Vol.2021-ARC-244 No.48
Vol.2021-SLDM-194 No.48
Vol.2021-EMB-56 No.48
2021/3/26

同時に起こる事象を考慮した区間振る舞いモデルの提案

張 漢明[1]　野呂 昌満[1]　沢田 篤史[1]

概要：高い信頼性が求められる組込みシステムでは，際どいタイミングで起こる事象を適切に処理する必要がある．際どいタイミングで起こる事象を「同時事象」として扱い，事象の同時性を区間に局所化した区間振る舞いモデルを提案する．事象の同時性を考慮した並列システムの振る舞いを簡易に記述するための構造を解明し定式化する．我々がこれまでに提示した区間振る舞いモデルに対して，一般化および洗練化したモデルの構造を再定義し，その形式的な意味をプロセス代数 CSP を用いて定義する．区間振る舞いモデルは，並列性の観点から分割統治の視点を与え，事象に関する抽象化写像の導入がモデル検査の実用化に寄与する．

1. はじめに

並列システムでは，同時に起こる事象処理の欠陥（fault）が重大な障害（failure）の原因となる場合がある．メッセージの送受信が並列に実行される環境において，ある順序で起こった事象が別のプロセスを経由して事象を伝達すれば，到達先のプロセスでその順序が保存されない場合がある．例えば，外部環境から事象 1 と事象 2 が，図 1 の UML のシーケンス図で記述された順序で，別のセンサーを経由して制御に伝達された場合，事象 1 と事象 2 の受信の順序が制御では逆転する．事象 1 の方が事象 2 より優先度が高い場合，制御が事象 2 を受信した時点で事象 2 の処理を行うと，優先度が低い事象が先に処理されてこれが障害の原因となる場合がある．このような状況は際どいタイミングで稀にしか起こらないので，障害の記録などから実行履歴をたどってこのような欠陥を発見することは難しい．大域的な時計がない分散環境では，あるプロセスで受信した事象の順序は，それらのきっかけとなった事象の順序と同じとは限らないので，順序の情報を捨象して同時に起こる事象（以降，同時事象という）の概念が必要である．

並列システムの振る舞いは，UML[1] のアクティビティ図や，CSP[2] や CCS[3] に代表されるプロセス代数などの計算モデルを用いて表現される．ただし，後述するように，これらの表記法や計算モデルはいずれも，仕様定義から設計に至る段階において，同時事象を見通しよく記述するという観点からは不十分である．アクティビティ図では，フォークとジョインを用いてアクティビティの並列性を表現することができるが，複数の事象が「同時」に動作することを直接表現する方法はない．

プロセス代数では，並行合成演算子の意味は一般的にインターリービング（interleaving）の概念を用いて与えられる．インターリービングによって意味づけされる並行プロセスは，非決定性を用いた遂次プロセスと等価なプロセスとして表現されるので，事象が同時に起こることを直接表現することはできない．並行プロセスの並列性を表現するために，インターリービングとは違うステップ意味論（step semantics）[4] がある．ステップ意味論では，アクションを 1 つの集合 ({x},{y},{x,y}) とみなすことにより，状態遷移モデルで振る舞いの並列性を表現できる．ステップ意味論を用いると同時事象を表現することができるが，どの範囲で同時とみなすかを表現することはできない．

組込みシステムは一般的に複数の物理装置で構成されるので，システムの並列性に起因する事象の同時性を考慮する必要がある．並列性を持つプログラムは，非決定的な実行の特性から，実行時におこる障害から欠陥を特定することが難しい．仮に特定できた場合であっても，一時的なプログラム修正がプログラム全体の見通しを悪くする．際ど

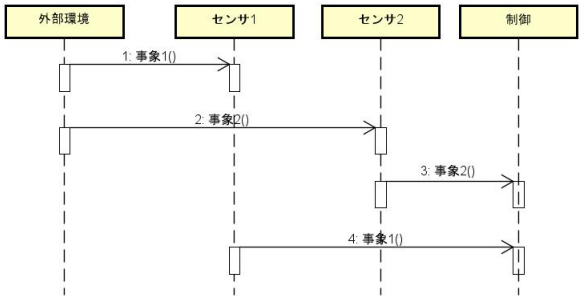

図 1　事象順序の逆転

[1] 南山大学
Nanzan University, Showa, Nagoya 466–8673, Japan

いタイミングでおこる事象の順序は，プロセス間の事象の通知で逆転する場合があるので，それらを同時事象として扱う表記法が必要である．並列システムの振る舞いに関する性質をプログラムのテストで検証することは困難なので，仕様・設計の段階で形式仕様言語を用いて記述・検証することが望ましい．仕様・設計段階の振る舞いを自動で検証する技術としてモデル検査がある．モデル検査を実用化するには，状態爆発の問題を解決するための抽象化技術が必要である．インターリービングで意味づけされた計算モデルは，遂次の概念で意味づけされ単純なので実用的なモデル検査ツールが多数存在する．同時事象を含んだ並列の振る舞いをインターリービングで意味づけすれば，既存のモデル検査ツールの利用が可能となる．これまでの議論により，本研究では以下の 3 つの研究課題を設定する．

