B-027

# 時間ペトリネットでモデル化された GALS システムを対象とした UPPAAL による自動検証手法
# Formal Verification of GALS Systems modeled by Timed Petri Nets using UPPAAL

三輪 陽介 † 横川 智教 † 宮崎 仁 ‡ 近藤 真史 † 佐藤 洋一郎 †
Yosuke MIWA Tomoyuki YOKOGAWA Hisashi MIYAZAKI Masafumi KONDO Yoichiro SATO

## 1. まえがき

本論文では，実時間システムを対象としたモデル検査ツールの一種である UPPAAL [1] を用いて，大域非同期局所同期 (GALS) システムの特性を検証する手法を提案する．ここでは，検証対象となる GALS システムは時間ペトリネット (TdPN) でモデル化されていると仮定する．本手法では，与えられた TdPN を UPPAAL の入力形式である拡張時間オートマトン (XTA) へと変換することで検証を行う．まず，TdPN の各プレイスの振る舞いをそれぞれ XTA によって表現し，その上でトランジションによる接続関係をもとに各 XTA の動作を同期することで，TdPN 全体の振る舞いを表現する．

## 2. モデル

### 2.1 TdPN

TdPN は，プレイス $P$，トランジション $T$，アーク $F \subseteq P \times T \cup T \times P$，プレイス遅延の最大値 $x_{\max} : P \to (0, \infty]$ および最小値 $x_{\min} : P \to [0, \infty)$，初期マーキング $M_0 \subset P$ から定義される．各プレイス $p$ のトークンは獲得後 $x_{\min}(p)$ から $x_{\max}(p)$ が経過するまでのいずれかのタイミングで有効となる．全ての入力プレイスのトークンが有効となった瞬間，トランジションは発火する．

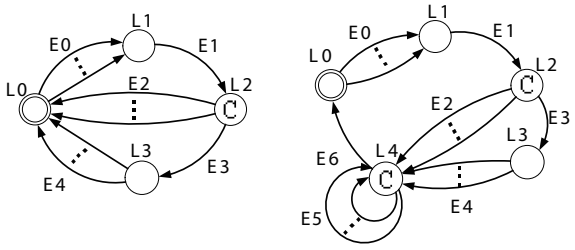
抑止アーク $F_{in} \subseteq P \times T$ は特殊なアークであり，$\langle p, t \rangle \in F_{in}$ に対して $p$ のトークンが有効であるとき，$t$ は発火できない．このとき $p$ を $t$ の抑止プレイス，$t$ を $p$ の被抑止トランジションという．

プレイス $p$ の入力 (出力) トランジションを $\bullet p (p\bullet)$ と記し，トランジション $t$ の入力 (出力) プレイスを $\bullet t (t\bullet)$ と記す．$t$ の抑止プレイスを $\circ t$ と記し，$p$ の被抑止トランジションを $p\circ$ と記す．

### 2.2 XTA

XTA は，標準的な時間オートマトンに対して UPPAAL 独自の拡張を行ったものである．XTA は状態と遷移，そして変数から構成される．状態は不変条件を，遷移は変数値の更新アクションとガード条件をもつことができる．通常の変数のほかに，時間経過とともに増加するクロック変数が使用できる．また特殊な状態として，時間経過および他の遷移の割込みがない *committed* 状態が使用できる．XTA 間の遷移は，同期チャネル (送信：`c!`，受信：`c?`) で同期する．

†岡山県立大学 大学院情報系工学研究科
‡川崎医療福祉大学 医療技術学部 臨床工学科




(a)通常のプレイス (b)抑止プレイス

図 1: プレイスを表現する XTA の構造

## 3. TdPN から XTA への変換

本手法では，各プレイスの振る舞いを 1 つの XTA で表現し，トランジションの発火によるトークンの移動を XTA 間の同期やアクションで表現する．まず，プレイス $p_i$ に対して，経過時間を表すクロック変数 `x_i` と有効なトークンをもつことを表す二値変数 `p_i` を定義し，トランジション $t_j$ に対して，$t_j$ の発火を表す同期チャネル `t_j` を定義する．その上で，定義した変数および同期チャネルを用いて各プレイスの振る舞いを XTA で表現する．以下，プレイスの XTA 表現法を，通常のプレイスの場合と抑止プレイスの場合に分けてそれぞれ述べる．

