自動運転システムを対象としたシナリオ開発のためのモデリング言語

青木 利晃[1,a)]　冨田 尭[1,b)]　河井 達治[1,c)]　川上 大介[2,d)]　千田 伸男[2,e)]

概要：自動運転システムの検証は，シナリオに基づいて行われることが一般的である．シナリオでは，自動車の位置関係や動作を表現するために，しばしば，アイコンなどを用いた図が用いられる．しかしながら，そのような図の解釈は曖昧であるため，見るものによる理解の齟齬を生じやすく，高信頼システムの開発には適していない．そこで，本論文では，統一された記法の提案を目指して，車両位置関係図 (Vehicle Position Relation Diagram, VPRD) を提案する．VPRD は，大量のシナリオを簡潔に表現でき，シナリオの分析設計に適している．また，VPRD に基づいたモデルを命題論理式に変換し，SAT ソルバを用いてすべてのシナリオを列挙する手法も提案する．シナリオを列挙するプロトタイプツールを実装し，典型的な挙動について実験を実施した．その結果，大量のシナリオを簡潔に記述できること，および，シナリオを列挙することの有効性を示すことができた．

Modeling Language for Scenario Development of Autonoumous Driving Systems

Abstract: We usually use scenarios in order to verify automated driving systems(ADS). In the scenarios, graphical diagrams with icons are often used to represent the position and behavior of vehicles. However, such diagrams are not appropriate for developing safety critical systems because their interpretation is ambiguous and can easily cause misunderstandings among engineers. In this paper, we propose a diagram named Vehicle Position Relation Diagram (VPRD) for analysis and design of the scenarios. VPRD allows us to represent a large number of scenarios as a compact model. We also propose a method to encode a VPRD-based model into a propositional logic formula and enumerate all scenarios using a SAT solver. We developed a prototype tool to do such enumeration and conducted some experiments about typical vehicles' behavior. As a result, we found that a large number of scenarios can be described concisely in VPRD and that enumerating scenarios is effective to convince us the reliability of VPRD.

1. はじめに

自動運転車を取り巻く状況は膨大であり，それらすべてに関してテストを実施することは不可能である．そこで，対象範囲を絞り込み，シナリオに基づいてテストを実施することが一般的である．NHTSA(National Highway Traffic Safety Administration) は，系統的に状況の絞り込みを行う手法を提案している [1]．この手法では，まず，ODD(Operational Design Domain) と OEDR(Object and Event Detection and Response) を明確にする．ODD とは，自動運転車が前提とする走行条件であり，OEDR は，それが検知する事象，および，実施する応答である．そして，ODD と OEDR に基づいてテストシナリオを作成する．文献 [2] においては，ODD や外乱を明確にすることにより，シナリオを獲得する手法が提案されている．

シナリオ記述では，自動車の位置関係や動作を表現するために，しばしば，図 1 のような図が用いられる．左から順に，文献 [1]，文献 [2]，文献 [8] のものである．このような図は，理解の助けにはなるが，見る者による理解の齟齬を生じやすく，正確さにも欠ける．よって，自動運転システムのような高信頼システム開発のための記法としては適

[1] 北陸先端科学技術大学院大学
Japan Advanced Institute of Science and Technology
[2] 三菱電機株式会社
Mitsubishi Electric Corporation
[a)] toshiaki@jaist.ac.jp
[b)] tomita@jaist.ac.jp
[c)] tkawai@jaist.ac.jp
[d)] Kawakami.Daisuke@ab.MitsubishiElectric.co.jp
[e)] Chida.Nobuo@ab.MitsubishiElectric.co.jp

切ではないと考えられる．システム開発では，UML[3] や SysML[4] などの，統一された標準記法を使うことが一般的であるが，同様に，シナリオ開発においても，それらに相当する統一された標準記法が必要である．

