 つ (1: 選択, 0: 非選択)

$$(p,q,r,s) = (0,0,0,0),(0,0,0,1),(0,0,1,0),(0,1,0,0),$$
$$(0,1,0,1),(1,0,0,0),(1,0,0,1)$$

提案変換の説明 (例)

- 次のグラフ上の極大独立集合問題を考える.
- 独立集合条件：隣接する 2 頂点を同時に選ばない.



CNF Φ (独立集合条件)

$$\neg p \vee \neg q$$
$$\neg q \vee \neg r$$
$$\neg r \vee \neg p$$
$$\neg r \vee \neg s$$

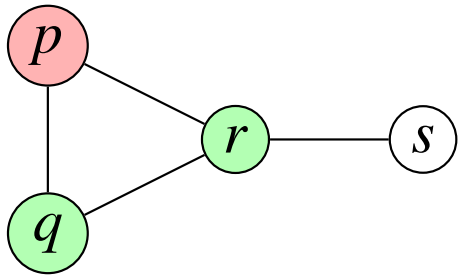
追加する節集合 $M(\Phi)$

$$(\neg q \wedge \neg r) \to p$$
$$(\neg p \wedge \neg r) \to q$$
$$(\neg p \wedge \neg q \wedge \neg s) \to r$$
$$\neg r \to s$$

- 最終的に残ったモデル (極大独立集合)

$$(p,q,r,s) = \underbrace{(0,0,1,0)}_{\{r\}}, \underbrace{(0,1,0,1)}_{\{q,s\}}, \underbrace{(1,0,0,1)}_{\{p,s\}}$$

提案変換の説明 (例)



CNF Φ (独立集合条件)

$$\neg p \vee \neg q$$
$$\neg q \vee \neg r$$
$$\neg r \vee \neg p$$
$$\neg r \vee \neg s$$

追加する節集合 $M(\Phi)$

$$(\neg q \wedge \neg r) \to p$$
$$(\neg p \wedge \neg r) \to q$$
$$(\neg p \wedge \neg q \wedge \neg s) \to r$$
$$\neg r \to s$$

- 変数 p は第 1, 3 節に負リテラルとして含まれる.
- 隣接頂点 q, r が選ばれていない $(q = r = 0)$ なら，$p = 1$.

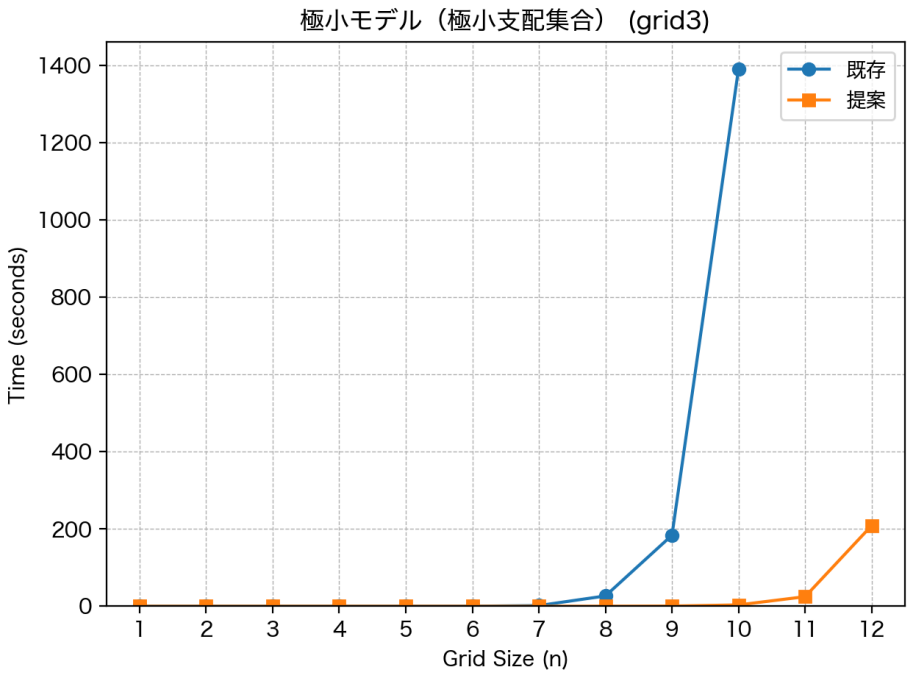
$$(p, q, r, s) = \cancel{(0,0,0,0)}, \cancel{(0,0,0,1)}, (0,0,1,0), (0,1,0,0),$$
$$(0,1,0,1), (1,0,0,0), (1,0,0,1)$$