研究課題

(1) 同時事象を含む振る舞いの簡易な表記法
(2) 事象の数に関する抽象化支援
(3) 既存の仕様言語との融合

本研究の目的は，並列システムにおいて同時に起こる事象を簡易に表現できる振る舞いの表記法を提示することである．我々は，同時事象を表現するためのモデルとして，区間振る舞いモデル [5] を提案してきた．区間振る舞いモデルの基本的なアイデアは，振る舞い記述に開始と終了の時間範囲を規定する区間の概念を導入することにより，事象の同時性を区間に局所化することである．提案した区間振る舞いモデルは，プロセス代数 CSP に依存した表記法なので，振る舞いを記述するためには CSP の知識が必要である．また，これまでのモデルには抽象化の概念を含んでいない．本研究では，既存の形式仕様言語に依存しない，区間振る舞いモデルの構造を抽出して定式化することを目指す．区間振る舞いモデルの構造を，UML などの既存の図式表現や CSP や CCS などの既存の形式表現に対応づけることにより，区間振る舞いモデルの実用性が高まることが期待できる．

本研究では，

- 区間の構造の一般化と洗練化，および，
- 簡易な構造

の観点から，区間振る舞いモデルを再定義する．これまでに提案した区間振る舞いモデル [5] から，区間を定義するために必要な概念を抽出し，それらの構造を数学モデルで表現する．これらの構造に対して，形式仕様言語としてプロセス代数 CSP，および図式表現として UML 記法と対応づける．CSP は，プロセス記述に関数プログラミングの機能を有する言語 CSP_M[6] が提供されており，FDR[7] などの実用的なモデル検査器が存在するので，振る舞いを表現するための形式言語として適切である．

本稿では，提案する区間振る舞いモデルの

- 区間の構造
- CSP による区間の意味
- UML のアクティビティ図による表現

を提示する．区間振る舞いモデルでは，ステップ意味論の記法を取り入れて，事象を集合を用いて表現する．区間は，区間の範囲を規定する開始境界と終了境界，入力事象集合と出力事象集合，および，入力事象集合と出力事象集合間の抽象化写像から構成される．区間は，基本区間を基にして，逐次区間，選択区間，および，並列区間の複合区間から構成される．自動販売機システムと現金自動預払機システムを事例として，区間振る舞いモデルを用いた記述例を示す．

区間振る舞いモデルは，

- 抽象度の高い検証可能な振る舞い表現を提供し，
- モデル検査の実用化

に寄与する．区間の構造は，同時に起こる判断を区間に限定し，区間の構成に関わる事象だけに事象を限定する．区間の振る舞いはブラックボックス化して，区間内で同時事象を含めた起こり得る入力事象と出力事象を定義する．入力事象と出力事象の関係を抽象化写像を用いて定義する．区間間の関係は，出力事象を用いて表現されるので，入力事象が未定義でも区間間の振る舞いを定義することができる．これが，モデル検査を実用化するための事象の数の削減に寄与する．入力事象と抽象化写像を用いて，詳細な振る舞いを表現することも可能である．

以降，第 2 節では区間振る舞いモデルの基本的なアイデアとその構造を定義する．第 3 節ではプロセス代数 CSP を用いた振る舞いの意味定義を示す．第 4 節では UML のアクティビティ図を用いて区間振る舞いモデルの記述例を示し，最後に提案する区間振る舞いモデルの妥当性について議論する．

