1M1-OS-02a-2

SAT技術を用いたペトリネットのデッドロック検出手法の提案

Proposal of SAT-based Method to Detect Deadlock of Petri nets

寸田 智也 *1 Tomoya Sunda　宋 剛秀 *2 Takehide Soh　番原 睦則 *2 Mutsunori Banbara　田村 直之 *2 Naoyuki Tamura

*1神戸大学大学院システム情報学研究科 Graduate School of System Informatics, Kobe University

*2神戸大学情報基盤センター Information Science and Technology Center, Kobe University

In this paper, we introduce SAT-based method to detect deadlock of Petri nets which has more than one tokens. We represent the transition relation of a Petri net as constraints on integers and encode them into SAT by order encoding. We propose four constraint models (basic model, restricted model, multiple firing model and integrated model) for deadlock detection. In restricted model, the amount of token changes in each place in each step is bound by arc weights, so restricted model can be encoded into smaller SAT instance than the one generated by basic model. Multiple firing model allows more than one firings of each transition in each step and integrated model allows the firing of successive transitions in some specified order. Both models can detect deadlock with shorter steps than basic model. We evaluate these four constraint models and compared with exising methods through benchmark set of Model Checking Contest 2015. We can detect deadlocks of Petri nets that LoLA2, winner tool of Model Checking Contest 2015, failed to detect.

1. はじめに

C. A. Petri によって提唱されたペトリネットは，コンピュータ・通信システムなどの並行的，非同期的，分散的，非決定的な動作のモデルであり，一般的な情報・制御システムの記述・設計・解析・検証に有用である [奥川 95].

命題論理の充足可能性判定 (SAT) は，与えられた命題論理式が真になる値割り当てが存在するかどうかを判定する問題である [井上 10]．この SAT を解くソルバーである，SAT ソルバーの性能は，近年飛躍的に向上しており，ペトリネットの検証にも応用されている．しかし，SAT ソルバーでは，直接的には整数変数を扱うことができないため，既存の SAT 型手法の多くは，各プレイスのトークン数が 1 以下の安全ペトリネットのみを対象としている [Ogata 04].

そこで本稿では，ペトリネットの遷移を整数上の制約として記述し，順序符号化 [Tamura 09] を用いて SAT 符号化することで，トークン数が整数値である一般のペトリネットでのデッドロック検出を行う．順序符号化は，整数変数上の制約充足問題を SAT 問題に符号化する手法の 1 つで，多くの問題に対してよい性能を示している．

ペトリネットの検証には，有界モデル検査 [番原 10, Biere 09, Biere 99] の手法を用いる．有界モデル検査では，検証するステップ長の値を増やしつつ，SAT ソルバーを繰り返し起動して問題を解く．本稿では，SAT に変換した時の節数と，短いステップ長でデッドロックを検出することに着目し，4 つの制約モデル (基本モデル，制限モデル，多重発火モデル，統合モデル) を提案する．このうち最初の 3 つに関しては，文献 [寸田 16] で正規ペトリネット向けに提案された制約モデルであり，本稿ではそれを多重度に適用可能な制約モデルとして拡張する．統合モデルは文献 [Ogata 04] の手法を取り入れて，2 個以上のトークンや多重度ありの場合に適用できるよう拡張した制約モデルである．

連絡先: 神戸大学大学院システム情報学研究科，神戸市灘区六甲台町 1-1, 078-803-5365, sunda@stu.kobe-u.ac.jp

2. ペトリネットとデッドロック

2.1 ペトリネットの定義

ペトリネットは，組 $(\mathbf{P}, \mathbf{T}, F, M_0)$ で与えられる．ここで，$\mathbf{P}$ はプレイスの有限集合，$\mathbf{T}$ はトランジションの有限集合，$F : (\mathbf{P} \times \mathbf{T}) \cup (\mathbf{T} \times \mathbf{P}) \to \mathbb{N}$ はアークの重み (アークがない場合は 0) を与える関数，$M_0 : \mathbf{P} \to \mathbb{N}$ は初期マーキングである．また，$\bullet p$ でプレイス p の入力トランジションの集合，$p\bullet$ で出力トランジションの集合，$\bullet t$ でトランジション t の入力プレイスの集合，$t\bullet$ で出力プレイスの集合を表す．ペトリネットの状態とはマーキング $M : \mathbf{P} \to \mathbb{N}$ を意味する．