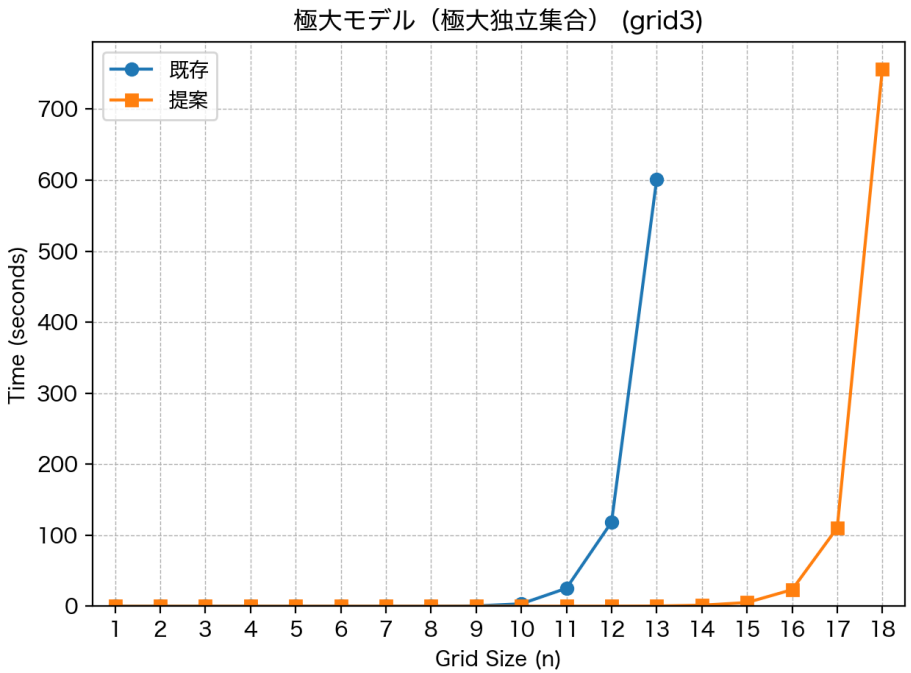
実験概要

既存研究 [足立，’23] の変換と本研究の提案変換を評価するために実験を行った.

- 比較対象
  - [Koshimura+,’09] の極小・極大モデル計算法（既存手法）
  - [足立，’23] の極小，本研究の極大モデル計算法（提案手法）
- ベンチマーク：3 行 n 列 $(n = 1, 2, ..., 19)$ の Grid graph の全 19 問
  - 極小モデル：極小支配集合問題の全解列挙
  - 極大モデル：極大独立集合問題の全解列挙
- 制限時間：1 問あたり 30 分
- 使用ソルバー：*CaDiCaL* 3.0.0
- 使用言語：*Rust*
- 実験環境：Mac mini Apple M4，32GB メモリ

実験結果（カクタスプロット）





- 極小モデル計算は 2 問多く，極大モデル計算は 5 問多く制限時間内に解くことができた.
- 解くことができた全ての問題において，既存手法より提案手法の方が速く解くことができた.

