SAT ソルバーを用いた高速な部分グラフ探索ツールの実装と評価

川原 征大　宋 剛秀　番原 睦則　田村 直之

部分グラフ探索とは，与えられたグラフから制約を満たす部分グラフを見つけることであり，回路配線等の様々な応用が知られている．本論文では，SAT ソルバーを用いた高速な部分グラフ探索ツールを提案・評価する．既存研究としては Zero-suppressed decision diagram(ZDD) を用いた部分グラフ探索ツール Graphillion がある．ZDD は解の数が多くなっても高速に全列挙可能という特長を持つが，大きなグラフでは ZDD の構築に失敗することがある．一方，SAT ソルバーは大規模な問題に対しても，高速な単解探索が可能であるという特長を持つ．本研究ではまず部分グラフ探索で使われるグラフ上の制約の SAT 符号化方法を提案する．特に頂点の連結性に関する制約モデルについては推移関係を用いた方法と次数の制約を用いた方法の 2 種類を提案する．計算機実験の結果，k クリーク問題，マルチパス問題，ハミルトン閉路の単解探索において，ZDD が利用不可能な 1000 頂点以上のグラフでも高速に求解できることを確認した．

Subgraph search is to find subgraphs satisfying given constraints, and it has various applications such as circuit wiring. In this paper, we propose and evaluate a subgraph search tool using SAT solvers. There is a well known subgraph search tool Graphillion using ZDD, which can enumerate over millions of subgraphs. However, it fails to construst ZDD when a given graph becomes large. On the other hand, the advantage of SAT solver is that it can find one solution even if a given graph becomes large. In the proposed method, we first formalize constraints on graphs as CSP. In particular, for connectivity constraints, we propose transitivity constraint model and degree constraint model. Then, we find subgraphs by using SAT-based constraint solver. We evaluate the performance of the proposed tool on k-Clique Problem, multi-path problem and Hamiltonian Cycle Problem, and confirm it works well on graphs of over 1000 nodes which cannot be solved by ZDD.

1　はじめに

部分グラフ探索とは，与えられたグラフから制約を満たす部分グラフを見つける探索のことである．部分グラフ探索には電子回路設計などの様々な実世界の応用があり，部分グラフ探索を高速に行うツールを研究・開発することは重要である [8] [16] [17] [5].

部分グラフ探索ツールは大きく 2 種類ある．一つ目は部分グラフをグラフ理論における専用アルゴリズムを用いて求めるツールであり [16] [17] [5]，二つ目は部分グラフが満たすべき条件を制約式で表して汎用的なソルバーを用いて求めるツールである [8]．一つ目のツールは専用アルゴリズムを用いた高速な求解が特長であるが，部分グラフに追加の条件があった場合にはアルゴリズムを書き換える必要があり対応が難しい．一方，二つ目のツールは汎用的なソルバーを使うため，専用アルゴリズムに比べて求解性能が劣ることがあるが，追加の条件に対して新たに制約を記述することで対応することができる．

二つ目の汎用的なソルバーを用いる部分グラフ探索ツールの代表的なものとして Zero-suppressed deci-

SAT ソルバーを用いた高速な部分グラフ探索ツールの実装と評価

Masahiro Kawahara, 神戸大学大学院システム情報学研究科, Graduate School of System Informatics, Kobe University.

Takehide Soh, Mutsunori Banbara, Naoyuki Tamura, 神戸大学情報基盤センター, Information Science and Technology Center, Kobe University.

sion diagram (ZDD) [13] を用いた Graphillion[†1] が提案・公開されている [8]．Graphillion はフロンティア法を利用した ZDD 構築アルゴリズム [10] [11] によって，制約を満たす部分グラフ集合を高速に全列挙可能という特長を持っている．これまで電力網の評価 [9] や鉄道網分析 [†2] など実問題への応用が活発に行われるなど，Graphillion は優れた部分グラフ探索ツールである．しかしながら，探索対象となる与えられたグラフのサイズが大きくなると ZDD の構築に非常に時間がかかったり，メモリ不足のために ZDD の構築ができないことがある．この場合，部分グラフを 1 つも得られないため，ZDD の構築ができない大きなグラフに対しても高速に解を返すことができる汎用的な部分グラフ探索ツールの研究が課題となっている．

命題論理の充足可能性判定問題 (SAT) は与えられた連言標準形の命題論理式を充足する値割り当てを見つける問題であり，SAT 問題を解くプログラムは SAT ソルバーと呼ばれる [2] [7] [15]．近年の SAT ソルバーの性能向上は目覚ましく，数百万変数から構成されるような巨大な SAT 問題でも高速に単解を計算可能である．このような SAT ソルバーの求解性能を利用した問題解法が様々な分野で成功を収めており [7] [1] [23] [14], 特に整数有限領域上の制約充足問題 (CSP) を SAT に符号化して SAT ソルバーで解く SAT 型制約ソルバーが多く開発されている [22] [20] [24] [12] [6] [21]．