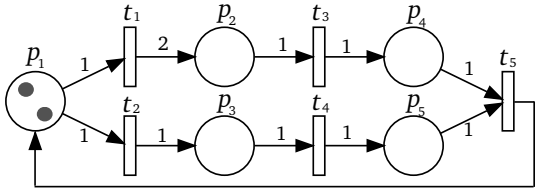

図 1: ペトリネットの例

2.2 トランジションの発火

ペトリネットはトランジションの発火により，次のマーキングへ遷移する．トランジション t は，すべての入力プレイスがアークの多重度以上のトークンを持つとき，発火可能となる．

トランジション t の発火でマーキング M から M' に遷移することを，$M \xrightarrow{t} M'$ で表す．また，トランジション t が n 回発火してマーキング M から M' に遷移することを，$M \xrightarrow{t^n} M'$ と表す．$M \xrightarrow{t^n} M'$ は以下のように定義できる．

$$M \xrightarrow{t^n} M' \overset{def}{\Longleftrightarrow} \forall p \in \mathbf{P} \quad (M(p) \geq nF(p,t) \wedge M'(p) = M(p) + n(F(t,p) - F(p,t))) \tag{1}$$

2.3 可達性

初期マーキング M からマーキング M' へ到達可能であるようなトランジションの発火系列が存在するとき，マーキング M' は M から可達であるといい，$M \rightarrow^* M'$ で表す．

2.4 デッドロック

デッドロックとは，すべてのトランジションが発火可能でないようなマーキングを表す．また，デッドロックが初期マーキングから可達であるとき，そのペトリネットはデッドロックを持つという．

ペトリネットがデッドロックを持つことは以下のように表すことができる．

$$\exists M(M_0 \rightarrow^* M \ \wedge \ \forall t \in \mathbf{T} \ \exists p \in {\bullet t} \ (M(p) < F(p,t)) \quad (2)$$

3. ペトリネットの制約モデルの提案

本節では，ペトリネットの可達性を検証するための制約モデルについて説明する．まず，ペトリネットの遷移をいくつか定義する．そして，それらを制約モデルとして記述する．

3.1 多重発火モデル

多重発火モデルは，多重発火，すなわち，トランジションが 1 度の遷移で複数回発火することを許した制約モデルである．

$E_{\mathrm{M}}(M)$ は，マーキング M において，同時に発火可能なトランジションの多重集合の集合を表す．

$$E_{\mathrm{M}}(M) := \{\{t_1^{n_1}, \cdots, t_m^{n_m}\} | \{t_1, \cdots, t_m\} \subseteq \mathbf{T}, \ n_1, \cdots, n_m \in \mathbb{N}, \\ \forall p \in \mathbf{P} \ M(p) \geq \sum_{i=1}^{m} n_i F(p, t_i)\} \quad (3)$$

M から M' へ，多重発火可能なペトリネットの遷移を $M \rightarrow_{\mathrm{M}} M'$ と表し，以下のように定義する．

$$M \rightarrow_{\mathrm{M}} M' \stackrel{def}{\Longleftrightarrow} \\ \exists \{t_1^{n_1}, \cdots, t_m^{n_m}\} \in E_{\mathrm{M}}(M) \ \ (M \xrightarrow{t_1^{n_1}} \cdots \xrightarrow{t_m^{n_m}} M') \quad (4)$$

$M \rightarrow_{\mathrm{M}} M'$ のとき，定義から $M \rightarrow^* M'$ である．また，$M \rightarrow^* M'$ のとき，$M \xrightarrow{t_1^{n_1}} \cdots \xrightarrow{t_m^{n_m}} M' \Leftrightarrow M \rightarrow_{\mathrm{M}} \cdots \rightarrow_{\mathrm{M}} M'$ であるので，$M \rightarrow_{\mathrm{M}} M'$ でも可達性は変わらない．

$M \rightarrow_{\mathrm{M}} M'$ に対応する制約モデルを多重発火モデルと呼ぶ．各プレイスのトークン数の上限を $N(p)$，トランジション t の発火回数の上限を $N(t)$ と表す．まず，i ステップ目のプレイス p のトークン数を表す変数 $m_p^i \in \{0, \cdots, N(p)\} \quad (p \in \mathbf{P}, \ 0 \leq i \leq k)$ を導入する．m_p^i は $M(p)$ に，m_p^{i+1} は $M'(p)$ に対応する．また，i ステップ目でトランジション t が発火する回数を表す変数 $f_t^i \in \{0, \cdots, N(t)\} \quad (t \in \mathbf{T}, \ 0 \leq i < k)$ を導入する．