両手法における SAT ソルバー起動回数

	極小モデル（極小支配集合）			極大モデル（極大独立集合）		
Benchmark	既存	提案	割合 [%]	既存	提案	割合 [%]
grid3×1	7	2	**28.6**	6	2	**33.3**
grid3×2	23	7	**30.4**	11	4	**36.4**
grid3×3	62	16	**25.8**	23	10	**43.5**
grid3×4	207	53	**25.6**	48	18	**37.5**
grid3×5	594	154	**25.9**	101	38	**37.6**
grid3×6	1743	436	**25.0**	203	78	**38.5**
grid3×7	5502	1268	**23.0**	440	156	**35.5**
grid3×8	17406	3660	**21.0**	907	321	**35.3**
grid3×9	49864	10610	**21.3**	1848	651	**35.2**
grid3×10	138754	30744	**22.2**	3808	1335	**35.1**
grid3×11	-	89079	-	7896	2706	**34.2**
grid3×12	-	258251	-	16127	5518	**34.2**
grid3×13	-	-	-	33281	11228	**33.8**
grid3×14	-	-	-	-	22884	-
grid3×15	-	-	-	-	46634	-
grid3×16	-	-	-	-	94978	-
grid3×17	-	-	-	-	193518	-
grid3×18	-	-	-	-	394286	-
grid3×19	-	-	-	-	-	-

- SAT ソルバー起動回数が 3〜4 分の 1 程度になった.
- SAT ソルバー起動に伴うオーバーヘッドの削減で CPU 時間の短縮に寄与したと考えられる.

まとめ

提案変換の方法について説明し，実装，評価した.

1. 効率的に極大モデルが求まる CNF への変換方法を考案
2. CNF 変換による極小・極大モデル計算の実装
3. CNF 変換による極小・極大モデル計算のグラフ問題を用いた実験

今後の課題

- グラフ問題以外への応用の検証
- 変換自体の効率化
- 全解列挙に対応した SAT ソルバーの使用

References I

[1] Miyuki Koshimura, Hidetomo Nabeshima, Hiroshi Fujita, and Ryuzo Hasegawa.
Minimal model generation with respect to an atom set.
FTP, 9:49–59, 2009.

[2] 足立 啓一.
リテラルの充足に対する制約に基づく命題論理式の充足可能性を保つ変換に関する研究.
神戸大学 修士論文, 2023.

補助スライド

SAT と CNF

SAT(充足可能判定問題)

- 与えられた命題論理式を充足する値割り当てが存在するかどうかを判定する問題.
- 命題論理式を充足する割り当てが存在するとき，充足可能 (SAT) であるといい，存在しないとき充足不能 (UNSAT) であるという.
- 一般的には CNF(連言標準形) で記述する.

CNF

- **CNF** は，節の論理積 (連言) である.
- 節 (clause) は，リテラルの論理和 (選言) である.
- リテラル (literal) は，命題変数，もしくはその否定である.

例

A, B, C を命題変数としたとき，
$(\neg A \vee B) \wedge (A \vee C) \wedge (\neg B \vee \neg C)$

先行研究の詳細

SAT ソルバーの複数回起動による極小モデルを求める方法 [1]

1. CNF のモデルを一つ求める.
2. そのモデルで偽となる変数について，その変数が偽であるという単位節を CNF に追加し，それ以外の変数についてそれらの変数のいずれかが偽となるという制約を CNF に追加する.
3. 2 で生成した CNF が UNSAT であれば，1 で求めたモデルを極小モデルとして出力する.SAT であれば，2 の操作を繰り返す.

SAT ソルバーの 1 回起動による極小モデルを求める方法 [2]

- 与えられた CNF を Φ，変換後の CNF を Ω とすると，提案変換は，Φ に以下のような制約 $M(\Phi)$ を追加し，$\Omega = \Phi \wedge M(\Phi)$ とするものである.

$$M(\Phi) = \bigwedge_{p \in Var(\Phi)} p \rightarrow \neg Cl(\Phi, p)$$

$$Cl(\Phi, p) \equiv \bigwedge_{c \in \Phi, p \in c} c \backslash \{p\}$$

提案変換が有効に働く問題の例

CNF の例

$$\left(\bigwedge_i C_i^+\right) \wedge \left(\bigwedge_i C_j^-\right)$$

- ここで，C_i^+ は正リテラルのみを含む節で，C_j^- は負リテラルのみを含む節である.