本論文では，SAT ソルバーを用いた新しい部分グラフ探索の方法を提案し，提案方法を実装したツール Scarabellion を使ってその性能の評価を行う．SAT ソルバーを用いて部分グラフ探索を行うために，本論文ではまず部分グラフ探索問題とグラフ制約の定義を行い，その制約モデルを提案する．特にグラフ制約の中でも頂点の連結に関する制約は SAT 符号化した際の符号化節数が多くなることが問題となるが，本研究ではこの制約に対し推移モデルと次数モデルという 2 つの異なる特長を持つ制約モデルを提案する．最終的に部分グラフ探索問題は提案した制約モデルと SAT 型制約ソルバー Scarab [20] を用いて SAT 問題へと符号化する．これらの提案方法をまとめて部分グラフ探索ツール Scarabellion の実装を行った．

提案方法を評価するために，k クリーク問題，マルチパス問題，ハミルトン閉路問題を用いた計算機実験を行った．その結果，提案方法を実装した Scarabellion は既存ツールである Graphillion と比較して，より多くの問題を高速に解き，またより大きなグラフの問題を解くことができることを確認した．また提案した 2 つの制約モデルについても，それぞれ異なる問題に適していることを確認した．

本論文の構成は以下のようになる．まず 2 節で部分グラフ探索問題とグラフ制約を定義する．3 節でその制約モデルを説明する．4 節では制約モデルを実装した部分グラフ探索ツール Scarabellion について説明する．5 節では提案方法を評価するための計算機実験について説明する．6 節で本論文のまとめを行う．

2 部分グラフ探索問題

本節では，まず部分グラフ探索問題とその制約について説明を行い，その具体例を示す．

2.1 部分グラフ探索問題と制約

$\mathcal{V}$ を頂点集合，$\mathcal{E}$ を辺集合とするとき，部分グラフ探索の入力となるグラフ（ユニバース）を $\mathcal{G}(\mathcal{V}, \mathcal{E})$ で定義する．**部分グラフ探索問題** とは与えられた制約を全て満たすユニバースの全域部分グラフ $G(\mathcal{V}, E) \subset \mathcal{G}$ を求める問題である．

部分グラフ探索で入力として与えられる制約には様々なものが考えられるが，本研究では代表的な部分グラフ探索ツールの一つである Graphillion [8] で用いられている以下 5 つの制約を対象とする．これらの一つ一つは単純な制約であるが，組み合わせることで様々なグラフ上の問題を汎用的に記述することが可能になっている．ここで括弧内は Graphillion の中で使われている制約名である．

- 頂点集合制約 (Vertex Groups)
  入力: 頂点集合の集合 $\{V_1, ..., V_m\}$
  制約: 各 $V_i \in \{V_1, ..., V_m\}$ について，任意の 2 頂点 $u, v \in V_i$ は G 上で連結しなければならない．

†1 https://github.com/takemaru/graphillion
†2 http://www.nysol.jp/apps/ekillion

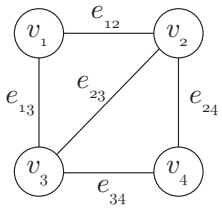

図 1　ユニバース $\mathcal{G}$

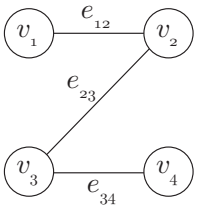

図 2　部分グラフ G

- 次数制約 (Degree Constraints)
  入力: 頂点から整数集合への写像 $f : V \to 2^{\mathbb{N}}$
  制約: G 上の任意の頂点 $v \in \mathcal{V}$ について $deg_G(v) \in f(v)$ が成り立たなければならない．ここで $deg_G(v)$ は G における頂点 v の次数を表す．
- 辺数制約 (Number of Edges)
  入力: 整数集合 $N \subseteq \mathbb{N}$
  制約: G の辺集合 E について，$|E| \in N$ が成り立たなければならない．
- 非閉路制約 (No Loop)
  入力: なし
  制約: G に閉路が含まれてはいけない．
- 候補グラフ集合制約 (Graph Set)
  入力: 辺集合の集合 $\{E_1, ..., E_m\}$
  制約: G の辺集合 E について，$E \in \{E_1, ..., E_m\}$ が成り立たなければならない．