2. 区間振る舞いモデル

本節では，区間振る舞いモデルの基本的なアイデアを説明し，区間振る舞いモデルの構造を定義する．

2.1 基本的なアイデア

区間振る舞いモデルの基本的なアイデアは，

(1) 同時事象の局所化と単一事象化，
(2) 事象の抽象化，および，
(3) 区間の形式化

である．

2.1.1 同時事象の局所化と単一事象化

区間振る舞いモデルでは，区間を定義するために

- 区間の境界を導入し，
- 並列で起こる事象を単一事象で表現

する．単一事象化は，並列実行環境において区間内で

- 事象が起こる順序を捨象して，
- 同時事象を「事象の集合」で表現

Vol.2021-ARC-244 No.48
Vol.2021-SLDM-194 No.48
Vol.2021-EMB-56 No.48
2021/3/26

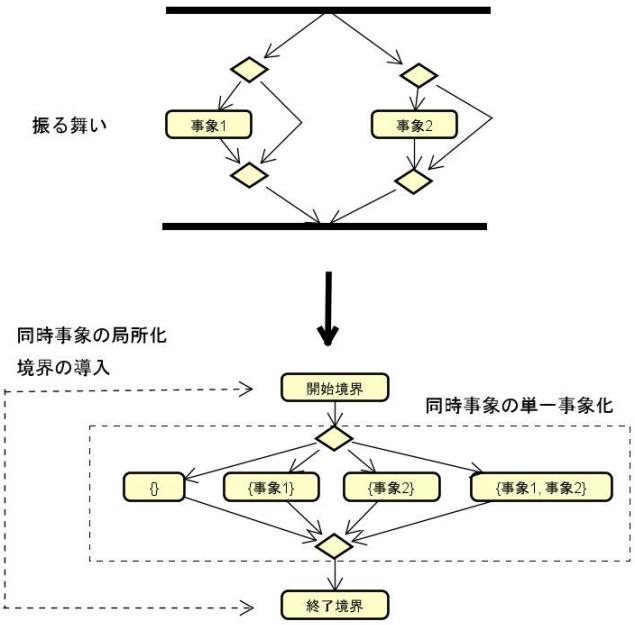

図 2 同時事象の局所化と単一事象化

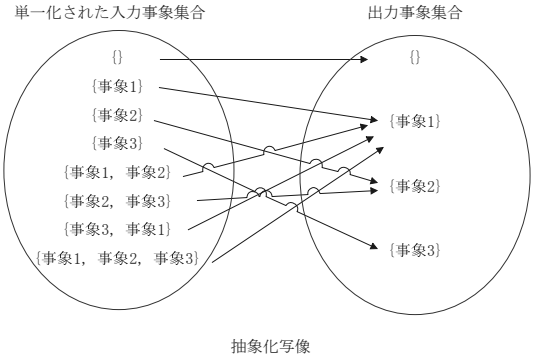

図 3 区間内事象の抽象化の例

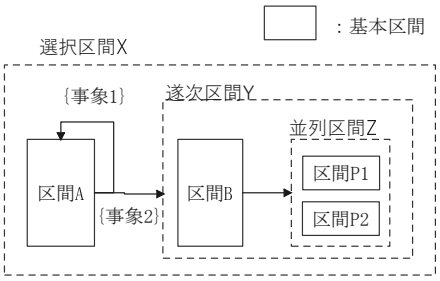

図 4 区間間の構造

する．事象の集合の表現は，ステップ意味論 [4] の表現を採用した．

図 2 は，UML のアクティビティ図を用いて，同時事象の局所化と単一事象化の具体例を表している．図中の下向きの矢印（↓）の上の部分は，振る舞いの表現として，事象 1 が起こるか起こらないか，並行して，事象 2 が起こるか起こらないかを表している．区間振る舞いモデルでは，この振る舞いを矢印の下の部分のように表現する．上の部分の並行な振る舞いを，何も起こらない（{}），事象 1 が起こる（{ 事象 1}），事象 2 が起こる（{ 事象 2}），事象 1 と事象 2 が同時に起こる（{ 事象 1, 事象 2}），のどれかが起こると表現する．

これは，開始境界と終了境界の間で起こり得る事象だけに着目して，これらの事象が境界間で同時に起こり得る事象を列挙することにより，並列性を排除した表現である．区間内の振る舞いの詳細および事象が起こる順序を捨象している．更に，事象が起こる回数も捨象している．事象が起こる回数を考慮するには多重集合（multiset）を用いれば良いが，事象を簡易に扱うために回数は考慮しないことにする．

2.1.2 事象の抽象化

区間において考慮すべき事象の数を削減するために，同時事象を含んだ起こり得る全ての事象に対する抽象化の対応関係を，抽象化写像として導入する．区間では，入力事象集合と出力事象集合を構成要素とする．入力事象集合は，起こり得る事象の集合である．図 2 における入力事象集合は { 事象 1, 事象 2} である．抽象化写像は，単一事象化された入力事象集合から出力事象集合への写像である．単一事象化された入力事象集合は，入力事象集合の全部分集合，つまり，べき集合である．

図 3 は，抽象化写像の具体例を示している．単一化された入力事象集合は，入力事象集合 { 事象 1, 事象 2, 事象 3} のべき集合である．左から右への矢印 (→) は，それぞれ対応する出力事象を表している．抽象化写像の定義域の要素に対して対応が全て定義されていれば，この区間での抽象化は全て網羅されていることになる．抽象化写像の終集合（値域）は任意である．抽象化写像は，区間内で事象の名前をグループ化したり変更したりすることを可能とする．区間の出力事象集合は，抽象化写像の定義がなくても，定義が可能である．対象の区間で期待する事象の集合を出力事象集合で事前に検討することができる．抽象化写像は全射である必要はない．

2.1.3 区間による振る舞い表現

対象とする並列システムの振る舞いを表現するために必要な構造として，

- 基本区間と
- 複数の区間から構成される複合区間

を導入する．基本区間は，これ以上分割することができない原子的な区間である．複合区間は，複数の区間から構成される区間で，「逐次区間」，「選択区間」，「並列区間」の 3 種類の区間がある．逐次区間と選択区間は，区間間の順序を規定するために必要な構造である．区間の繰り返しは，逐次区間と選択区間において，次の区間に自分自身を指定することにより再帰を用いて表現する．並列区間は，並列に実行する区間のグループ化を可能にする．並列実行区間のグループ化は，並列区間の全体で起こり得る事象の組み合わせを削減する．