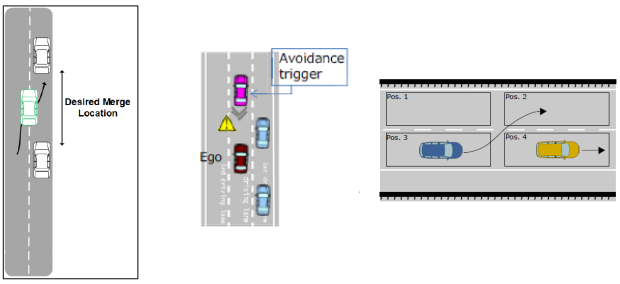

図 1 Scenario Description

シナリオ開発における分析設計の段階では，複数のシナリオを見た目で分析できるくらいのサイズで表現するのが望ましい．一方，シナリオの数は，爆発的に多くなる．シナリオに登場する要素とそれらの位置関係の組み合わせが膨大であるのみならず，それらの時系列的な変化を考慮するからである．よって，シナリオ開発で用いる記法は，大量のシナリオを簡潔に表現できることが望ましい．

本論文では，シナリオ開発のためのグラフィカルな記法である道路位置関係図 (Vehicle Position Relation Diagram, VPRD) を提案する．統一された標準記法の提案を目的にしているため，VPRD の記法は，システム開発における標準記法である UML のアクティビティ図をベースにしている．UML と同様の記法は，標準記法として受け入れやすいと考えたからである．しかしながら，その意味は，アクティビティ図のものとは全く異なる．VPRD は，ペトリネットに基づいて解釈し，SAT ソルバにより取り扱い可能な形式に変換することができる．VPRD では，複数のシナリオを 1 つの図で簡潔に表現でき，SMT ソルバを用いて，自動的にシナリオを探索，および，列挙することができる．

本論文の構成は以下の通りである．2 節で関連研究について述べる．3 節では，VPRD を提案し，4 節でその形式化について述べる．5 節で実験について紹介し，6 節で実験結果について議論を行う．そして，7 節でまとめる．

2. 関連研究

シナリオに基づく手法では，シナリオ (scenario), シーン (scene), 状況 (situation) という言葉が頻繁に使われるため，それらの意味を明確にしておく必要がある．Ulbrich らは，シナリオとそれに関連する語彙を整理し，それらを定義している [5]．シナリオとは，シーン (scene) を時間順に並べたものであり，シーンとは，背景，動的要素などを含む環境のスナップショットであると定義している．本論文では，この用語体系を用いる．シナリオには，様々な抽象度のものがある．Menzel らは，シナリオ開発の工程に応じて，Functional, Logical, Concrete の 3 つの抽象度のシナリオに分類している [6]．Functional シナリオでは，自然言語などを用いて道路の構成や動的オブジェクトなどを列挙し，Logical シナリオでは，それらの状態やパラメータなどを明確にする．Concrete シナリオでは，パラメータの値を決定し，テストを実行できる具体的なシナリオを作成する．Schuldt は，シナリオを，Road-level(L1), Traffic infrastructure(L2), Temporary manipulation of L1 and L2 (L3), Objects (L4), Environment (L5) の 5 つのレイヤに分解している [7]．Bagschik らの文献 [8] に記載されている例を図 1 の右に示す．L1 では，2 つのレーンと硬い側壁，L2 では，道路の境界には白実線が，レーンの境界には白点線が存在することを表現する．L3 に該当するものは無い．L4 では，右の車線において黄色い車の後方を走行している青い車が，左の車線に車線変更していることを表現する．L5 では，図 1 には出現していないが，通常の天気と温度であることを表現する．VPRD では，L4 の Logical シナリオをモデル化することに焦点を当てている．

自動運転シミュレータやツールにおけるデータ交換を目的として，シナリオの統一的なフォーマットが提案されている．ASAM(Association for Standardization for Automation and Measuring Systems) は，OpenSCENARIO[9] と呼ばれるフォーマットを提案している．道路ネットワークなどの静的要素は OpenDrive[10] と呼ばれるフォーマットを用いて表現する．Lanelets[12] は，道路を多角形の区間で分割し，それらを組み合わせることにより，道路ネットワークを表現するフォーマットである．Lanelets を用いて，地図データ OSM(Open Street Map)[11] を拡張する試みも行われている．同様の目的で，シナリオ記述のための DSL(Domain Specific Language) も提案されている [13], [14], [15].