2.2　例: パス探索問題

前節で説明した制約を組み合わせることで，これまで研究されてきた様々なグラフ上の問題を部分グラフ探索問題として定式化することができる．例として，図 1 に示すユニバース $\mathcal{V} = \{v_1, v_2, v_3, v_4\}$ と $\mathcal{E} = \{e_{12}, e_{13}, e_{23}, e_{24}, e_{34}\}$ が与えられた時に 2 頂点 v_1 と v_4 間のパスを求める問題を考える．この問題は (i) **頂点集合制約**に $\{\{v_1, v_4\}\}$，(ii) **次数制約**に $\{(v_1, \{1\}), (v_2, \{0, 2\}), (v_3, \{0, 2\}), (v_4, \{1\})\}$，そして (iii) **非閉路制約**の 3 つの制約を持つ．頂点集合制約は 2 頂点 v_1 と v_4 が G 上で連結しなければならないことを表し，次数制約はパスの始点と終点の次数が 1 であり，その他の頂点の次数は 0 または 2 であることを表している．但し，これらの制約だけでは，余分な閉路を含む部分グラフでも解となるため，非閉路制約が必要となる．これら全ての条件を満たす，全域部分グラフ G は 2 頂点 v_1 と v_4 間のパスになる (図 2)．

この他にも，節 5 で説明するように，ハミルトン閉路問題，k クリーク問題，マルチパス問題などの様々な問題を部分グラフ探索問題として定式化することができる．

3　部分グラフ探索問題の制約モデル

本節では，節 2.1 で説明したグラフ上の制約の有限整数領域上の制約充足問題としての定式化，すわち制約モデルについて説明を行う．

定式化のための定義を行う．部分グラフ探索問題におけるユニバースを $\mathcal{G}(\mathcal{V}, \mathcal{E})$ とする．求める全域部分グラフを $G(\mathcal{V}, E) \subset \mathcal{G}$ とし，G を CSP 中で表現するために，辺集合 E の特性関数を表す以下の変数を導入する．

$$e_{uv} \in \{0, 1\} \quad (\{u, v\} \in \mathcal{E})$$

すなわち e_{uv} は解となる部分グラフ G を表す変数となる．例えば，図 2 のグラフ G は e_{uv} によって以下のように表される．

$$e_{12} = 1, \quad e_{13} = 0, \quad e_{23} = 1, \quad e_{24} = 0, \quad e_{34} = 1$$

以降，特に混乱が生じない時は 0-1 変数を命題変数として扱い，$\neg$,$\to$,$\wedge$,$\vee$ などの論理演算子と一緒に表記する．例えば $e_{uv} = 1$ および $e_{uv} = 0$ を，それぞれ e_{uv} および $\neg e_{uv}$ と表すことがある．

3.1　頂点集合制約の制約モデル

頂点集合制約の入力として頂点集合の集合 $\{V_1, ..., V_m\}$ が与えられたとする．V_i の任意の 2 頂点が G において連結している制約を $connected(V_i)$ と表すと，頂点集合制約は以下で記述できる．

$$\bigwedge_{i \in \{1,..,m\}} connected(V_i)$$

$connected(V_i)$ の定式化は一般的に難しい．そこで，頂点の連結性の推移関係を表す制約 [25] [4] を利用した**推移モデル**と，頂点の次数制約を利用した新たな**次数モデル**を実装した．

3.1.1 推移モデル

2 頂点 s,t が G 上で連結している必要十分条件は，G の全域部分グラフ G' 上で s,t が連結することである．まず G' 上で，s,t が連結する制約を表すために，2 つの命題変数を定義する．

$$a_{uv}, a_{vu} \in \{0,1\} \qquad (\{u,v\} \in \mathcal{E})$$
$$r_{uv} \in \{0,1\} \qquad (u,v \in \mathcal{V})$$

a_{uv}, a_{vu} は G' の辺集合 E' の特性関数を表す変数であり，各辺に対して双方向に定義される (無向辺を有向辺と見ることで制約条件を簡潔に記述できるため)．r_{uv} は G' 上で，u から v に向かうパスが存在することを表す変数である．s から t に向かうパスを求めるために，r_{su} $(u \in \mathcal{V}\backslash\{s\})$ に関する制約を記述する (ただし，$\mathcal{A} = \{(u,v) \mid \{u,v\} \in \mathcal{E}\} \cup \{(v,u) \mid \{u,v\} \in \mathcal{E}\}$ とする)．

$$r_{st}$$
$$r_{su} \rightarrow \bigvee_{\{v,u\} \in \mathcal{E}} a_{vu} \qquad (u \in \mathcal{V} \setminus \{s\})$$
$$a_{uv} \rightarrow r_{su} \qquad ((u,v) \in \mathcal{A})$$

ただし，上式だけでは，閉路の存在を許し，パスが求まらないことがある．有向閉路を禁止するためには，それぞれの有向辺 $(u,v) \in \mathcal{A}$ に対して頂点間の連結性の推移関係を表す以下の制約が必要になる．

$$a_{uv} \rightarrow (r_{uv} \wedge \neg r_{vu})$$
$$(a_{uv} \wedge r_{vw}) \rightarrow r_{uw} \qquad (w \in \mathcal{V} \setminus \{u,v\})$$