Vol.2021-ARC-244 No.48
Vol.2021-SLDM-194 No.48
Vol.2021-EMB-56 No.48
2021/3/26

図 4 は，区間間の構造の具体例である．逐次区間 Y は，区間 B と並列区間から構成されている．逐次区間 Y では，区間 B が終了した後で，並列区間 Z が実行される．選択区間 X は，区間 A で事象 1 が起こった後で区間 A が再帰的に選択され，区間 A で事象 2 が起こった後は逐次区間 Y が選択される．並列区間 Z では，区間 P1 と区間 P2 が並列に実行される．

2.2 区間振る舞いモデルの定義

上記で示された，区間振る舞いモデルの構造を定義する．

2.2.1 事象

区間振る舞いモデルでは，操作などの実行時間を捨象したものを事象という．事象には，これ以上分割できない「原子事象」と，同時に起こる事象を表す「事象の集合」から構成される．

定義．原子事象の全集合を BE とする．原子事象を $be \in BE$ として表す．

定義．原子事象の集合 $bes \subset BE$ は事象である．事象の全集合 $E = Power(BE)$ とする．ただし，$Power(X)$ は集合 X のべき集合である．

2.2.2 区間

区間は基本区間と，区間から構成される複合区間がある．複合区間は，逐次区間，選択区間，並列区間から構成される．

定義．基本区間の全集合を BI，逐次区間の全集合を SI，選択区間の全集合を CI，並列区間の全集合を PI とする．区間の全集合を $I = BI \cup SI \cup CI \cup PI$ とする．ただし，BI, SI, CI, PI はそれぞれ互いに集合の交わりは空集合である．また，基本区間を $bi \in BI$，逐次区間を $si \in SI$，選択区間を $ci \in CI$ ，並列区間を $pi \in PI$ ，区間を $i \in I$ で表す．

区間の共通構成要素

区間の共通構成要素を定義する．

(1) 開始境界と終了境界

定義．境界の全集合を B とする．区間 i の開始境界を $SB_i \in B$，終了境界を $EB_i \in B$ と表す．

(2) 入力事象集合と出力事象集合

定義．入力事象集合を $In_i \subset E$, 出力事象集合を $Out_i \in E$ と表す．

(3) 抽象化写像

定義．抽象化写像を $A_i \in E \to E$ とする．ただし，$X \to Y$ は集合 X から集合 Y への全関数であり，$A_i(In_i) \subset Out_i$ とする．

基本区間

定義．基本区間は 5 項組 $(SB_{bi}, EB_{bi}, In_{bi}, Out_{bi}, A_{bi})$ の共通構成要素で構成される．

逐次区間

定義．逐次区間は共通構成要素に「始めの区間」と「次の区間」を加えた 7 項組 $(SB_{si}, EB_{si}, In_{si}, Out_{si}, A_{si}, Fst_{si}, Snd_{si})$ から構成される．ただし，始めの区間を $Fst_{si} \in I$，次の区間を $Snd_{si} \in I$ とする．また，$In_{si} = In_{Fst_{si}}$，$Out_{si} = Out_{Snd_{si}}$ である．

選択区間

定義．選択区間は共通構成要素に「選択元区間」と「選択写像」を加えた 7 項組 $(SB_{ci}, EB_{ci}, In_{ci}, Out_{ci}, A_{ci}, Src_{ci}, Cho_{ci})$ から構成される．ただし，選択元区間を $Src_{ci} \in I$，選択写像を $Cho_{ci} \in Out_{Src_{ci}} \to I$ とする．また，$In_{ci} = In_{Src_{ci}}$，$Out_{ci} = \bigcup_{i \in ran(Cho_{ci})} Out_i$ である．ただし，$ran(M)$ は写像 M の終集合（値域）とする．選択写像は，選択元区間の出力事象から選択される区間への写像である．

2.2.3 並列区間

定義．並列区間は共通構成要素に「並列合成区間集合」を加えた 6 項組 $(SB_{pi}, EB_{pi}, In_{pi}, Out_{pi}, A_{pi}, Par_{pi})$ から構成される．

並列合成区間集合は $par_{pi} \subset I$ である．ただし，$\forall i \in par_{pi}(S_i = S_{pi} \wedge E_i = E_{pi})$ である．

また，A_i に関して，$dom(A_i) = set(\Pi_{\{pi \in par_i\}} Out_{pi})$ である．ただし，$\Pi_{\{i \in \lambda\}} X_i$ は λ を添付集合とする X_i の直積，$set(X)$ は集合 X の要素（組）から集合への変換，$dom(M)$ は写像 M の定義域とする．

3. プロセス代数 CSP による意味定義

本節では，第 2.2 節で示した区間振る舞いモデルの構成要素の意味を CSP のテキスト表現である CSP_M を用いて定義する．

事象

事象 E の定義を以下に示す．

```
-- 基本事象
datatype BE = be0 | be1
-- 事象
channel ev: power(<be0, be1>)
nametype E = {| ev |}
-- べき集合
power(x) = {set(y) | y <- power0(x)}
power0(<>) = {<>}
power0(<h>^tl) =
  union({<h>^x | x <- power0(tl)}, power0(tl))
```