自動車の位置関係を視覚的に表現する手法としては，Traffic Sequence Chart[16]，および，Back らの手法 [17] がある．Traffic Sequence Chart では，UML のシーケンス図を拡張して，視覚的にシーンの列を表現することができる．Back らの手法では，映画のフィルムのように，連続したシーンを列挙する．それぞれのシーンには，追従や接近などの，動的情報も記述することができる．これらの手法では，視覚的にシナリオを表現していることから，個々のシナリオの分析はしやすいと考えられる．しかしながら，大量のシナリオを簡潔に表現する工夫がされておらず，シナリオの網羅的な分析には適していないと考えられる．

自動運転車のシナリオ生成に関しては，既存データからシナリオを抽出する手法 [18]，オントロジーを利用する手法 [8]，サーチベーステストに基づく手法 [19], [20], [21], [22], [23], [24] が提案されている．Althoff らの手法 [18]，および，Bagschik らの手法 [8] は，過

去のデータや知見に基づいてシナリオを生成するものである．サーチベーステストに基づく手法では，それぞれ，独自の探索空間を設定して，適切なシナリオを抽出する．探索空間において，自動車の位置などの情報を取り扱っているが，抽象度は我々のものと比較して低く，Concrete シナリオに近いものである．我々は，シナリオ開発におけるモデル化に焦点を当てており，これらの手法と比較して，抽象度が高い．そして，思考錯誤の結果として獲得されるモデルに基づいて，シナリオを生成する点で，これらの研究とは異なる．

3. 道路位置関係図

3.1 概要

図 2 の左に VPRD による記述例を示す．VPRD は，道路を上空から見たシナリオを表現している．Left Lane, Right Lane は，それぞれ，道路の左レーンと右レーンを表現している．この例には，2 台の車 LCar と RCar が出現する．LCar(0), LCar(1), LCar(2) は，LCar の位置を表現しており，**ボックス**と呼ぶことにする．最初は，LCar は LCar(0) の位置に存在する．LCar(0) から LCar(1)，LCar(1) から LCar(2) の矢印は自動車の移動を表現しており，**ボックス遷移**と呼ぶ．LCar は，最初，LCar(0) にあり，次に，LCar(1) の位置に，その次に LCar(2) の位置に移動する．RCar についても同様である．図 2 の右に，この道路位置関係図が意味するシナリオを木構造で表現している．LCar と RCar は，それぞれ，赤い車，青い車で表現している．最初は，0 に示すとおり，LCar と RCar は並走している．次は，1 と 2 の 2 つある．1 では，RCar の方が LCar より，相対的に前に出て走行している．2 では，その逆で，LCar の方が RCar より，相対的に前に出て走行している．

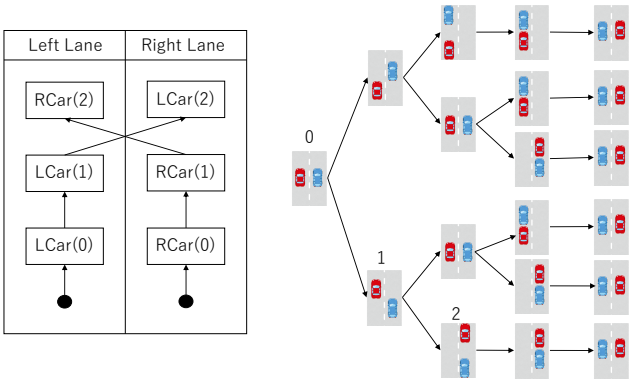

図 2 An Example

このようなシーンの列，すなわち，シナリオは，ペトリネットのトークンの考え方を用いて獲得することができる．初期状態では，トークンが LCar(0) と RCar(0) にある．トークンがあるボックスのみを抜き出してシーンを構成することにより，シーン 0 を獲得できる．次に，トークンが LCar(0) から LCar(1) に移動した場合は，LCar(1) と RCar(0) にトークンがあるので，それらのボックスを抜き出してシーンを構成し，シーン 1 を獲得する．さらに，LCar のトークンを進めると，LCar は Right Lane に移動し，シーン 2 になる．このように，トークンを 1 つずつ次に進めて，トークンが存在するボックスから構成されるシーンを獲得し，シナリオを構成する．この例では，図 2 の右に示すとおり，全部で，6 通りのシナリオを表現していることになる．