これらの元で，$M \rightarrow_{\mathrm{M}} M'$ に対応する制約を考える．$\tau = \{t_1^{n_1}, \cdots, t_m^{n_m}\} \in E_{\mathrm{M}}(M)$ $M \xrightarrow{t_1^{n_1}} \cdots \xrightarrow{t_m^{n_m}} M'$ とする．$\tau \in E_{\mathrm{M}}(M)$ の定義から，以下の制約を加える．

$$\bigwedge_{p \in \mathbf{P}} (m_p^i \geq \sum_{t \in p\bullet} F(p,t) \cdot f_t^i) \quad (5)$$

次に，$M = M_1 \xrightarrow{t_1^{n_1}} \cdots \xrightarrow{t_m^{n_m}} M_{m+1} = M'$ とする．これは，任意の $p \in \mathbf{P}, \ 1 \leq j \leq m$ に対して，

$$M_j(p) \geq n_j F(p, t_j) \quad (6)$$

$$M_{j+1}(p) = M_j(p) + n_j(F(t_j, p) - F(p, t_j)) \quad (7)$$

が成り立つことと同値である．式 (6) は式 (5) が成り立つとき，常に成り立つので，式 (6) に対応する制約は省略する．式 (7) より，以下の制約を加える．

$$\bigwedge_{p \in \mathbf{P}} (m_p^{i+1} = m_p^i + \sum_{t \in \bullet p} F(t,p) \cdot f_t^i - \sum_{t \in p\bullet} F(p,t) \cdot f_t^i) \quad (8)$$

3.2 基本モデル

基本モデルでは，同じトランジションが一度の遷移で発火できる回数を 1 回にした制約モデルである．基本モデルでは，多重発火モデルに $f_t^i \leq 1 \ (t \in \mathbf{T}, 0 \leq i < k)$ の制約を加える．

3.3 制限モデル

制限モデルでは，基本モデルに同時発火できるトランジションに制限を加えた制約モデルである．制限モデルでは，基本モデルに以下を加える．

$$\bigwedge_{p \in \mathbf{P}} (\sum_{t \in \bullet p} F(t,p) \cdot f_t^i \leq \max_{t \in \bullet p} F(t,p) \\ \sum_{t \in p\bullet} F(p,t) \cdot f_t^i \leq \max_{t \in p\bullet} F(p,t)) \quad (9)$$

基本モデル，制限モデルでは，多重発火モデルに制限が加えられているが，可達性は変わらない．

3.4 統合モデル

統合モデルは，文献 [Ogata 04] の手法を取り入れて，各プレイスのトークン数が 2 以上の場合や多重度に対応できるよう拡張した制約モデルである．

トランジションを適当な順序で並べたものを $t_1, \cdots, t_l$ とする $(l = |\mathbf{T}|)$．この時，ペトリネットの遷移 $M \rightarrow_{\mathrm{I}} M'$ を以下のように定義する．

$$M \rightarrow_{\mathrm{I}} M' \stackrel{def}{\Longleftrightarrow} \\ \exists n_1, \cdots, n_l \in \mathbb{N} \quad (M \xrightarrow{t_1^{n_1}} \cdots \xrightarrow{t_l^{n_l}} M') \quad (10)$$

このようにペトリネットの遷移を定義しても，可達性は変わらない．

$M \rightarrow_{\mathrm{I}} M'$ に対応する制約モデルを統合モデルと呼ぶ．変数には

$$m_p^{il+j} \in \{0, \cdots, N(p)\} \quad (p \in \mathbf{P}, \ 0 \leq i < k, \ 1 \leq j \leq l+1) \\ f_t^i \in \{0, \cdots, N(t)\} \quad (t \in \mathbf{T}, \ 0 \leq i < k) \quad (11)$$