最後に，G' が求める G の全域部分グラフである条件を記述する．

$$(a_{uv} \vee a_{vu}) \rightarrow e_{uv} \qquad (\{u,v\} \in \mathcal{E})$$

以上が 2 頂点 s,t が G で連結する制約である．この制約を $\tau(s,t)$ とすると (τ は trainsition の意)，$connected(V_i)$ は以下の連言によって記述できる (但し $V_i = \{v_1, v_2, ..., v_n\}$ とする)．

$$\tau(v_1,v_2) \wedge \tau(v_2,v_3) \wedge \cdots \wedge \tau(v_{n-1},v_n)$$

しかし，各 $\tau(s,t)$ について，新しく上記の制約を記述する必要はない．つまり，$\tau(v_1,v_2)$ の制約を記述した上で，さらに v_1 から $t \in \{v_3, ..., v_n\}$ のそれぞれに向かうパスが存在することを表す，次の制約を加えれば良い．

$$\bigwedge_{t \in \{v_3,\ldots,v_n\}} r_{v_1 t}$$

3.1.2 次数モデル

2 頂点 s,t が G で連結している必要十分条件は，以下を満たす全域部分グラフ $G'(\mathcal{V}, E') \subset G$ が存在することである．ただし，$deg_{G'}(u)$ は G' における u の次数を表す．

$$deg_{G'}(u) = 1 \qquad (u \in \{s,t\})$$
$$deg_{G'}(u) = 0 \ or \ 2 \qquad (u \in \mathcal{V} \setminus \{s,t\})$$

すなわち，s,t の次数が 1 で，他の頂点の次数が 0 または 2 である G' が存在すれば良い．この条件は，以下の制約で表現できる ($adj(u)$ は $\mathcal{G}$ における u の隣接頂点の集合を表す)．ただし e'_{uv} は頂点 s,t と辺 $\{u,v\} \in \mathcal{E}$ に対し導入された新しい変数であり，G' の辺集合 E' を表す．

$$e'_{uv} \in \{0,1\} \qquad (\{u,v\} \in \mathcal{E})$$
$$e'_{uv} \leq e_{uv} \qquad (\{u,v\} \in \mathcal{E})$$
$$\sum_{v \in adj(u)} e'_{uv} = 1 \qquad (u \in \{s,t\})$$
$$\sum_{v \in adj(u)} e'_{uv} = 0 \ or \ 2 \qquad (u \in \mathcal{V} \setminus \{s,t\})$$

この制約を $\delta(s,t)$ と表す (δ は degree の意)．$connected(V_i)$ は以下の連言によって記述できる (但し $V_i = \{v_1, v_2, ..., v_n\}$ とする)．

$$\delta(v_1,v_2) \wedge \delta(v_2,v_3) \wedge \cdots \wedge \delta(v_{n-1},v_n)$$

3.1.3 両モデルの変数の数と節数

入力が $\{V_1, ..., V_m\}$ で与えられた時，節 3.1.1 と節 3.1.2 で説明した推移モデルと次数モデルの変数の数と節数は以下のようになる．ここで，基数制約 $\sum_{i=1}^{l} x_i \leq k$ の SAT 符号化節数は $O(l \cdot k)$ と仮定している [18]．また入力の各頂点集合の平均要素数 $n = (\sum_i |V_i|)/m$ とする．

	変数の数	節数
推移モデル	$O(m \cdot \lvert\mathcal{V}\rvert^2)$	$O(m \cdot \lvert\mathcal{E}\rvert \cdot \lvert\mathcal{V}\rvert)$
次数モデル	$O(m \cdot n \cdot \lvert\mathcal{E}\rvert)$	$O(m \cdot n \cdot \lvert\mathcal{E}\rvert)$

この表から分かるように，それぞれのモデルの符号化節数は異なるパラメータに依存していることが分かる．このことから両モデルには得意・不得意な問題があることが予測できる．節 5 の計算機実験では，マルチパス問題とハミルトン閉路問題を用いて，両モデルの求解性能を比較する．

3.2 次数制約の制約モデル

次数制約の入力として頂点から整数集合への写像 $f : V \to 2^{\mathbb{N}}$ が与えられたとする．この時，次数制約は次のように定式化できる．

$$\bigvee_{d \in f(u)} \sum_{v \in adj(u)} e_{uv} = d \qquad (u \in \mathcal{V})$$