3.2 ボックス遷移

ボックス遷移には図 2 に示した通常の遷移の他に 3 種類の遷移がある．図 3 に，それらの記法を示す．左から順に，**存在条件付遷移 (exist conditional transition)**，**非存在条件付遷移 (non-exist conditional transition)**，**同期遷移 (synchronous transition)** と呼ぶことにする．存在条件付遷移，および，非存在条件付遷移では，破線で他のボックスを指定する．そして，前者は，そのボックスにトークンがある時に遷移が発火，後者は，そのボックスにトークンが無い時に遷移が発火する．図 3 の左の例では，LCar(0)，RCar(0) にトークンが存在する時に，LCar(0) から LCar(1) への遷移が発火する．RCar(0) にトークンが無い時は，その遷移は発火しない．図 3 の真ん中の例では，LCar(0) にトークンが存在し，RCar(0) にトークンが存在しない時に，LCar(0) から LCar(1) への遷移が発火する．RCar(0) にトークンが存在する時には，その遷移は発火しない．同期遷移では，複数の遷移が，同時に発火する．図 3 の右の例では，LCar(0) および RCar(0) にトークンが存在する時，2 つの遷移が同時に発火し，次ステップでは，LCar(1)，RCar(1) にトークンが存在する．LCar(0) と RCar(1)，および，LCar(1) と RCar(0) にトークンが存在するシーンは存在しない．

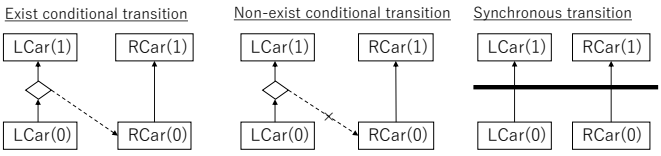

図 3 Transitions

図 4 に車線変更の例を示す．Left Lane, Right Lane, Crossing Lane は，それぞれ， 左車線，右車線，車線をまたいでいる状況を表現している．この例では，LCar, RCar, EgoCar の 3 台の自動車が存在し，EgoCar が左車線から右車線に変更する状況を表現している．EgoCar(0) から EgoCar(1) の存在条件付遷移には，else が付いている EgoCar(0) から EgoCar(7) への遷移が付随している．この

遷移は，RCar(1) にトークンが存在しない時に発火する．これは，非存在条件付遷移と同じであるが，見やすさのため，このような記法にしている，いわゆる，syntax sugar である．なお，EgoCar(6) と LCar(2)，および，EgoCar(3) と RCar(4) は，それぞれの車線において同じ高さの位置にあるので，それらの衝突表現している．

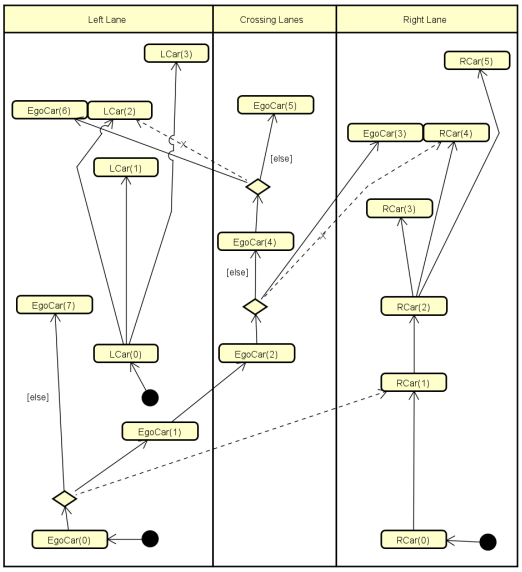

図 4 Lane Change

3.3 メタモデル

VPRD のメタモデルを図 5 に示す．VPRD は複数のレーン (Lane) から構成されており，それぞれのレーンはボックス (Box) を持つ．ボックスには，Concrete ボックスと Parametric ボックスの 2 種類がある．Concrete ボックスでは，具体的な位置が整数で表現されており，VPRD でボックスが置かれた位置により決定する．それぞれのボックス b の位置を $Pos(b)$ で表現することにすると，例えば，図 2 のボックスが Concrete ボックスの場合，$Pos(LCar(0)) = Pos(RCar(0)) = 0$，$Pos(LCar(1)) = Pos(RCar(1)) = 1$，$Pos(LCar(2)) = Pos(RCar(2)) = 2$ と割り当てる．Parametric ボックスは，位置が変数として表現されており，他のボックスとの関係が制約として与えられている．制約には，例えば，衝突，追従，先行などがある．例えば，図 2 のボックスが Parametric ボックスの場合，$Pos(LCar(0)) < Pos(LCar(1))$，$Pos(RCar(0)) < Pos(RCar(1))$，$Pos(LCar(1)) < Pos(LCar(2))$，$Pos(RCar(1)) < Pos(RCar(2))$，などの制約を割り当てる．Parametric ボックスの場合は，位置に自由度をもたせることができる．ボックス遷移には，3.2 節で示したとおり，通常遷移，存在条件付遷移，非存在条件付遷移，同期遷移があり，それぞれ，Normal, Exist Conditional，Non-exist Conditional，Synchronous クラスにより表現されている．