基本事象 BE は列挙型を用いて定義する．事象は CSP のイベントとして定義する必要がある．事象 E は基本事象の集合として表現する．CSP のイベント名に集合を用いることができないので，`ev` という接頭辞をつけて事象をデータとして扱う．事象は集合を用いて `ev.{be0,be1}` のように記述する．

区間

区間 I の定義を以下に示す．

```
-- 区間
datatype I = i0 | i1 | i2
```

区間 I はその名前を列挙型を用いて定義する．

境界

開始境界 SB と終了境界 EB の定義を以下に示す．

```
-- 開始境界、終了境界
channel s, e : I
nametype B = {| s, e |}
SB :: (I) -> Event
SB(i0) = s.i0
EB :: (I) -> Event
EB(i0) = e.i0
```

境界 B は CSP のイベントとして定義する．開始境界には接頭辞 s，終了境界には接頭辞 e をつけてそのデータ部を区間 I とする．SB(i0) = s.i0 は，区間 i0 の開始境界を関数のパターン定義を用いて記述する．区間 i1 の定義は，区間 i0 に続いて SB(i1) = s.i1 のように定義すればよい．SB :: (I) -> Event は関数の型を示している．

入力写像集合と出力写像集合

入力事象集合 In と出力事象集合 Out の定義を以下に示す．

```
-- 入力事象集合
In :: (I) -> {{BE}}
In(i0) = {{be0}, {be1}, {be0,be1}}
-- 出力事象集合
Out :: (I) -> {{BE}}
Out(i0) = {{be0}, {be1}}
```

In と Out は，関数のパターン定義を用いて記述する．

抽象化写像

抽象化写像 A の定義を以下に示す．

```
-- 抽象化写像
A :: (I) -> ({BE}) -> {BE}
A(i0) = A_i0
A_i0 :: ({BE}) -> {BE}
A_i0(x) = if x == {be0,be1} then {be0} else {be1}
```

A は，事象から事象への関数（Abst_i0）として定義する．

基本区間

基本区間 BI の振る舞い定義を以下に示す．

```
-- 基本区間
BI :: (Event, Event,
         {{BE}}, ({BE}) -> {BE}, {BE}) -> Proc
BI(sb, eb, in, abst, t) =
    sb -> BI_0(eb, in, abst, {}, t)
BI_0 :: (Event, {{BE}},
           ({BE}) -> {BE}, {BE}, {BE}) -> Proc
BI_0(eb, in, abst, occurred, t) =
    ([] x: in @ ev.x ->
      BI_0(eb, in, abst, union(occurred, x), t))
   []
      ev.t -> ev.abst(occurred) -> eb -> SKIP
```

BI の引数は，開始境界 sb，終了境界 eb，入力事象集合 in，抽象化写像 abst，終了トリガー t である．出力事象集合は抽象化写像の値域となるので，出力事象集合は構成要素から削除している．終了トリガーは，振る舞いを定義するために新たに必要な事象である．

終了トリガーの役割は，終了境界の直前に出力事象を決めるために必要となる．BI_0 では，入力事象集合の振る舞いに対して，起こった事象を集積する役割を果たす．起こった事象は occurred で保持する．出力事象は終了境界の直前に出力するので，区間内の終わりの印として終了境界とは別に事象を指定する．基本区間の振る舞いは，入力事象集合の振る舞いを X とすると，

```
X [| in |] BI(sb, eb, in, abst, t)
```

で表現される．

基本区間の振る舞い X を考慮しない場合の振る舞いは，以下の BI_A になる．

```
BI_A :: (Event, Event, {{BE}}) -> Proc
BI_A(sb, eb, out) =
    sb ->
        ([] x:out @ ev.x -> SKIP);
    eb -> SKIP
```

BI_A は出力事象集合のどれかの事象が選択されることを表している．

逐次区間

遂次区間 SI の振る舞い定義を以下に示す．

```
-- 逐次区間
SI :: (Event, Event, I, I) -> Proc
SI(sb, eb, fst, snd) =
    sb -> (Be(fst); Be(snd)); eb -> SKIP
```

SI は，始めの区間（fst）の後で，次の区間（snd）が実行されることを表している．関数（Be）は区間の振る舞いを表している．ここでは，入力事象集合，出力事象集合，および，抽象化写像は省略している．

区間の振る舞いは，

```
Be :: (I) -> Proc
Be(i0) = BI(SB(i0), EB(i0),
                In(i0), A(i0), T(i0))
```

のように定義される．

選択区間

選択区間 CI の振る舞い定義を以下に示す．

```
-- 選択区間
CI :: (Event, Event, I, ({BE}) -> I) -> Proc
CI(sb, eb, src, cho) =
  sb ->
    ([] x:Out(src) @ ev.x -> Be(cho(x)));
  eb -> SKIP
```