3.3 辺数制約の制約モデル

辺数制約の入力として整数集合 $N \subseteq \mathbb{N}$ が与えられたとする．この時，辺数制約は次のように定式化できる．

$$\bigvee_{n \in N} \sum_{\{u,v\} \in \mathcal{E}} e_{uv} = n$$

3.4 非閉路制約の制約モデル

非閉路制約の制約モデルはこれまでいくつか提案されているが [4]，グラフのサイズが大きくなると符号化節数が非常に多くなることが知られている．本研究では非閉路制約は明示的に記述せず CounterExample-guided abstraction refinement(CEGAR) [3] の手法を用いて，制約を暗黙的に表す．つまり，非閉路制約を除いた緩和問題の解を求め，解に閉路がふくまれていればその閉路を否定する制約を加えて，閉路を含まない解が見つかるまで探索を継続する方法を取る．ある緩和問題の解が辺集合で表される k 個の閉路 $\{E_1, ..., E_k\}$ を含んでいた場合，以下の制約を加えることにより，これらの閉路を否定する．

$$\bigwedge_{1 \le i \le m} \bigvee_{\{u,v\} \in E_i} \neg e_{uv}$$

3.5 候補グラフ集合制約の制約モデル

候補グラフ集合制約の入力として辺集合の集合 $\{E_1, ..., E_m\}$ が与えられたとする．候補グラフ集合制約は以下のように定式化できる．

$$\bigvee_{i=1}^{m} c_i$$

$$c_i \to \bigwedge_{\{u,v\} \in E_i} e_{uv} \wedge \bigwedge_{\{u,v\} \in \mathcal{E} \setminus E_i} \neg e_{uv}$$

ここで c_i $(1 \le i \le m)$ は補助命題変数である．

3.6 複数のグラフ制約の組合せに対する制約モデル

論文 [19] では，ハミルトン閉路問題を SAT ソルバーを用いて解く方法が提案されている．ハミルトン閉路を求めるためのグラフ制約は通常以下の 2 つになる．

- 頂点集合制約: $\{\{\mathcal{V}\}\}$
- 次数制約: $\{(v, \{2\}) \mid v \in \mathcal{V}\}$

しかし，論文 [19] では頂点集合制約を記述せずに CEGAR [3] を使って効果的に問題を解いている．

このような複数のグラフ制約の組合せに対する，制約モデルについても実装を行う．具体的な解法は，頂点集合制約を除いた緩和問題，すなわち次数制約のみで解を求める．その解が頂点集合制約を満たしていなければ，解に存在する閉路を否定した制約を加えて，頂点集合制約を満たすまで探索を継続する．

今回は頂点集合 $\{\{\mathcal{V}\}\}$ が与えられた時 (すなわちハミルトン閉路) に限って CEGAR の実装を行った．非閉路制約と同様に，辺集合で表される k 個の閉路 $\{E_1, ..., E_k\}$ $(k \ge 2)$ が解に存在した時，以下の制約を加えながら，頂点集合制約を満たす解が見つかるまで探索を継続する．

$$\bigwedge_{1 \le i \le m} \bigvee_{\{u,v\} \in E_i} \neg e_{uv}$$

4 SAT ソルバーを用いた部分グラフ探索ツールの実装

本節では，前節で説明したグラフ制約の SAT 符号化を用いた部分グラフ探索ツールの実装について説明する．SAT ソルバーを用いた部分グラフ探索問題の解法の手順は以下になる．

1. 部分グラフ探索問題を 0-1 変数上の CSP として表す.
2. CSP を SAT 問題へと符号化する．
3. SAT 問題を SAT ソルバーを用いて求解し，解を得る．
4. SAT 問題の解を部分グラフへと変換する．

手順 1 は前節で説明した制約モデルを用いて CSP を構成する．手順 2, 3, 4 については，既存のツール [22] [20] [24] [12] [6] [21] を利用することができる．

今回は，Scala 上に実現された SAT 型制約プログラミングシステム Scarab[20] を基にして，グラフ制約の提案制約モデルを実装した部分グラフ探索ツール Scarabellion を開発した．

5 計算機実験

本節では，k クリーク問題，マルチパス問題，ハミルトン閉路問題をベンチマークにして提案方法実装した部分グラフ探索ツール Scarabellion と既存ツール Graphillion との比較を行う．

特に頂点集合制約を用いるマルチパス問題，ハミルトン閉路問題については節 3.1 で説明した推移モデルと次数モデルの比較を行う．ハミルトン閉路についてはこれらの制約モデルに加えて節 3.6 で説明した CEGAR との比較も行う．実験に用いた計算機の CPU は Intel Xeon 3.00GHz でメモリは 16GB である．

5.1 k クリーク問題

実験の概要：まず k クリーク問題を用いて Scarabellion と Graphillion の求解性能の比較を行った．k クリーク問題は以下のグラフ制約を用いて部分グラフ探索問題として定式化できる．