を用いる．f_t^i はステップ i でトランジション t が発火した回数を表す．また，$M = M_1 \xrightarrow{t_1^{n_1}} \cdots \xrightarrow{t_l^{n_l}} M_{l+1} = M'$ とした時，m_p^{il+j} は $M_j(p)$ に対応する．$(1 \leq j \leq l+1)$

$$\begin{aligned} & M_j \xrightarrow{t_j^{n_j}} M_{j+1} \\ & \Leftrightarrow \bigwedge_{p \in {\bullet t_j} \setminus t_j \bullet} m_p^{il+j+1} = m_p^{il+j} - F(p, t_j) f_{t_j}^i \\ & \quad \wedge \bigwedge_{p \in t_j \bullet \setminus {\bullet t_j}} m_p^{il+j+1} = m_p^{il+j} + F(t_j, p) f_{t_j}^i \\ & \quad \wedge \bigwedge_{p \in {\bullet t_j} \cap t_j \bullet} m_p^{il+j} \geq F(p, t_j) f_{t_j}^i \\ & \qquad \wedge m_p^{il+j+1} = m_p^{il+j} + (F(t_j, p) - F(p, t_j)) f_{t_j}^i \\ & \quad \wedge \bigwedge_{p \in \mathbf{P} \setminus ({\bullet t_j} \cup t_j \bullet)} m_p^{il+j+1} = m_p^{il+j} \end{aligned} \quad (12)$$

$p \in \mathbf{P} \backslash (\bullet t_j \cup t_j \bullet)$ については，m_p^{il+j+1} と m_p^{il+j} を同じ変数とみなし，置き換えることによって省略することができる．

統合モデルは，多重発火モデルより短いステップ長で目的の状態に到達できる．

3.5 可達性の検証

提案手法では，有界モデル検査の手法を用いて可達性を検証する．有界モデル検査では，初期マーキングから k 回以内の遷移で，目的のマーキングに到達するかどうかを調べる．もし，k 回以内で目的のマーキングに到達できなければ，k の値を増加させ，再び調べる．

3.6 トランジションの発火順序

基本モデル，制限モデル，多重発火モデルでは，トランジションの発火順序によらず，デッドロック検出に必要なステップ長は同じである．しかし，統合モデルでは，トランジションの発火順序によって必要なステップ長が変わる．

例 1 の場合，t_1, t_2, t_3, t_4, t_5 の順に考えると，t_1 が 2 回，t_3 が 4 回発火でデッドロックとなるので，1 回の遷移でよい．しかし，t_5, t_4, t_3, t_2, t_1 の順に考えると，少なくとも 2 回遷移しなければならない．

そこで，トランジションの発火順序として，トークンを持つプレイスから深さ優先探索でグラフをたどり，到達した順番を用いた．

3.7 トークン数上限 $N(p)$ と発火回数上限 $N(t)$ の決め方

整数変数を SAT 符号化する場合，その変数のドメインが定まっていなければならない．しかし，ペトリネットの各プレイスのトークン数の上限を見積もるのは困難である．本研究では，マーキングインバリアントを用いることで各プレイスのトークン数の上限を大まかに見積もることにした．

マーキングインバリアントは非負整数ベクトルであり，これにより重み付けされたトークン数の合計は常に一定となる．

マーキングインバリアント $y = (q_1, \cdots, q_{|\mathbf{P}|})$，$yM = c$ とする．このとき，各プレイス p_i $(q_i \neq 0)$ のトークン数は c/q_i 以下となる．すべてのマーキングインバリアントに対し，上記のことを行うことで，トークン数上限 $N(p)$ を見積もることができる．なお，$N(p)$ が求まらない場合は，$N(p) = M_0(p)$ とした．

多重発火モデル･統合モデルでは，各トランジションの発火回数の上限を設定する必要がある．トランジション t が n 回発火するとすると，入力プレイス p のトークン数は $nF(p,t)$ 個減り，出力プレイス p のトークン数は $nF(t,p)$ 個加わる．入力プレイスから減るトークン数は，そのプレイスのトークン数上限以下でなければならないので，$\forall p \in \bullet t \quad nF(p,t) \leq N(p)$ となる．また，出力プレイスに加わるトークン数も，そのプレイスのトークン数上限以下でなければならないので，$\forall p \in t\bullet \quad nF(t,p) \leq N(p)$ となる．したがって，$N(t)$ は以下のように設定できる．

$$N(t) = \min\{\min_{p \in \bullet t}(N(p)/F(p,t)), \min_{p \in t\bullet}(N(p)/F(t,p))\} \quad (13)$$

