CSP を用いた安全性検証支援に関する研究
ーー交通システムを事例としてーー

2018SE100 渡邉　要
指導教員：張漢明

1　はじめに

交通システムは，人間の生命に関わる高い安全性が求められ, 交通システム制御の安全性検証は重要な技術である. 交通システムは，複数のシステムが連携した並行システムとして捉えられる．CSP は並行システムの記述・検証のための理論である．CSP を用いて安全性を検証するためには，CSP を用いた安全性検証の考え方を習得する必要がある.

本研究の目的は，CSP を用いた並行システムの安全性検証を支援することである．CSP を用いた安全性検証の概念を整理し，安全性検証のためのガイドラインを提示することを目指す．簡易な交通システムを事例として，提案するガイドラインの有用性を議論する.

2　関連技術

2.1　CSP の概要

CSP[1] (Communicating Sequential Processes) の略のことで, 並行システムを形式的に記述し，検証するための理論である．並行プロセスは，通信によるプロセス間の相互作用として表現される．C. A. R. Hoare が提唱した, CSP_M[2] の記法を用いた並行プロセスの記述例を以下に示す.

並行プロセス

並行プロセスとは同時並行して実行されているプロセスの集合である.

```
VMCT = coin -> (choc -> VMCT [] toffee -> VMCT)
GRCUST = (coin -> toffee -> GRCUST [
          coin -> choc -> GRCUST)
X1 = GRCUST [ {coin, choc, toffee}
   || {coin, choc, toffee} ] VMCT
```

自動販売機 VMCT はコインを入れ, チョコかコーヒーを買うことが出来る. 顧客 GRCUST はコインを入れてチョコ, コーヒーを買うことが出来る．P ｜｜ Q は P と Q が同じアルファベットを持つプロセスのときの記述方法である.

2.2　既存研究

CSP を用いた交通システムの検証に関する研究として木下らの研究 [3] がある．この研究では，自動運転システムとそれを運転するドライバの状態に着目し, それぞれのプロセスが予期せず停止することがないことを CSP を用いて安全性を検証している．UML を用いてシステムの記述を行い，仕様を LTL 式で表現する．LTL 式とは線形時相論理式（Linear Temporal　Logic）のことでシステムのふるまい・動作について, 真か偽かを判定することができる論理式である．この研究でのモデル検査の結論として, 自動運転システムが運転を続けるのが難しく, 且つドライバが何らかの理由で支障をきたしている場合にドライバの手動運転が続くパターンがみられた.

3　CSP を用いた安全性検証

3.1　提案する安全性検証の指針

CSP を用いた安全性検証の概念を整理し,CSP 記述に即した書き方をするためにガイドラインを示した. CSP 記述の流れを例を用いて第 2 章の既存研究で示したが, より理解しやすいようアクティビティ図で説明する. アクティビティ図とは UML の一種でシステムの実行の流れとそれに伴う条件分岐の流れを図示したものである. 以下に示す.

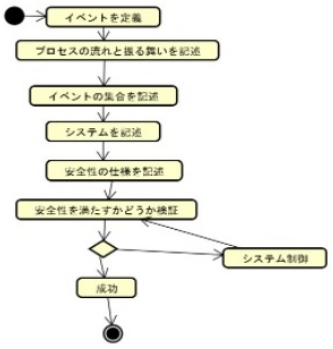

図 1　CSP 記述の流れを表すアクティビティ図

初めに, プロセスの振る舞いを表現するためにイベントを定義する. そして, その動作を定義するためのプロセスを与える. 安全性を検証するため, 危険性を含むプロセスも定義し, それを回避する記述を後に示すような手順で踏む. 次にイベントの集合を定義する. そして, 危険性を含んだシステムを記述し, 制御する前では危険性があることを確認し, 安全性の仕様を記述する.SYSTEM で危険性を確認した上で, その危険性を回避するためのシステム制御を行う. 最後に危険な状態が起こらないことを確認し, 安全性が保障されたことを確認する.

3.2　検証例

検証例として, 交差点における信号システムの安全性を検証する．この例の特徴は一方通行の交差点であり, 交差点による歩行者の通行は考えないものとする. また, 信号機の色は赤と青のみで考える. この例における安全性の仕様は交差する自動車同士が同時に交差点に入ることはないというものである. この CSP 記述の流れと実際の CSP 記述の一部を交えて説明する.