### 3.1 通常のプレイス

プレイスの動作は，図 1(a) に示す構造をもつ XTA で表現する．$p_i$ は，$t_k \in \bullet p_i$ の発火によりトークンを獲得する．そして $x_{\min}(p_i)$ から $x_{\max}(p_i)$ が経過するまでにトークンは有効となる．トークンが有効となった瞬間に $t_j \in p_i\bullet$ のいずれかが発火可能であれば，$t_j$ はただちに発火し，$p_i$ のトークンは消費される．発火可能なトランジションが存在しなければ，トークンをもったまま出力トランジションが発火可能となるまで待機する．以下，この XTA がもつ状態および遷移について述べる．

**L0:** トークンをもたない状態．

**E0:** $t_k \in \bullet p_i$ の発火によりトークンを獲得する．$t_k$ の発火を表す同期チャネル受信 `t_k?` とアクション `x_i:=0` をもつ．同期チャネルの受信は遷移に 1 つしか記述できないため，この遷移は $t_k$ ごとに用意する．

**L1:** 有効でないトークンをもつ状態．最大遅延値 $x_{\max}(p_i)$ を経過するまでにトークンは必ず有効となるため，不変条件 `x_i`$\leq x_{\max}(p_i)$ をもつ．

**E1:** トークンが有効となる．アクション `p_i:=1` をもつ．最小遅延値 $x_{\min}(p_i)$ を経過するまで有効とはなら

ないため，ガード条件 x_i≥ $x_{\min}(p_i)$ をもつ．

**L2:** トークンが有効となった瞬間の状態．もし出力トランジションが発火可能であればただちに発火が行われるため，*committed location* として定義する．

**E2:** $t_j \in p_i\bullet$ の発火によりトークンが消費される．この遷移は $t_j$ ごとに用意する．$t_j$ の発火を表す同期チャネル送信 t_j!とアクション p_i:=0 をもつ．また，$t_j$ が発火可能であることを表すガード条件 $\bigwedge_{p_l \in \bullet t_j \setminus p_i}(\mathtt{p\_l} == 1) \wedge \bigwedge_{p_m \in \circ t_j}(\mathtt{p\_m} == 0)$ をもつ．

**E3:** $t_j \in p_i\bullet$ が全て発火不可能であったときの分岐を表す．ガード条件 $\bigwedge_{t_j \in p_i\bullet}\{\neg(\bigwedge_{p_l \in \bullet t_j \setminus p_i}(\mathtt{p\_l} == 1) \wedge \bigwedge_{p_m \in \circ t_j}(\mathtt{p\_m} == 0))\}$ をもつ．

**L3:** 有効なトークンをもつが，出力トランジションがいずれも発火可能でない状態．

**E4:** 他のプレイスのトークンが有効になることで $t_j \in p_i\bullet$ が発火し，トークンを消費する．この遷移は $t_j$ ごとに用意する．同期チャネル受信 t_j?およびアクション p_i:=0 をもつ．

### 3.2 抑止プレイス

抑止プレイスの動作は図 1(b) に示す構造をもつ XTA で表現する．$p_i$ のトークンが有効ならば，$t_h \in p_i\circ$ は発火できない．$p_i$ のトークンが消費された瞬間にいずれかの $t_h$ が発火可能であれば，$t_h$ はただちに発火し，$p_i$ のトークンは消費される．以下，図 1(a) の XTA に対して追加もしくは変更された状態および遷移について述べる．