- 次数制約: $\{(v, \{0, k-1\}) \mid v \in \mathcal{V}\}$
- 辺数制約: $\{k(k-1)/2\}$

ベンチマーク：ベンチマークにはグラフ彩色問題で使われている color04[†3] の 119 個のグラフを使用し，クリークの次数は $k = 3$ ，制限時間は 10 分とした．グラフの頂点数毎の求解数の比較結果を表 1 に示す．表の列は左から，グラフの頂点数の範囲，その頂点数の範囲にある問題の数，Graphillion の求解数，提案方法 (Scarabellion) の求解数を表している．

求解した問題の大きさと数の比較：表より Graphillion はグラフの頂点数が 100 以下の問題しか解けなかったことが分かる．一方，提案方法では頂点数が 1000 以上になっても 2 問の問題を解くことに成功しており，より大きなグラフの問題に対応できていることが分かる．

†3 http://mat.gsia.cmu.edu/COLOR/instances.html

表 1 k クリーク問題 ($k = 3$) の求解数

頂点数	問題数	Graphillion	提案方法
1 - 100	26	8	**26**
101 - 200	23	0	**22**
201 - 500	37	0	**33**
501 - 1000	17	0	**10**
1001 - 2000	8	0	**2**
2001 -10000	8	0	0
合計	119	8	**93**

5.2 マルチパス問題

実験の概要：マルチパス問題は長方形のマス目内に数字が配置され、同じ数字の間を、縦横の線で重ならないように結ぶ問題であり、電子回路の自動配線と非常に親和性が高い問題である（ナンバーリンクとも呼ばれる[†4]）．頂点集合 $S = \{s_1, ..., s_n\}$ ，$T = \{t_1, ..., t_n\}$ に対して，頂点のペア $(s_1, t_1), ..., (s_n, t_n)$ のそれぞれがパスの両端点となるマルチパス問題は以下のグラフ制約を用いて部分グラフ探索問題として定式化できる．

- 頂点集合制約: $\{\{s_i, t_i\} \mid 1 \le i \le n\}$
- 次数制約: $\{(u, \{1\}) \mid u \in (S \cup T)\} \cup \{(u, \{0, 2\}) \mid u \in \mathcal{V} \setminus (S \cup T)\}$

本実験では Graphillion との比較に加えて，頂点集合制約に対する推移モデルと次数モデルの比較も行う．

ベンチマーク：ベンチマークには 2014 年のアルゴリズムデザインコンテスト[†5] で使われた 11 問を使用し．制限時間は 60 分とした．それぞれの問題の求解時間をと各制約モデルの節数を比較した結果を表 2 に示す．

2 つの制約モデルの比較：まず各制約モデルの符号化節数を比較すると，推移モデルが次数モデルより 10 倍から 100 倍多くなることが分かる．また推移モデルでは NL_Q07 以降の問題を制限時間内に符号化することができなかった．これは節 3.1.3 に説明したように，推移モデルの符号化節数がグラフの頂点数に大

†4 ナンバーリンクはニコリ社によるパズルである (http://www.nikoli.co.jp).

†5 http://www.sig-sldm.org/designcontest.html

表 2　マルチパスの求解時間 (秒)

問題名	マス目サイズ	線の数	Graphillion	提案方法 (求解時間)		提案方法 (節数)	
				推移モデル	次数モデル	推移モデル	次数モデル
NL_Q01	10x10	7	**0.1**	73.1	12.8	549275	11262
NL_Q02	10x10	10	**0.1**	49.9	5.4	784491	15778
NL_Q03	10x18	9	**1.2**	546.3	7.3	231306	26075
NL_Q04	10x18	20	**0.1**	1146.9	10.6	5157339	55983
NL_Q06	14x18	9	M.O.	T.O.	**284.2**	4602022	36806
NL_Q07	17x15	13	M.O.	T.O.	T.O.	—	52885
NL_Q08	16x19	28	**32.46**	T.O.	T.O.	—	133155
NL_Q09	18x18	18	147.67	T.O.	**55.8**	—	92375
NL_Q13	18x21	17	M.O.	T.O.	**112.4**	—	102260
NL_Q14	18x21	17	M.O.	T.O.	T.O.	—	102260
NL_Q15	35x19	35	M.O.	T.O.	**256.4**	—	367143
求解した問題の合計数			6	4	**8**		

表 3　ハミルトン閉路問題の求解数

頂点数	問題数	Graphillion	提案方法		
			推移モデル	次数モデル	CEGAR
1 - 100	23	10	**23**	17	**23**
101 - 200	21	0	9	0	**18**
201 - 500	34	0	5	0	**23**
501 - 1000	15	0	0	0	**5**
1001 - 2000	6	0	0	0	**1**
2001 - 10000	8	0	0	0	0
合計	107	10	37	17	**70**