応用例

- 整数ナップサック問題を符号化した CNF
  - 順序符号化し，提案方法で極大モデルを列挙することで，多重ナップサック問題のパレートフロントが求まる.
  - 多値符号化し，提案方法で極大モデルを列挙することで，圧縮解を求めることができる

提案変換のアルゴリズム

Algorithm 1 CNF Φ から CNF への変換方法 Ω(min/max)

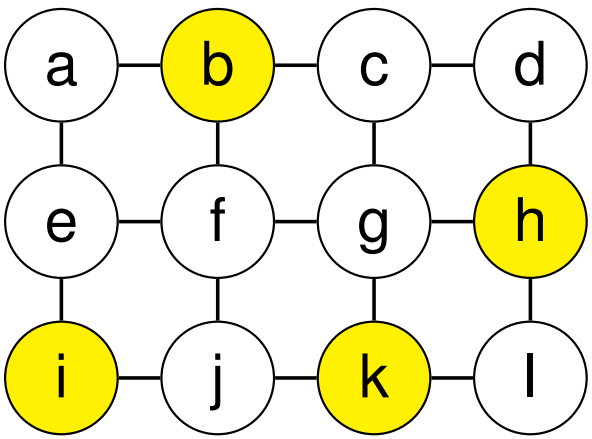
Require: CNF Φ, mode $\in \{\min, \max\}$
Ensure: CNF Ω

$P \leftarrow \mathrm{Var}(\Phi)$
$M \leftarrow \top$
for all $p \in P$ **do**
    $c_1 \leftarrow \bot$
    if $mode = \min$ **then**
        $t \leftarrow p$
        $h \leftarrow \neg p$
    else
        $t \leftarrow \neg p$
        $h \leftarrow p$
    end if
    for all $clause \in \Phi$ **do**
        if $t \in clause$ **then**
            $x \leftarrow$ new variable
            $c_2 \leftarrow \bot$
            for all $l \in clause$ **do**
                if $l \neq t$ **then**
                $M \leftarrow M \wedge (\neg x \vee \neg l)$
                $c_2 \leftarrow c_2 \vee l$
                end if
            end for
            $M \leftarrow M \wedge (c_2 \vee x)$
            $c_1 \leftarrow c_1 \vee x$
        end if
    end for
    $M \leftarrow M \wedge (h \vee c_1)$
end for
$\Omega \leftarrow \Phi \wedge M$
return Ω

3×4 Grid graph における MIS と MDS の例

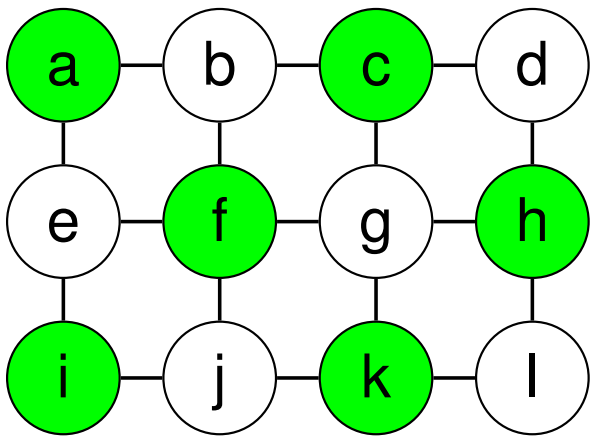
図 6.1 に 3×4 の Grid graph における極小支配集合（MDS）の例を，
図 6.2 に 3×4 の Grid graph における極大独立集合（MIS）の例を示す.

- 支配集合条件：任意の頂点は，選ばれた頂点そのものか，選ばれた頂点に隣接していなければならない.
- 独立集合条件：隣接する 2 頂点を同時に選ばない.