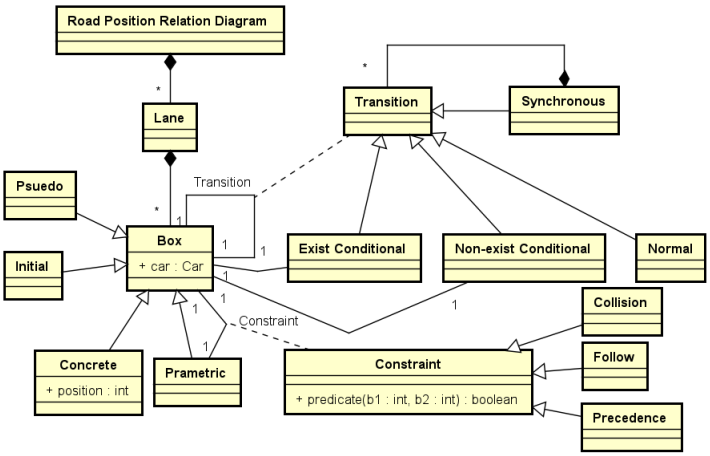

図 5 Metamodel

4. 道路位置関係図の形式化

4.1 基本要素

本節では，VPRD の形式化を行う．まず，VPRD を構成する基本要素を定義する．

定義 1 自動車と Box

Car を自動車を表現する識別子の集合とする．自動車の位置を表現する矩形の集合を Box とし，以下で定義する．

$$Box = Car \times N$$

自動車 $c \in Car$ の位置 $(c, i) \in Box, i \in N$ を $c(i)$ と書くことにする．また，自動車 $c \in Car$ に限定した位置集合を $Box(c)$ と表現し，$Box(c) = \{(x, i) \in Box | x = c\}$ とする．

定義 2 遷移グラフ

自動車 c の位置の遷移を表現する遷移グラフを $G(c)$ で表現し，以下で定義する．

$$G(c) = (Box(c), E(c), b_0(c))$$
where $b_0(c) \in Box(c)$,
$E(c) = E_n(c) \uplus E_e(c) \uplus E_{ne}(c)$,
$E_n(c) \subseteq Box(c) \times Box(c)$,
$E_e(c) \subseteq Box(c) \times Box(c) \times Box$,
$E_{ne}(c) \subseteq Box(c) \times Box(c) \times Box$

$E_n(c), E_e(c), E_{ne}(c)$ は，それぞれ，通常遷移，存在条件付遷移，非存在条件付遷移の集合を表現している．$(b, b') \in E_n$ は b から b' へのボックス遷移，$(b, b', c) \in E_e$ は，ボックス c が存在する時に発火する b から b' へのボックス遷移，$(b, b', c) \in E_{ne}$ は，ボックス c が存在しない時に発火する b から b' へのボックス遷移を意味している．

定義 3 VPRD の構造

$n \in N$ 台の自動車 $c_1, \cdots, c_n$ の位置の遷移を表現する遷移グラフを G で表現し，以下で定義 する．

$$G = (\{G(c_i)|0 \leq i \leq n\}, E_s)$$
$$\text{where } E_s \subseteq 2^{Box \times Box} s.t. \forall E \in E_s\ t \in E. \exists i. t \in E_n(c_i)$$

なお，E_s は同期遷移を表現している．E_s は，ボックス遷移集合上の集合族である．その要素はボックス遷移の集合であり，同時に発火する遷移を表現している．

定義 4　自動車の位置関係と走行レーン

$(X, \preceq)$ を全順序集合とする．自動車の位置を写像 $Pos : Box \to X$ で表現する．以降，全順序集合 X を自然数 N とするが，一般性を失わない．また，自動車の走行レーンを写像 $Lane : Car \to N$ で表現する．

