CSP を用いた状態マシン図とシーケンス図の整合性検証

横川 智教[†1] 片山 巧[†1] 宮崎 仁[†2]
佐藤 洋一郎[†1] 有本 和民[†1]

本稿では，UML の状態マシン図およびシーケンス図をプロセス代数形式のモデル記述言語である CSP を用いて表現し，検証ツール FDR を用いて整合性検証を行う手法について示す．また，例題への適用実験を通して，整合性違反が存在する場合はそれを正しく検出できることを示している．

Consistency verification of state machine diagrams and a sequence diagram using CSP

Tomoyuki YOKOGAWA ,[†1] Takumi KATAYAMA ,[†1]
Hisashi MIYAZAKI ,[†2] Yoichiro SATO [†1]
and Kazutami ARIMOTO [†1]

We propose a method to detect consistency errors in UML state machine diagrams and sequence diagrams by representing the diagrams as CSP, which is a process algebra style notation. We use FDR model checker to verify the consistency relation between the diagrams. We show an application example of the method where the correctness of consistency of example diagrams is checked.

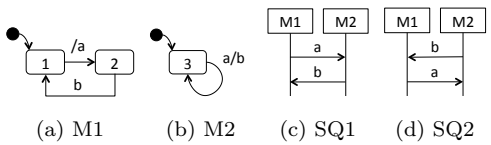
1. ま え が き

UML によるソフトウェアシステムの設計では，状態マシン図がシーケンス図の振る舞いを満たすこと，すなわち整合性を確認する必要があるが，この確認作業を人手で行うことは，全ての振る舞いを考慮する必要があるため非常に困難である．筆者らは，UML 図の振る舞いをプロセスとして表現し，プロセス間の弱模倣関係の判定によって UML 図間の整合性検証を行う手法を提案している[1),2)]．この手法では，階層をもつ状態マシン図およびシーケンス図を階層ごとにプロセスとして表現し，並行合成することにより，その振る舞いをモデル化していた．しかしながら，従来法で用いていたプロセス代数記述 FSP では複雑な階層構造を扱うことが困難である．そこで本稿では，プロセス代数形式のモデル記述言語である CSP を用いて状態マシン図およびシーケンス図の振る舞いを記述し，検証ツール FDR を用いて整合性検証を行う手法について示す．

†1 岡山県立大学
Okayama Prefectural University
†2 川崎医療福祉大学
Kawasaki University of Medical Welfare

2. UML 図の整合性

状態マシン図は，状態とそれらを結ぶ遷移によって構成される．遷移には，トリガおよびアクションとしてメッセージがそれぞれ割り当てられる．遷移はトリガの受信により実行され，状態の変化とともにアクションが送信される．また，初期状態として 1 つの状態が定められる．シーケンス図は，モジュールに対応したライフラインと，それらの間のメッセージ通信によって構成される．メッセージの送受信処理をオカレンスとよび，オカレンスの順序関係はライフライン上の位置で表現される．図 1(a)(b) に状態マシン図の例を，図 1(c)(d) にシーケンス図の例を示す．



(a) M1 (b) M2 (c) SQ1 (d) SQ2

図 1 状態マシン図とシーケンス図

状態マシン図とシーケンス図の振る舞いは，メッセージ処理系列 (実行とよぶ) の集合として表すことができる．図 1 の状態マシン図 M1 はメッセージ a を送

信する遷移とメッセージ b を受信する遷移を繰り返し，M2 は a を受信して b を送信する遷移を繰り返す．よって，メッセージ m の送受信を $m!$ と $m?$ で表すと，得られる実行は $a!a?b!b?\cdots$ の繰り返しのみとなる．シーケンス図からも同様に実行を求めることができ，SQ1 および SQ2 から得られる実行は $a!a?b!b?$ および $b!b?a!a?$ となる．

また，先行研究[2] と同じく，状態マシン図とシーケンス図の整合性は実行のプレフィックスの包含関係として定義する．図 1 の状態マシン図は，実行のプレフィックス $a!a?b!b?$ が SQ1 の実行と一致するので SQ1 との整合性は満たされるが，SQ2 の実行は $b!$ から開始するため SQ2 との整合性は満たされない．