4. 評価実験

提案手法の有効性を検証するため，Model Checking Contest 2015 のデッドロック検出部門の問題 267 問のうち，デッドロックがあることが分かっている問題 120 問を用いて性能評価を行った．提案した 4 種類の制約モデルを Sugar を用いて SAT 符号化し，SAT ソルバー GlueMiniSat を用いてデッドロックを探索する．予備実験として，ステップ数の増やし方を 1 ずつ増やす場合，1.5 倍ずつ増やす場合，2 倍ずつ増やす場合の 3 通りを比較した．その結果，すべての制約モデルで 2 倍ずつ増やす場合が最も結果がよくなった．評価実験では，すべてステップ数を 2 倍ずつ増やした結果を用いる．

ベンチマークは Ubuntu 14.04 (Xeon 3.5GHz, メモリ 8GB) 上で計測した．

4.1 提案する制約モデル 4 つの比較

提案した 4 つの制約モデル (基本モデル，制限モデル，多重発火モデル，統合モデル) が時間内にデッドロックを検出できた問題数と，検出にかかったステップ長を表 1 に示す．N の値は初期マーキングの各プレイスのトークン数の最大値である．N の値ごとに結果を示す．

4 つの制約モデルすべてを比較すると，トークン数 N が大きくなるに連れて，多重発火モデル・統合モデルがほかの制約モデルに比べて短いステップ長でデッドロックを検出できていることがわかる．これは，多重発火を認めたことにより，一度の遷移で多くのトークンを運ぶことができるためだと考えられる．また，$N = 6 \sim 10, 50 \sim 500$ では，統合モデルがほかの制約モデルよりはるかに短いステップ長で検出できており，結果も最もよかった．デッドロック検出にかかるステップ長が短く済むことが，性能向上に影響していることがわかる．

また，制限モデルのほうがよい結果を出す場合もあった．これは，制限モデルのほうが SAT に変換したとき，より少ない節数で表すことができるためであると考えられる．

4.2 既存手法や SMT ソルバー，PB ソルバー等を用いた場合との比較

提案した制約モデルを他の手法で実装した場合と比較する．また，Model Checking Contest 2015 デッドロック検出部門で優勝したツールである LoLA2 とも比較する．

図 2 は，多重発火モデルと統合モデルをさまざまな手法で実装した場合と LoLA2 を用いた場合に，デッドロック検出にかかった CPU 時間を示す.「glueminisat (order enc)」は順序符号化して SAT ソルバーで求解した場合の結果,「clasp (log enc)」は PB ソルバー Clasp で対数符号化を用いて求解した場合の結果,「z3」は SMT ソルバー Z3 で求解した場合の結果,「clasp (hybrid enc)」はハイブリッド符号化を用いた場合の結果,「LoLA2」は LoLA2 での結果を示す．ただし,「multiple」は多重発火モデル,「integrated」は統合モデルを用いたことを表す.

提案した制約モデルの実装方法としては，順序符号化を用いて SAT ソルバーで求解した場合が最も良い結果となった．制約モデルでは，z3 のときのみ多重発火モデルのほうが良い結果となり，それ以外の実装方法で統合モデルが良い結果となった．

既存手法との比較では，LoLA2 が最も良い結果となった．しかし，提案手法は LoLA2 が解くことができなかった問題 4 問を解くことができた．また，そのうち 1 問「Angiogenesis-PT-50」は Model Checking Contest 2015 でいずれのツールもデッドロックを検出できなかった問題である．この問題は可達な状態数が非常に多く，LoLA2 ではメモリーオーバーしていた．LoLA2 はペトリネットの状態を展開していく手法を用いているが，この手法は状態空間爆発が起こりやすい．一方で SAT 型検証手法では，他の手法に比べ状態空間爆発が起こりにくいため，状態数が多い問題でも解くことができると考えられる．

表 1: 各製薬モデルで検出できた問題数と平均ステップ数

N		検出できた問題数 基本	制限	多重	統合	平均ステップ長 基本	制限	多重	統合
1	(58)	44	45	45	41	11.4	11.9	11.9	7.2
2 ∼ 5	(10)	9	9	10	9	9.1	12.4	9.8	3.8
6 ∼ 10	(35)	11	11	11	18	10.5	14.5	7.6	1.6
11 ∼ 20	(7)	6	6	6	6	17.0	24.0	9.0	2.5
21 ∼ 49	(4)	3	4	4	4	4.7	26.0	7.5	1.8
50 ∼ 500	(6)	0	0	4	4	—	—	16.0	4.0
合計	(120)	73	75	80	82	11.2	14.1	10.8	4.8