Vol.2021-ARC-244 No.48
Vol.2021-SLDM-194 No.48
Vol.2021-EMB-56 No.48
2021/3/26

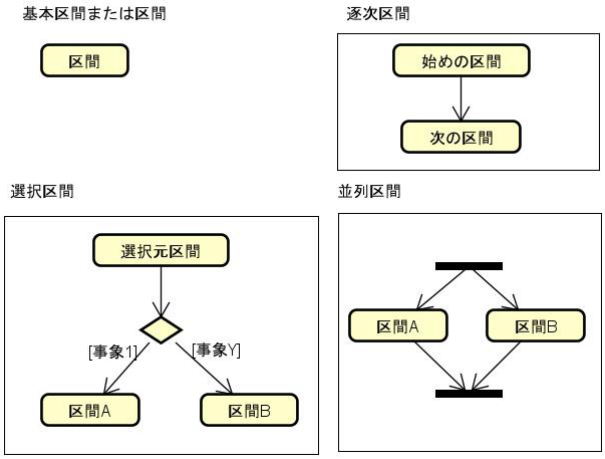

図 5 アクティビティ図による区間の表現

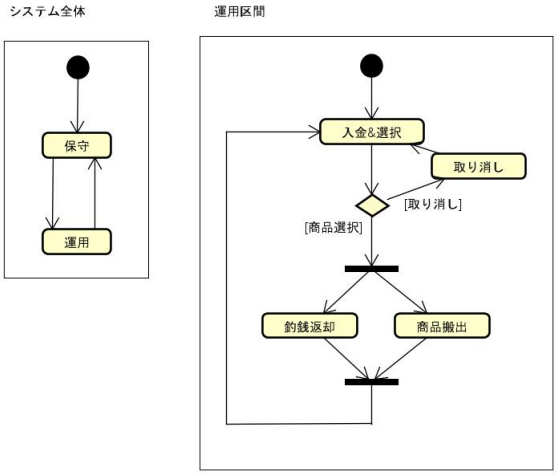

図 6 自動販売機システム

CI は, 選択元区間(src)の後で, 選択された区間(cho(x))が実行されることを表している．ここでは，入力事象集合，出力事象集合，および，抽象化写像は省略している．

並列区間

並列区間 PI の振る舞い定義を以下に示す．

```
PI :: (Event, Event, {I},
          ({BE}) -> {BE}, {BE}) -> Proc
PI(sb, eb, abst, par, t) =
  ([|{sb, eb}|] x:par @
          Be(x)[[SB(x) <- sb, EB(x) <- eb]])
    [| InIsE(par) |]
    BI(sb, eb, InIs(par), abst, t)
InIs :: ({I}) -> {{BE}}
InIs(is) = Union({In(i) | i <- is})
InIsE :: ({I}) -> {Event}
InIsE(is) = {ev.x | x <- InIs(is)}
```

PI は，並列合成区間集合（par）の各区間が開始境界と終了境界と同期をとって並列合成され，それを入力事象とした基本区間の振る舞いとなることを表している．ここでは，入力事象集合と出力事象集合は省略している．また，終了トリガーが追加されている．

4. 事例

本節では，

- 自動販売機システムと
- 現金自動預払機システム

を事例として，区間振る舞いモデルによる記述例を示す．

4.1 アクティビティ図による区間の表現

対象システムを区間振る舞いモデルを用いて図式表現するために，区間の構成要素をアクティビティ図の構成要素に対応づける．図 5 にアクティビティ図を用いた区間の表現を示す．

基本区間と区間　基本区間と区間はアクションノードに対応づける．

逐次区間　始めの区間と次の区間をフローでつなぐ．

選択区間　分岐を使って選択元区間の出力事象と選択される区間を対応づける．

並列区間　フォークとジョインを用いて並列合成区間を表す．

4.2 自動販売機システム

自動販売機システムの区間振る舞いモデルを用いた記述例を図 6 に示す．図の左側が自動販売機システム全体の区間の構造を表し，図の右側が運用区間の構造を表している．

システム全体

自動販売機システムは「保守」区間と「運用」区間から構成されている．保守から始まり，保守が終了すると運用に変わる．運用が終了すると保守に変わる．この順序で区間が繰り返し実行される．保守と運用は互いに逐次区間として合成されているので，区間間に関わる事象は存在しない．保守と運用がどのように終了するかは保守区間と運用区間で規定される．

運用区間

運用区間は「入金＆選択」区間,「釣銭返却」区間,「商品搬出」区間,「取り消し」区間から構成されている．入金＆選択では，入金は区間内で繰り返し行われることを想定している．入金＆選択が終了するのは，商品選択事象が起こるか，取り消し事象が起こる場合である．商品が選択された後は，釣銭返却と商品搬出が並列して行われ，入金＆選択に戻る．入金＆選択中で取り消しがあれば，取り消しの処理を行い入金＆選択に戻る．