定義 5　シナリオ

n 台の自動車 $c_i \in Car, 0 \leq i \leq n, n \in N$ のシナリオを π と表現する．π は以下の型を持つ写像である．

$$\pi : N \to \prod_{0 \leq i \leq n} Box(c_i)$$

π のそれぞれの要素 $\pi(j), j \in N$ をシーンと呼ぶ．

4.2　ボックス遷移の形式化

定義 3 に示した通り，VPRD はグラフの集合である．また，3.1 節に示した通り，個々のグラフは，トークンを用いて解釈することができる．ここで，それぞれのグラフにはトークンが唯一に存在する．よって，個々のグラフは，Safe ペトリネットのサブクラスとみなすことができる．Ogata らは [25]，Safe ペトリネットを命題論理式にエンコードし，SAT ソルバを用いて到達性解析を行う手法を提案している．本論文では，Ogata らの手法に基づいて，有限ステップの範囲で，VPRD を命題論理式に変換する．そして，SAT ソルバを用いて，到達解析を実施し，シナリオを網羅的に列挙する手法を提案する．なお，このようなシナリオの列挙手法は，VPRD の意味を決めていることにもなる．すなわち，VPRD の意味をシナリオの集合として定義しているのである．

$n \in N$ 台の自動車 $c_i \in Car, 0 \leq i \leq n$ のシーン $\pi(j) = (b_1, \cdots, b_n) = (c_1(m_1), \cdots, c_n(m_n)), j \in N$ を命題ベクトル $S = (s[b_1], \cdots, s[b_n])$ で表現する．ここで，$s[b_i]$ は自動車 c_i が位置 $b_i \in Box(c_i)$ に存在する時のみ真である命題とする．自動車 c のグラフ $G(c)$ における遷移 $t(c) \in E(c)$ によるシーン S から S' への変化は，以下の関数 $T_{t(c)}(S, S')$ で定義される．

$t(c)$ が通常遷移の時 $(t(c) \in E_n)$，

$$T_{t(c)}(S, S') = \bigwedge_{b \in \bullet t(c) \setminus t(c)\bullet} s[b] \wedge \bigwedge_{b \in \bullet t(c)} \neg s[b]' \wedge \bigwedge_{b \in t(c)\bullet} s[b]' \wedge \bigwedge_{b \in Box(c) \setminus (\bullet t(c) \cup t(c)\bullet)} (s[b] = s[b]')$$

ここで，$\bullet t(c)$ と $t(c)\bullet$ は，それぞれ，ボックス遷移 $t(c)$ の遷移前状態の集合，遷移後状態の集合を表現している．これは，ペトリネットの慣例に習う記法である．なお，VPRD の通常遷移，存在条件付遷移，非存在条件付遷移では，それらは，単一要素のみを持つ集合となる．

$t(c) = (b, b', b_c)$ が存在条件付遷移の時 $(t(c) \in E_e)$，

$$T_{t(c)}(S, S') = s[b_c] \wedge \bigwedge_{b \in \bullet t(c)} s[b] \wedge \bigwedge_{b \in \bullet t(c) \setminus t(c)\bullet} \neg s[b]' \wedge \bigwedge_{b \in t(c)\bullet} s[b]' \wedge \bigwedge_{b \in Box(c) \setminus (\bullet t(c) \cup t(c)\bullet)} (s[b] = s[b]')$$

$t(c) = (b, b', b_c)$ が非存在条件付遷移の時 $(t(c) \in E_{ne})$，

$$T_{t(c)}(S, S') = \neg s[b_c] \wedge \bigwedge_{b \in \bullet t(c)} s[b] \wedge \bigwedge_{b \in \bullet t(c) \setminus t(c)\bullet} \neg s[b]' \wedge \bigwedge_{b \in t(c)\bullet} s[b]' \wedge \bigwedge_{b \in Box(c) \setminus (\bullet t(c) \cup t(c)\bullet)} (s[b] = s[b]')$$