きく依存するのに対し，次数モデルでは頂点集合制約の入力における集合の要素数に依存することが原因である．マルチパス問題では入力となる集合の要素数は 2 であり，次数モデルが適していると考えられる．

求解時間の比較：次に求解時間を比較すると，符号化節数が少ない次数モデルが推移モデルより多くの問題を高速に解いていることが分かる．また推移モデルは Graphillion と比較しても，多くの問題を解くことに成功した．また推移モデルは Graphillion がメモリ不足により ZDD を構築できなかった最も大きな問題である NL_Q15 を解くことにも成功した．

5.3　ハミルトン閉路問題

実験の概要：ハミルトン閉路問題を用いて Graphillion と Scarabellion に実装された 3 つの提案方法，推移モデル，次数モデル，CEGAR の比較を行う．ハミルトン閉路問題は以下のグラフ制約を用いて部分グラフ探索問題として定式化できる．

- 頂点集合制約: $\{\{\mathcal{V}\}\}$
- 次数制約: $\{(v, \{2\}) | v \in \mathcal{V}\}$

ベンチマーク：実験で用いるグラフは節 5.1 の k クリークと同様にグラフ彩色問題で使われている color04 のグラフを使用し，制限時間は 10 分とした．ただし，119 個のグラフの中の 12 問については次数が 1 の頂

点を含んでおり明らかにハミルトン閉路を含まないので除外し，残りの 107 個のグラフを実験に使用した．求解数の結果を表 3 に示す．

3 つの制約モデルの比較：表より 3 つの制約モデルでは CEGAR モデルが最も多くの数の問題と最も大きなグラフの問題を解いたことが分かる．次数モデルと推移モデルの比較では，マルチパス問題と異なり推移モデルがより多くの問題を解いた．これは次数モデルの符号化後の変数の数と節数が，頂点集合制約の入力となる集合の要素数に大きく依存するためである．ハミルトン閉路の制約の場合，頂点集合制約の入力は要素数 $|\mathcal{V}|$ 個の頂点集合のみなので，節 3.1.3 で述べた推移モデルと次数モデルの変数の数と節数は以下のようになる．

	変数の数	節数
推移モデル	$O(\|\mathcal{V}\|^2)$	$O(\|\mathcal{E}\| \cdot \|\mathcal{V}\|)$
次数モデル	$O(\|\mathcal{V}\| \cdot \|\mathcal{E}\|)$	$O(\|\mathcal{E}\| \cdot \|\mathcal{V}\|)$

この表から分かる通り，変数の数において，次数モデルが推移モデルを上回っているため，次数モデルの求解数が少なくなったと考えられる．実際に，推移モデルが解けて次数モデルが解けなかった 1-Insertions_5 (頂点数 202，辺数 1227) の問題では，推移モデルの変数が 44284 個，節数が 1016249 個であったのに対し，次数モデルの変数が 534681 個，節数が 1640367 個と，次数モデルの変数の数が推移モデルに比べて 10 倍程度多いことが分かる．さらに，推移モデルで解けて次数モデルで解けなかった 201 頂点以上のグラフ 5 問中 2 問については次数モデルは制限時間内に符号化できなかった．

求解時間の比較：表より 3 つの制約モデルいずれも既存ツールである Graphillion より，多くの問題を解くことに成功した．特に CEGAR モデルでは頂点数が 1000 以上のグラフについてもハミルトン閉路を求めることに成功しており，より大きなグラフの問題に対応できていることが分かる．

6 おわりに

本論文では，部分グラフ探索について SAT ソルバーを用いた新しい解法を提案し，提案方法を実装したツール Scarabellion を使ってその性能を評価した．論文ではまず部分グラフ探索問題とグラフ制約を定義し，その制約モデルを示した．特に頂点集合制約については推移モデルと次数モデルの 2 つの制約モデルを提案した．k クリーク問題，マルチパス問題，ハミルトン閉路問題を用いた計算機実験の結果，提案方法は ZDD を用いた既存の部分グラフ探索ツール Graphillion より多くの問題，また大きなグラフの問題を解くことができることを確認した．

今後の課題は，今回実験した 3 つの問題以外の問題でも評価を行うことと各制約モデルのさらなる改良が挙げられる．また長期的な課題として ZDD と SAT の両方を融合した部分グラフ探索方法の研究が挙げられる．

参 考 文 献

[1] 番原睦則, 田村直之: SAT によるシステム検証, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 122–129.

[2] Biere, A., Heule, M., van Maaren, H., and Walsh, T.(eds.): *Handbook of Satisfiability*, Frontiers in Artificial Intelligence and Applications (FAIA), Vol. 185, IOS Press, 2009.