1. 初めに, プロセスの振る舞いを表現するためにイベントを定義する. この記述では datatype で型を定義し,channel で型を宣言している.

```
channel car:{1,2}.D
datatype D = approach | enter | leave
```

車線が違う車を考慮してそれぞれ car.1,car.2 と名前を区別して定義をする.

2. そして, 信号と車の動作を定義するための式を与える. 車の動作は交差点に侵入する, 交差点内に入る, 交差点から出るの 3 つの動作を繰り返すプロセスを定義する. 信号の動作は信号が青になる, 信号が赤になるの 2 つの動作を繰り返すプロセスを定義する. ここでは車の動作を記述する.

```
CAR1 = car.1.approach -> car.1.enter ->
          car.1.leave -> CAR1
```

3. 安全性を検証するため, 危険性を含むプロセスを定義し, それを回避する記述を後に示すような手順で踏む.
4. 各プロセスの振る舞いとアルファベットの集合を定義する. その一部を記載する.

```
LIGHT1 = light.1.red -> light.1.blue ->
         LIGHT1 [] car.1.enter -> LIGHT1
  E_C1 = {|car.1|}
```

5. 危険性を含んだシステムを記述し, 制御する前では危険性があることを確認する.SYSTEM では信号の制御が出来ていないので, 結果危険性である接触が起こってしまう.
6. 安全性の仕様を記述する. これは危険な状態が起こるまで繰り返し振る舞いが行われることを指している.

```
RUN(X) = [] x: X @ x -> RUN(X)
SPEC = RUN(diff(Events, {crash}))
```

7. SYSTEM で危険性を確認した上で, その危険性を回避するためのシステム制御を行う. 制御を交差点に入る, 出ていくという動作ができるのはその車線に対応した信号が青である場合である, というシステムにして, その逆の車線では信号が赤になっているというシステムにする.

```
CONTROL1 = (car.1.approach -> light.2.red
-> car.1.enter -> car.1.leave ->
light.2.blue -> CONTROL1)[] (car.2.approach
-> car.2.enter -> car.2.leave -> CONTROL1)
CONTROL2 = (car.2.approach -> light.1.red
-> car.2.enter -> car.2.leave ->
light.1.blue -> CONTROL2) [] (car.1.approach
-> car.1.enter -> car.1.leave -> CONTROL2)
```

8. 車と車の接触である crash が起こらないことを確認し, 安全性が保障されたことを確認する. SYSTEM は SPEC の詳細化であるかどうかを assert で確かめて SYSTEM は SPEC の詳細化ではなかった. よって安全性を満たしてないので,SAFE_SYSTEM を記述し,assert の結果詳細化関係がみられ安全性が保障された.

```
assert SPEC [T= SYSTEM
assert SPEC [T= SAFE_SYSTEM
```

4 考察

4.1 ガイドラインの有用性

以下の 3 つの有用性があると考えた.

1. 検証に必要な概念を整理することが可能になる
2. 危険な状態をイベントで表現する方法が分かる
3. 危険が起こらない振る舞いは関数で表記する

4.2 既存研究との比較

CSP モデルでは, イベントに基づいて対象の振る舞いを記述する. これを既存研究である自動運転システムと運転するドライバの状態に着目し, それぞれのプロセスが危険性を起こさないために必要な CSP 記述では, 安全性検証を CSP 記述ではなく LTL 式で表している. 先程より, 仕様と対象はともに CSP 記述となり, 新たに LTL 式を導入する必要はないことが言える. よって検証に CSP を自然に利用するためには, 状態遷移モデルをイベントに基づいた記述にする必要がある. そしてイベントベースの CSP 記述で安全性検証を行った場合, 状態ベースからの変換する必要がない, という手間が省けるという点で嬉しくなると考えた. 既存研究で安全性検証が上手くいかなかったのは状態ベースで書かれていたからである.

5 おわりに

本稿では, CSP を用いて安全性を検証するための概念を整理してガイドラインとして提示した. 今後の課題は, 信号交差点においての信号制御の CSP 記述を作成するにあたり, 歩行者の振る舞いも考慮した上で, その CSP 記述作成を行い, ガイドラインを提案しようと考えている.

参考文献

[1] C. A. R. Hoare: Communicating Sequential Processes, Prentice-Hall, 1985.

[2] FDR4: https://cocotec.io/fdr/.

[3] 木下聡子, 西村秀和, ユンソンギル, 北村憲康：自動運転車を取り巻く System of Systems の安全性要求の妥当性確認と検証. SEC journal, Vol.12, No.4, 2017.