本事例において，区間振る舞いモデルを用いて明示されることを以下に示す．

- 取り消しは「入金＆選択」中に起こる可能性がある．
- 選択事象と取り消し事象の同時性については,「入金＆選択」で責任を持つ．

Vol.2021-ARC-244 No.48
Vol.2021-SLDM-194 No.48
Vol.2021-EMB-56 No.48
2021/3/26

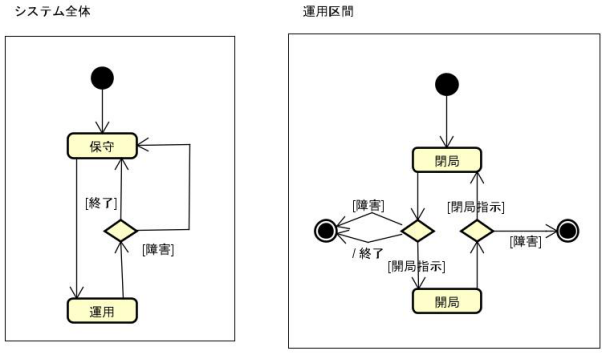

図 7 現金自動預払機システム

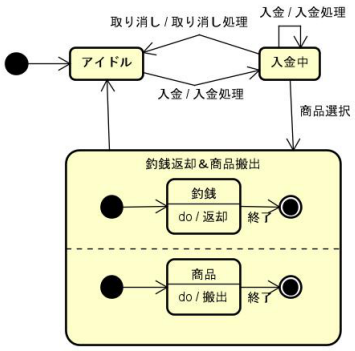

図 9 自動販売機（運用）の状態マシン図

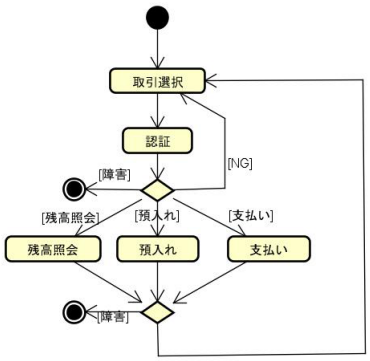

図 8 現金自動預払機（開局）

- 「釣銭返却」と「商品搬出」を並列に処理する．

4.3 現金自動預払機システム

現金自動預払機システムの区間振る舞いモデルを用いた記述例を 7 に示す．自動販売機システムと同様に，図の左側が現金自動預払機システム全体の区間の構造を表し，図の右側が運用区間の構造を表している．

システム全体と運用区間

現金自動預払機システムは，自動販売機システムと同様に，「保守」区間と「運用」区間から構成されている．自動販売機システムとの違いは，運用から障害事象が起こることを想定している．運用区間は，「開局」区間と「閉局」区間から構成されている．運用では，閉局においても開局においても障害事象が起こる可能性がある．この事例では，下位で起こった障害事象が上位の区間に伝達されることを表している．

開局区間

図 8 に開局区間の構造を示す．開局では，利用者の「認証」と残高照会や支払いなどの取引中に障害が発生することを想定している．この構造から，認証の障害と取引の障害が区別されていることがわかる．

本事例において，区間振る舞いモデルを用いて明示されることを以下に示す．

- 障害が発生する区分が明確になっている．
- 障害の処理をどこで行うかが明示される．システム全体，運用区間，開局区間では障害の処理は行われずに通知を行うだけである．障害の処理は発生した区間内で責任を負う．

5. 考察

区間振る舞いモデルに基づいた振る舞い記述の妥当性について，「UML の振る舞いモデル」と「境界導入の意義」の観点から考察する．

5.1 UML の振る舞いモデルとの比較

UML の振る舞いモデルである状態マシン図とアクティビティ図に対して，区間振る舞いモデルの表現と比較する．

状態マシン図

図 9 は，第 4.2 節で用いた自動販売機の運用区間の振る舞いを，状態マシン図を用いて記述したものである．状態マシン図による記述では状態と遷移が基本的な考え方なので，図 9 のように「アイドル」状態と「入金中」状態をわけることが考えられる．状態マシン図において計算の実行は，遷移に付随するアクションと，状態に不随するアクションで表現される．並列性は「釣銭返却＆商品搬出」状態で表されている並列状態で表現される．

状態マシン図でモデル化された表現では，同時事象の記述は「遷移」を追加するか「アクション」内で対応することになる．遷移を追加すると元の状態マシン図と構造が変わるので，状態マシン図間の整合性を考慮する必要がある．アクションで同時事象の対応を行うと，事象がアクション内に埋もれ散在することになるので，全体の見通しが悪くなる．

アクティビティ図

アクティビティ図では，区間の概念をアクティビティ・アクションに対応づけ，並列性をフォークとジョインに対応づけると，区間振る舞いモデルの構造を自然に表現することができる．分岐のガードは，直前のアクションで出力事象が起こった事象を表す．図 6 の例では，「入金＆選択」区間で「商品選択」もしくは「取り消し」が起こったことを表現している．アクティビティ図は，区間振る舞いモデ