[3] Clarke, E., Grumberg, O., Jha, S., Lu, Y., and Veith, H.: Counterexample-guided abstraction refinement, *Computer aided verification*, Springer, 2000, pp. 154–169.

[4] Gebser, M., Janhunen, T., and Rintanen, J.: SAT modulo graphs: Acyclicity, *Logics in Artificial Intelligence*, Springer, 2014, pp. 137–151.

[5] Hagberg, A. A., Schult, D. A., and Swart, P. J.: Exploring network structure, dynamics, and function using NetworkX, *Proceedings of the 7th Python in Science Conference (SciPy2008)*, Pasadena, CA USA, August 2008, pp. 11–15.

[6] Hebrard, E., O'Mahony, E., and O'Sullivan, B.: Constraint Programming and Combinatorial Optimisation in Numberjack, *Proceedings of the 7th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR 2010), LNCS 6140*, 2010, pp. 181–185.

[7] 井上克巳, 田村直之: SAT ソルバーの基礎, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 57–67.

[8] Inoue, T., Iwashita, H., Kawahara, J., and Minato, S.-i.: Graphillion: software library for very large sets of labeled graphs, *International Journal on Software Tools for Technology Transfer*, (2014), pp. 1–10.

[9] Inoue, T., Yasuda, N., Kawano, S., Takenobu, Y., Minato, S., and Hayashitakeru, Y.: Distribution Network Verification for Secure Restoration by Enu-

merating All Critical Failures, *IEEE Trans. Smart Grid*, Vol. 6, No. 2(2015), pp. 843–852.

[10] Iwashita, H. and Minato, S.: Efficient top-down ZDD construction techniques using recursive specifications, Technical Report TCS-TR-A-13-69, Division of Computer Science, Hokkaido University, 2013.

[11] Kawahara, J., Inoue, T., Iwashita, H., and Minato, S.: Frontier-based search for enumerating all constrained subgraphs with compressed representation, Technical Report TCS-TR-A-14-76, Division of Computer Science, Hokkaido University, 2014.

[12] Metodi, A. and Codish, M.: Compiling finite domain constraints to SAT with BEE, *Theory and Practice of Logic Programming*, Vol. 12, No. 4-5(2012), pp. 465–483.

[13] Minato, S.: Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems, *DAC*, 1993, pp. 272–277.

[14] 鍋島英知: SAT によるプランニングとスケジューリング, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 114–121.

[15] 鍋島英知, 宋剛秀: 高速 SAT ソルバーの原理, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 68–76.

[16] Naveh, B. et al.: Jgrapht, *Internet: http://jgrapht. sourceforge. net*, (2008).

[17] Siek, J. G., Lee, L., and Lumsdaine, A.: *The Boost Graph Library - User Guide and Reference Manual*, C++ in-depth series, Pearson / Prentice Hall, 2002.

[18] Sinz, C.: Towards an optimal CNF encoding of boolean cardinality constraints, *Principles and Practice of Constraint Programming-CP 2005*, Springer, 2005, pp. 827–831.

[19] Soh, T., Berre, D. L., Roussel, S., Banbara, M., and Tamura, N.: Incremental SAT-Based Method with Native Boolean Cardinality Handling for the Hamiltonian Cycle Problem, *Logics in Artificial Intelligence - 14th European Conference, JELIA 2014, Funchal, Madeira, Portugal, September 24-26, 2014. Proceedings*, 2014, pp. 684–693.

[20] Soh, T., Tamura, N., and Banbara, M.: Scarab: A Rapid Prototyping Tool for SAT-based Constraint Programming Systems, *Proceedings of the 16th International Conference on Theory and Applications of Satisfiability Testing (SAT 2013), LNCS 7962*, July 2013, pp. 429–436.

[21] Stojadinovic, M. and Maric, F.: meSAT: multiple encodings of CSP to SAT, *Constraints*, Vol. 19, No. 4(2014), pp. 380–403.

[22] Tamura, N., Taga, A., Kitagawa, S., and Banbara, M.: Compiling Finite Linear CSP into SAT, *Constraints*, Vol. 14, No. 2(2009), pp. 254–272.

[23] 田村直之, 丹生智也, 番原睦則: 制約最適化問題と SAT 符号化, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 77–85.

[24] Tanjo, T., Tamura, N., and Banbara, M.: Azucar: A SAT-Based CSP Solver Using Compact Order Encoding — (Tool Presentation), *Proceedings of the 15th International Conference on Theory and Applications of Satisfiability Testing (SAT 2012), LNCS 7317*, Springer, June 2012, pp. 456–462.

[25] Velev, M. N. and Gao, P.: Efficient sat techniques for relative encoding of permutations with constraints, *AI 2009: Advances in Artificial Intelligence*, Springer, 2009, pp. 517–527.