**L4:** トークンが消費された瞬間の状態．もし被抑止トランジションが発火可能であればただちに発火が行われるため，*committed location* として定義する．

**E5:** トークンの消費により $t_h \in p_i\circ$ が発火する．この遷移は $t_h$ ごとに用意する．$t_h$ の発火を表す同期チャネル送信 t_h!をもつ．$t_h$ が発火可能であることを表すガード条件 $\bigwedge_{p_l \in \bullet t_h}(\mathtt{p\_l} == 1) \wedge \bigwedge_{p_m \in \circ t_h \setminus p_i}(\mathtt{p\_m} == 0)$ をもつ．

**E6:** $t_h \in p_i\circ$ が全て発火不可能であれば何も処理は行わないため，ガード条件 $\bigwedge_{t_h \in p_i\circ}\{\neg(\bigwedge_{p_l \in \bullet t_h}(\mathtt{p\_l} == 1) \wedge \bigwedge_{p_m \in \circ t_h \setminus p_i}(\mathtt{p\_m} == 0))\}$ をもつ．

## 4. 比較実験

本節では，TdPN を対象としたモデル検査ツールである Roméo[2] との比較実験を行う．Roméo は TdPN を対象とした検証が可能であり，さらに TdPN から UPPAAL の入力記述を生成する機能をもつ．そこで本実験では，GALS システムをモデル化した TdPN に対して，提案法および Roméo で生成された XTA を用いた UPPAAL による検証，そして Roméo による TdPN を対象とした検証に対して，検証コストに関する比較を行う．

### 4.1 検証コストの比較

本実験では，27 のプレイスと 26 のトランジションからなる TdPN でモデル化された小規模な GALS システムが，デッドロックをもたないことについて検証を行った．検証時間の比較結果を表 1 に示す．提案法で生成した XTA を用いた検証と，Roméo を用いた検証では，どちらも非常に短い時間で検証が完了し，差は現れなかった．また，Roméo で生成された XTA を用いた検証では，メモリ不足により結果が得られなかった．生成された XTA のサイズに関する比較結果を表 2 に示す．Roméo で生成された XTA のサイズは，提案法と比べて，個数，状態数，遷移数のいずれについても大きくなる．

表 1: 検証時間の比較

| 検証手法 | 検証時間 | 検証結果 |
|---|---|---|
| 提案法+UPPAAL | 0.5[Sec] | True |
| Roméo+UPPAAL | N/A | N/A |
| Roméo | 0.5[Sec] | True |

表 2: 生成された XTA サイズの比較

| 変換手法 | XTA 数 | 状態数 | 遷移数 |
|---|---|---|---|
| 提案法 | 27 | 112 | 250 |
| Roméo | 54 | 163 | 412 |

### 4.2 考察

比較実験の結果，提案法に基づく検証と Roméo による検証のコストはほぼ同等であった．Roméo のクエリ言語は UPPAAL と異なり，クロックの値などを直接扱うことができないため，応答時間等の GALS システムの性能に関する性質を検証することは不可能である．従って，GALS システムを対象とした検証を行う上では，検証ツールとして UPPAAL を用いることが望ましい．しかしながら，Roméo が生成する XTA は提案法に比べて非常にサイズが大きく，検証コストも大きくなる．そのため，Roméo と比較した場合，提案法と UPPAAL を組み合わせて検証を行う手法がより有用だと考えられる．

## 5. あとがき

本稿では，TdPN でモデル化された GALS システムを XTA へと変換し，UPPAAL で検証する手法を提案した．さらに，既存の検証手法との比較を行い，同等の検証性能を実現できることを示した．今後の課題として，さらに大規模な GALS システムに対して提案法を適用し，手法の有効性を示す必要がある．

## 参考文献

[1] *UPPAAL*. http://www.uppaal.com, 2004.

[2] G. Gardey, D. Lime, M. Magnin, and O. H. Roux. Roméo: A tool for analyzing time petri nets. In *CAV'05, Computer Aided Verification, LNCS 3576*, pages 418–423. Springer, 2005.