● MDS の頂点（極小支配集合）

図 6.1：3×4 Grid graph の MDS



● MIS の頂点（極大独立集合）

図 6.2：3×4 Grid graph の MIS

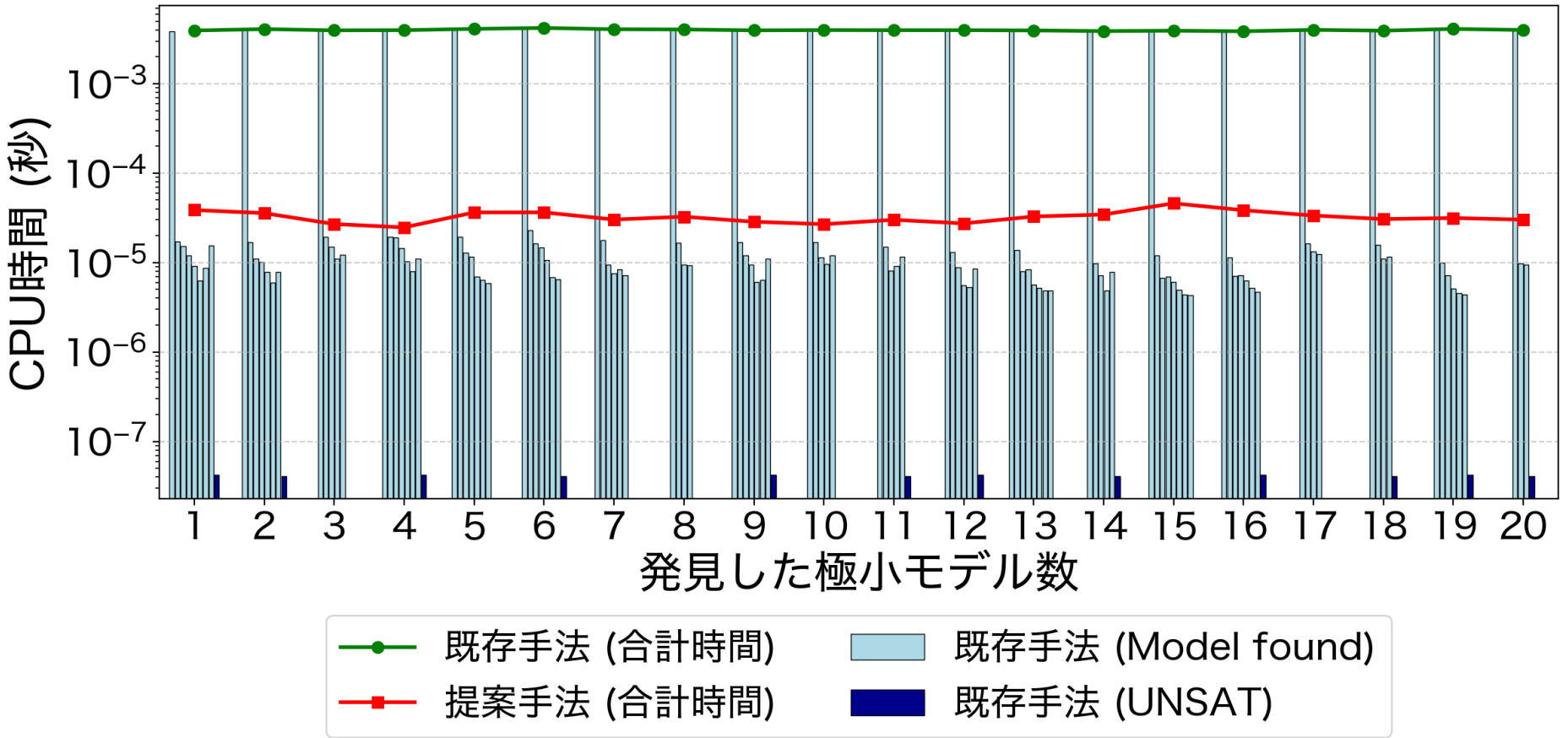
実験結果（実行時間と結果数）

Benchmark	極小モデル（極小支配集合）			
	既存		提案	
	time	results	time	results
grid3×1	< 0.0	2	< 0.0	2
grid3×2	< 0.0	7	< 0.0	7
grid3×3	< 0.0	16	< 0.0	16
grid3×4	< 0.0	53	< 0.0	53
grid3×5	0.01	154	< 0.0	154
grid3×6	0.15	436	0.01	436
grid3×7	1.86	1268	0.03	1268
grid3×8	26.32	3660	0.13	3660
grid3×9	182.35	10610	0.58	10610
grid3×10	1361.24	30744	3.620	30744
grid3×11	T.O.	-	24.84	89079
grid3×12	T.O.	-	207.90	258251
grid3×13	T.O.	-	T.O.	-

- 全ての問題において早い時間で解けた.
- MIS では 5 問多く解けた.