3. CSP によるプロセス表現

CSP では，プロセスの動作をアクションの列として記述する．`Q=e->P` は，`Q` がアクション `e` の後にプロセス `P` として振る舞うプロセスであることを表す．プロセスの並行合成は `Q=P1[`$aP1$`||`$aP2$`]P2` と記述する．ここで，aP はプロセス `P` に現れるアクションの集合を表す．このとき，`Q` は `P1` と `P2` を共通するアクションで同期して実行するプロセスとなる．また，特定のアクション集合 $aX(\in aP1 \cap aP2)$ のみで同期して実行するプロセスは `Q=P1[|`aX`|]P2` と記述し，同期のないプロセス合成は `Q=P1|||P2` と記述する．

状態マシン図の振る舞いは，遷移によるメッセージ処理と状態の変化をプロセスとして記述し，並行合成することでプロセス表現できる．例として，状態マシン図 M1 および M2 の振る舞いは，それぞれを表すプロセス `M1` および `M2` の並行合成 `SMD` として以下のように表現できる．

```
M1 = S1
S1 = as->S2
S2 = br->S1
M2 = S3
S3 = ar->bs->S3
SMD = M1 ||| M2
```

ここで，`as`(`bs`) と `ar`(`br`) はそれぞれメッセージ $a(b)$ の送信と受信を表すイベントである．

また，メッセージは送信後に受信されるので，この振る舞いもプロセスとして表現する．

```
A = as->ar->A
B = bs->br->B
MSG = A ||| B
```

そして，それらをアクション {`as`,`bs`,`ar`,`br`} で同期し，並行合成することで得られたプロセス `SYS` として，状態マシン図全体の振る舞を表現できる．

```
SYS = SMD[|{as,bs,ar,br}|]MSG
```

シーケンス図は，ライフライン上のオカレンスの順序関係をプロセスとして記述し，並行合成することでプロセス表現できる．例として，シーケンス図 SQ1 は，ライフラインを表すプロセスの並行合成 `SQ1` として以下のように表現できる．

```
SQ1M1 = as->br->END
SQ1M2 = ar->bs->END
  SQ1 = SQ1M1 ||| SQ1M2
```

そして状態マシン図と同様に，前述のメッセージを表すプロセス `MSG` と合成することで得られたプロセス `SPEC1` としてシーケンス図の振る舞いを表現できる．

```
SPEC = SQ1[|{as,bs,ar,br}|]MSG
```

4. FDR による詳細化関係の検証

FDR では共通するアクションに関して，検証性質プロセスが検証対象プロセスの実行順序を含むか否かを検証できる．本手法では状態マシン図のプロセス `SYS` を検証対象プロセス，シーケンス図のプロセス `SPEC` を検証性質プロセスとして以下の文を実行することで整合性検証を実現する．

```
assert SPEC [F= SYS
```

M1,M2 と SQ1 に関して整合性検証を行った結果，違反は検出されなかった．一方で，SQ2 に関して同様に検証を行ったところ，以下の反例が出力された．

```
perform:bs
```

この反例は検証性質プロセス，すなわち SQ2 の $b!$ に違反があることを示している．前述の通り，本例題での整合性違反は SQ2 で最初に処理される $b!$ が状態マシン図側で実行できないことであるから，詳細化違反とその原因が正しく検出されている．

5. おわりに

本稿では，状態マシン図とシーケンス図を CSP で表現することで，整合性検証を行う手法を提案した．今後の課題として，変換処理の自動化および大規模かつ複雑な階層構造をもつ例題への適用実験を行う．

参考文献

1) Miyazaki, H., et al.: Synthesis and Refinement Check of Sequence Diagrams., *IEICE Trans. on Inf. and Syst.*, Vol.E95-D, No.9, pp.2193–2201 (2012).
2) 横川ほか：プロセスの模倣性判定に基づく UML 設計間の整合性検証，ウィンターワークショップ 2013・イン・那須予稿集 (2013).