$t_1, \cdots, t_n, n \in N$ が同期遷移の時 $(\{t_1, \cdots, t_n\} \in E_s)$，

$$T_{t_1, \cdots, t_n}(S, S') = \bigwedge_{b \in \bullet t_1} s[b] \wedge \bigwedge_{b \in \bullet t_1 \setminus t_1\bullet} \neg s[b]' \wedge \bigwedge_{b \in t_1\bullet} s[b]' \wedge \cdots \wedge \bigwedge_{b \in \bullet t_n} s[b] \wedge \bigwedge_{b \in \bullet t_n \setminus t_n\bullet} \neg s[b]' \wedge \bigwedge_{b \in t_n\bullet} s[b]' \wedge \bigwedge_{b \in Box \setminus (\cup_{0 \leq i \leq n} (\bullet t_i \cup t_i\bullet))} (s[b] = s[b]')$$

$n \in N$ 台の自動車 $c_i \in Car, 0 \leq i \leq n$ の初期ボックスを以下の関数 $I(S)$ で定義する．

$$I(S) = \bigwedge_{0 \leq i \leq n} s[b_0(c_i)] \wedge \bigwedge_{b \neq b_0(c_i), b \in Box(c_i), 0 \leq i \leq n} \neg s[b]$$

$n \in N$ 台の自動車の遷移関数は以下の関数 $T(S, S')$ で定義される．

$$T(S, S') = \bigvee_{1 \leq i \leq n, t(c_i) \in E(c_i)} T_{t(c_i)}(S, S') \vee \bigvee_{ts \in E_s} T_{ts}(S, S') \vee D(S, S')$$

ここで，$D(S, S')$ は，VPRD におけるどのグラフも遷移しないことを表現しており，以下で定義される．

$$D(S, S') = \bigwedge_{b \in Box(c_i), 0 \leq i \leq n} s[b] = s[b]' \wedge \bigwedge_{b \in \bullet t(c_i), t \in E_n(c_i), 0 \leq i \leq n} \neg s[b] \wedge \bigwedge_{(b, b', c) \in E_e(c_i), 0 \leq i \leq n} \neg(s[b] \wedge s[c]) \wedge \bigwedge_{(b, b', c) \in E_{ne}(c_i), 0 \leq i \leq n} \neg(s[b] \wedge \neg s[c]) \wedge \bigwedge_{t \in E_s} \bigvee_{b \in \bullet t} \neg s[b]$$

ここで，$\bullet t$ の t はボックス遷移集合であり，これまでの記法とは異なる．しかしながら，$\bullet t$ の意味は同様であり，それぞれのボックス遷移の遷移前ボックスを集めた集合を表現するものとする．

この関数 $T(S, S')$ を用いて，$k \in N$ ステップの遷移は以下の命題論理式となる．

$$I(S_0) \wedge T(S_0, S_1) \wedge T(S_1, S_2) \wedge \cdots \wedge T(S_{k-1}, S_k)$$

この命題論理式の充足可能性を判定し，充足する場合は，その割当がシナリオを表現している．充足可能性判定により獲得された命題 $s[b]$ への充足可能な真偽の割当を $\sigma[b]$ で表現すると，それとは異なる割当を表現する制約は $\neg(\wedge_{b \in Box} s[b] \leftrightarrow \sigma[b])$ である．この制約を追加して充足可能判定を繰り返すことにより，有限ステップの範囲内で，すべてのシナリオを列挙することが可能となる．なお，定義 2 により，自動車の遷移グラフには循環が含まれても良い．循環が含まれる場合は，シナリオの数は無限に存在することになり，定義 5 のシナリオの形では網羅的に列挙することはできない．網羅的に列挙するためには，シナリオにおいて循環を記述できるようにするなどの工夫が必要である．一方，遷移グラフに循環が含まれない場合は，有限ステップで，すべてのシナリオを網羅することができる．

5. 実験

Python と Z3py[27] を用いて，VPRD から命題論理式へ変換し，シナリオを列挙するプロトタイプツールを実装した．なお，本ツールは，GitHub で公開してある [29]．以下では，このプロトタイプツールを用いた実験について紹介する．