ルに対して，状態マシン図より親和性が高い．アクティビティ図から区間振る舞いモデルの構造を抽出することにより，同時性を考慮した表現を得ることが考えられる．

区間振る舞いモデル

区間振る舞いモデルの表現では，同時事象は区間内に限定されることを規定している．図 6 の記述では，分岐が区間が変わる重要な事象として明示されている．「入金＆選択」区間からの分岐では，「商品選択」事象と「取り消し」事象が区間の選択を決定している．同時事象に関しては，「入金＆選択」区間が責任を負う．同時事象の詳細は「入金＆選択」区間で定義されるが，全体の構造を変更する必要はない．区間振る舞いモデルを用いた振る舞い記述では，仕様・設計の最初の段階から同時事象の扱いを考慮した構造を決定する．区間振る舞いモデルは，同時事象の詳細を未定義のままで構造の妥当性の検討を可能にし，その構造が同時事象に関する後工程の仕様となる．

5.2 境界導入の意義

境界は同時事象を判定する責任の範囲を明確にする．「開始境界」と「終了境界」のうち，終了境界が同時事象の扱いで重要な役割を果たす．図 1 の事象順序の逆転の例では，制御で事象 2 を受信したさいに他の事象を受信していないか調べる必要がある．このとき，同時事象を厳密に扱うためには，事象の通知や他の処理を止めることが必要な場合がある．障害が発生したときに，何をもって同時とするかが仕様としての契約の上で重要になる．

区間振る舞いモデルを用いた対象システムの振る舞い定義は，同時事象を考慮する境界を明確にする．図 6 の自動販売機の例では，商品を選択するときがそのような境界になる．図 7，図 8 の現金自動預払機の例では，障害事象が境界の区分として重要となる．区間振る舞いモデルでは，どのような障害がどの区間で考慮すべきかを，障害の詳細を未定義のままで検討することができる．

6. おわりに

区間振る舞いモデルを用いたシステム記述の特徴は，「事象の同時性に着目した区間の概念を用いた振る舞い記述のモジュール化」である．区間は，

- 想定している入力事象の集合と出力事象の集合，
- 入力事象と出力事象の具象抽象関係（抽象化写像）

から構成される．対象システムの振る舞いは，基本区間を基にして，逐次区間，選択区間，および，並行区間の複合区間を用いて簡易に表現される．区間は，同時事象を考慮する責任の範囲を明確にして，全体の区間の構造を維持したうえで，詳細を後工程で決定することを可能にする．

本稿では，区間振る舞いモデルの意味をプロセス代数 CSP を用いて定義した．区間振る舞いモデルの構造を既存の形式言語で対応づけることにより，振る舞いの性質を仕様・設計の段階で形式的に検証することを可能にする．CSP では FDR などのモデル検査器が利用可能で，入力事象集合と出力事象集合を対応づける抽象化写像が，事象の数の削減につながりモデル検査の実用化に寄与する．

今後の課題として，

- CSP を用いた形式検証の事例の提示
- 並列性を備えたプロセス代数との関係

があげられる．本稿では，形式仕様を用いた検証の事例を示していない．仕様・設計の段階で振る舞いの性質を検証することは重要である．今後，自動販売機システムと現金自動預払機システムの事例の詳細化を行い，区間振る舞いモデルの実用性を検討する．並列性を扱うプロセス代数として，事象を集合で扱う step 意味論 [4]，場所の概念を用いた分散システムを対象としたプロセス代数 [8][9]，通信路を扱うことができる π–Calculus[10] などがある．これらの計算モデルと区間振る舞いモデルとの関係を検討する．

謝辞 本研究の一部は，JSPS 科研費 19K11911，20K11759，2020 年度南山大学パッヘ研究奨励金 I-A の助成を受けて実施した．

参考文献

[1] Object Mnagement Group: Unified Modeling Language, https://www.omg.org/spec/UML.
[2] Hoare, C.A.R. : Communicating Sequential Processes, Prentice Hall International Series in Computer Science, 1985.
[3] Milner, R. : Communication and Concurrency, Prentice Hall International Series in Computer Science, 1989.
[4] Milne, G.J. : CIRCAL and the Representation of Communication, Concurrency, and Time, ACM Trans. Program. Lnag. Syst., Vol. 7, No. 2, pp. 270–298, 1985.
[5] 張漢明, 野呂昌満, 沢田篤史: 複数事象の発生を含意した区間振る舞い記述法とその検証法の提案. コンピュータソフトウェア, Vol. 34, No. 2, pp. 3–15, 2017.
[6] CSP_M : https://cocotec.io/fdr/manual/cspm.html.
[7] $FDR4$: https://cocotec.io/fdr/.
[8] Castellani, I. and Hennessy, M.: Distributed Bisimulations, Journal of the ACM, Vol. 36, No. 4, pp. 887–911, 1989.
[9] Boudol, G., Castellani, I., Hennessy, M. and Kiehn, A. : Observing localities, Theoretical Computer Science, Vol. 114, No. v, pp. 31–61, 1993.
[10] Milner, R. : Communicating and Mobile Systems: the *pi*–Calculus, Cambridge University Press, 1999.