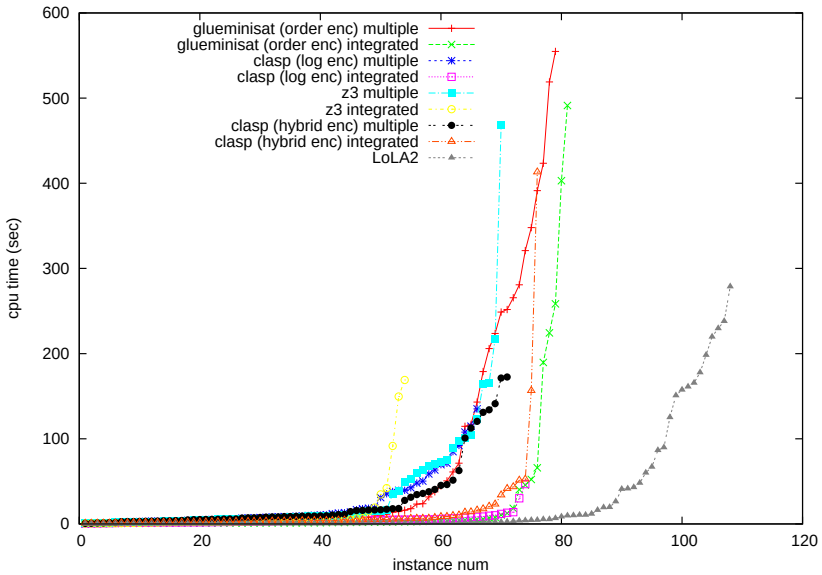

図 2: 各手法での実験結果

5. まとめ

本稿では，SAT 技術を用いて一般のペトリネットのデッドロック検出を行った．文献 [寸田 16] の制約モデルを元に，多重度に対応した制約モデル，基本モデル，制限モデル，多重発火モデルを提案し，文献 [Ogata 04] の手法を安全でないペトリネットに拡張した統合モデルを提案した．これらを評価した結果，デッドロック検出に必要なステップ数が性能に大きく寄与することがわかった．制限モデルは，他の制約モデルに比べて少ない節数で表現できるため，よい性能を示すことがあった．多重発火モデルや統合モデルは，多重発火を認めたことでトークン数の上限が大きい問題に対してより少ない遷移回数でデッドロックを検出できることが確かめられた．

提案手法の実装方法としては，順序符号化と SAT ソルバーを用いる場合が最も良い結果となり，ほとんどの実装方法で統合モデルが最もよい結果であった．

既存手法との比較では，Model Checking Contest 2015 デッドロック検出部門で優勝したツール LoLA2 と比較した．デッドロックを検出できた問題数では LoLA2 より少なかったが，LoLA2 でメモリーオーバーした問題に対してもデッドロックを検出でき，提案手法の有効性が確認できた．

参考文献

[番原 10] 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, pp. 122–129 (2010)

[Biere 99] Biere, A., Cimatti, A., Clarke, E. M., and Zhu, Y.: Symbolic Model Checking without BDDs, in *Proceedings of the 5th International Conference on Tools and Algorithms for Construction and Analysis of Systems (TACAS 1999), LNCS 1579*, pp. 193–207 (1999)

[Biere 09] Biere, A.: Bounded Model Checking, in *Handbook of Satisfiability*, pp. 457–481, IOS Press (2009)

[井上 10] 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, pp. 57–67 (2010)

[Ogata 04] Ogata, S., Tsuchiya, T., and Kikuno, T.: SAT-Based Verification of Safe Petri Nets, in *Automated Technology for Verification and Analysis: Second International Conference, ATVA 2004, Taipei, Taiwan, ROC, October 31-November 3, 2004. Proceedings*, pp. 79–92 (2004)

[奥川 95] 奥川 峻史 ペトリネットの基礎, 共立出版 (1995)

[Tamura 09] Tamura, N., Taga, A., Kitagawa, S., and Banbara, M.: Compiling Finite Linear CSP into SAT, *Constraints*, Vol. 14, No. 2, pp. 254–272 (2009)

[寸田 16] 寸田 智也, 宋 剛秀, 番原 睦則, 田村 直之 SAT 技術を用いた正規ペトリネットのデッドロック検出手法の提案, 33 , pp. 513–521, 日本ソフトウェア科学会 (2016)