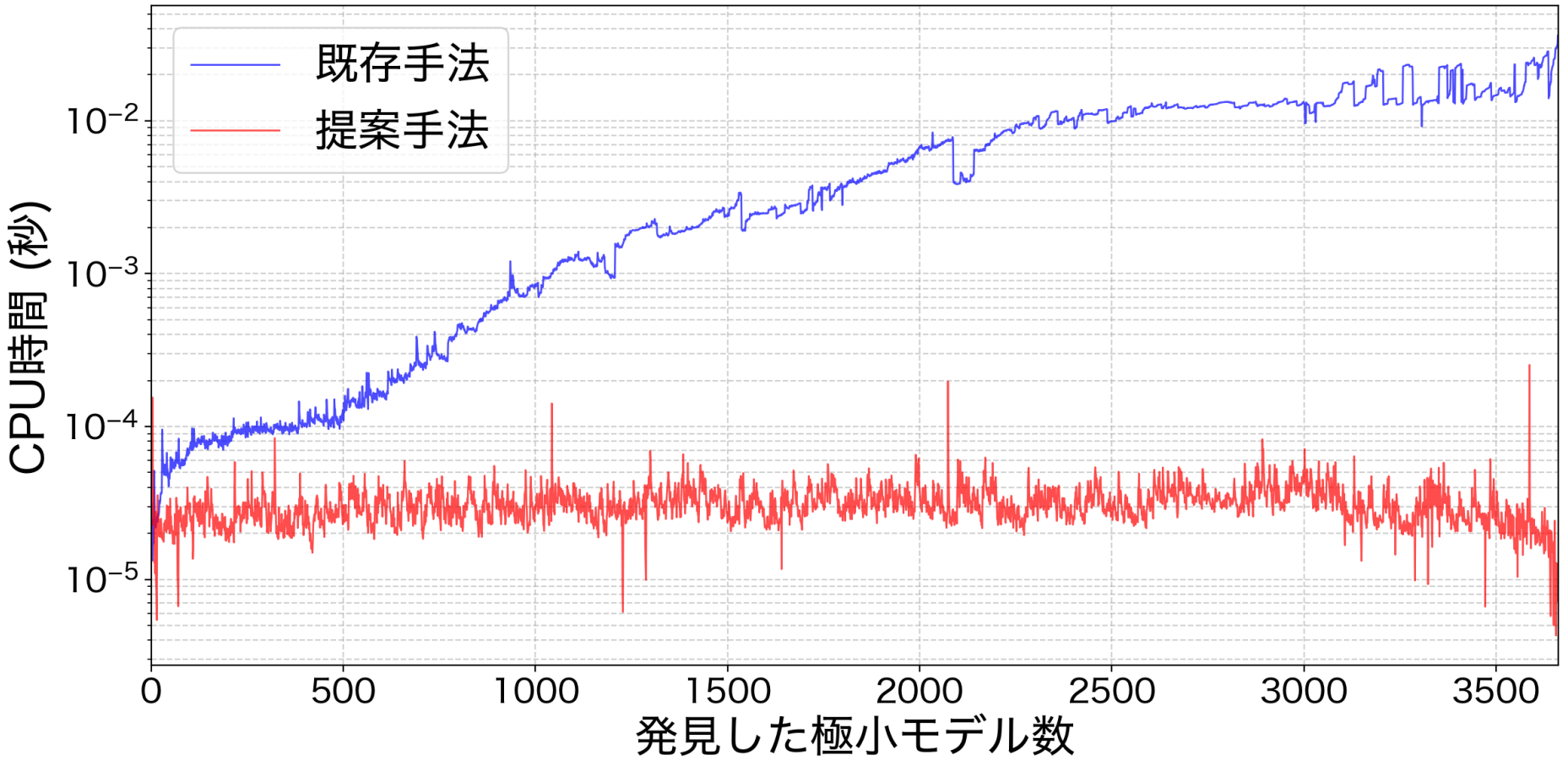
Benchmark	極大モデル（極大独立集合）			
	既存		提案	
	time	results	time	results
grid3×1	< 0.0	2	< 0.0	2
grid3×2	< 0.0	4	< 0.0	4
grid3×3	< 0.0	10	< 0.0	10
grid3×4	< 0.0	18	< 0.0	18
grid3×5	< 0.0	38	< 0.0	38
grid3×6	< 0.0	74	< 0.0	74
grid3×7	0.01	148	< 0.0	148
grid3×8	0.01	290	< 0.0	290
grid3×9	0.04	578	0.01	578
grid3×10	0.14	1156	0.01	1156
grid3×11	0.36	2706	0.05	2706
grid3×12	114.72	5518	0.14	5518
grid3×13	584.70	11228	0.36	11228
grid3×14	T.O.	-	1.52	22884
grid3×15	T.O.	-	5.26	46634
grid3×16	T.O.	-	23.53	94978
grid3×17	T.O.	-	110.72	193518
grid3×18	T.O.	-	756.59	394286
grid3×19	T.O.	-	T.O.	-

SAT ソルバー起動により要した CPU 時間



- 既存手法では，頻繁な SAT ソルバーの起動がオーバーヘッドとなり，CPU 時間全体の増加を招いていることが分かる．
- 提案手法ではソルバーの起動回数が大幅に抑制されているため，起動に伴うコストが低減され，効率的な計算が実現できている．

既存手法と提案手法の計算したモデル数と CPU 時間



- 既存手法と比較して，提案手法は各モデルを計算するのに要する時間がモデル数の増加に伴い短くなり，かつ安定して推移していることが読み取れる

両手法における CNF の変数数と節数

	極小モデル（極小支配集合）				極大モデル（極大独立集合）			
	既存		提案		既存		提案	
Benchmark	vars	clauses	vars	clauses	vars	clauses	vars	clauses
grid3×1	3	3	10	16	3	2	3	5
grid3×2	6	6	26	60	6	7	6	13
grid3×3	9	9	42	110	9	12	9	21
grid3×4	12	12	58	160	12	17	12	29
grid3×5	15	15	74	210	15	22	15	37
grid3×6	18	18	90	260	18	27	18	45
grid3×7	21	21	106	310	21	32	21	53
grid3×8	24	24	122	360	24	37	24	61
grid3×9	27	27	138	410	27	42	27	69
grid3×10	30	30	154	460	30	47	30	77
grid3×11	-	-	170	510	33	52	33	85
grid3×12	-	-	186	560	36	57	36	93
grid3×13	-	-	-	-	39	62	39	101
grid3×14	-	-	-	-	-	-	42	109
grid3×15	-	-	-	-	-	-	45	117
grid3×16	-	-	-	-	-	-	48	125
grid3×17	-	-	-	-	-	-	51	133
grid3×18	-	-	-	-	-	-	54	141
grid3×19	-	-	-	-	-	-	-	-

- 極小モデル計算では Tseitin 変換により，変数の数が増加した.
- Tseitin 変換による CNF の複雑化のため，極大モデル計算の方が既存手法と提案手法の性能差が大きく現れていると考えられる.