5.1 基本動作

自動運転の挙動には様々なものがあり，それらの基本的な動作の 1 つは車線変更である．例えば，追い越し，割り込み，合流といった動作は，車線変更の組み合わせである．そこで，車線変更を対象として，シナリオに参加する自動車の台数を変更してモデル化を行い，シナリオを列挙した．結果を表 1 に示す．Car は自動車の台数, Box はボックスの数, Trans は遷移の数 (括弧の中は，左から順に，通常遷移，存在条件付遷移または非存在条件付遷移，同期遷移の数), Total はシナリオの数，Col は自動車が衝突するシナリオの数である．図 4 の例は，表 1 の case 2-2 である．この場合は，EGO Car が 1 台，POV(Principal Other Vehicle) が 2 台の，合計 3 台の自動車が参加している．case 2-2 では，衝突するシナリオは 66 通りある．EgoCar が EgoCar(3) まで遷移した後に，RCar が RCar(2) から RCar(4) に遷移する場合が存在するためである．同期遷移を導入することにより衝突を回避するシナリオにすることができる (case 2-1)．また，すべての遷移の存在条件付遷移，非存在条件付遷移，同期遷移を通常遷移に変更したものが case 2-3 である．case の番号の後の記号 nc, c, n は，それぞれ，衝突しないシナリオのみを含む場合，(非) 存在条件付遷移はあるが衝突するシナリオを含む場合，すべて通常遷移の場合を表現している．

5.2 パフォーマンス評価

シナリオ生成のパフォーマンスに関する実験を実施した．

表 1 Results of Change Lane

case	Car	Box	Trans	Total	Col
1-1(nc)	2	11	7 (3, 2, 2)	4	0
1-2(n)	2	11	9 (9, 0, 0)	72	20
2-1(nc)	3	18	13 (7, 4, 2)	150	0
2-2(c)	3	18	15 (9, 6, 0)	522	66
2-3(n)	3	18	15 (15, 0, 0)	6480	1260
3-1(nc)	4	22	15 (8, 4, 3)	195	0
3-2(c)	4	22	19 (9, 10, 0)	1038	321
3-3(n)	4	22	19 (19, 0, 0)	169560	52240

用いた PC は，iMAC，CPU: 3.6GHz Intel Core i9, メモリ:128G である．図 6 の左上に対象としたモデルを示す．２台の自動車が存在し，それぞれのボックスの数を n 個とする．そして，n の数を増やしてシナリオを生成した．図 6 の上段左のグラフは n と生成されたシナリオの数の関係を示している．図 6 の上段右のグラフは，シナリオの列挙にかかった時間を示している．赤色の線は制約を追加するのにかかった時間，青色の線はすべてのシナリオを列挙するのにかかった時間を示している．なお，時間の単位は秒であり，以降，同様である．なお，横軸は n の値である．図 6 の下段は，1 つのシナリオを求める場合の結果である．横軸は n の値を示している．左のグラフは，シナリオを求めるのにかかった時間 (単位は秒) を示している．青色の線は制約の追加にかかった時間，赤色の線は充足可能性判定にかかった時間，緑色の線は CNF(Conjuctive Normal Form) に変換するのにかかった時間を示している．真ん中のグラフは，用いたメモリ容量 (単位は MB)，右は命題の数 (赤色の線) と CNF 節の数 (青色の線) を示している．なお，Tseitin[28] 変換により CNF に変形している．

図 6 に示したとおり，モデルのサイズが大きくなると，シナリオ生成にかかる計算時間が非常に長くなる．なお，図 6 の左上のモデルでは，シナリオの数を理論的に求めることができ，それは $\frac{(2n)!}{(n!)^2}$ である．シナリオの数は，n に関して階乗的に増加することがわかる．シナリオの数に応じて計算時間も長くなる．なお，$n = 10$ の場合，シナリオの数は，184756 個であり，それらを列挙するのにかかる時間は 394890.606019 秒 (約 4.57 日) である．一方，1 つのシナリオ生成には，それほど，時間がかかっていない．$n = 100$ の場合でも，2807.6733 秒である．また，その時間の多くは，制約の追加にかかっている．プロトタイプツールでは，Python を用いて 4 節に示した方法により制約を生成している．n に応じて，関数 $T(S, S')$ を展開する回数が多くなり，制約式のサイズも大きくなる．これにより，Python による制約追加にかかる時間が長くなっているのである．メモリ容量はそれほど消費しておらず，メモリサイズが問題になることは無いと考えられる．また，命題変数に比較して，CNF の節は非常に多くなっている．$n = 100$ の場合，命題変数の数が 120809 個に対して，CNF 節の数は

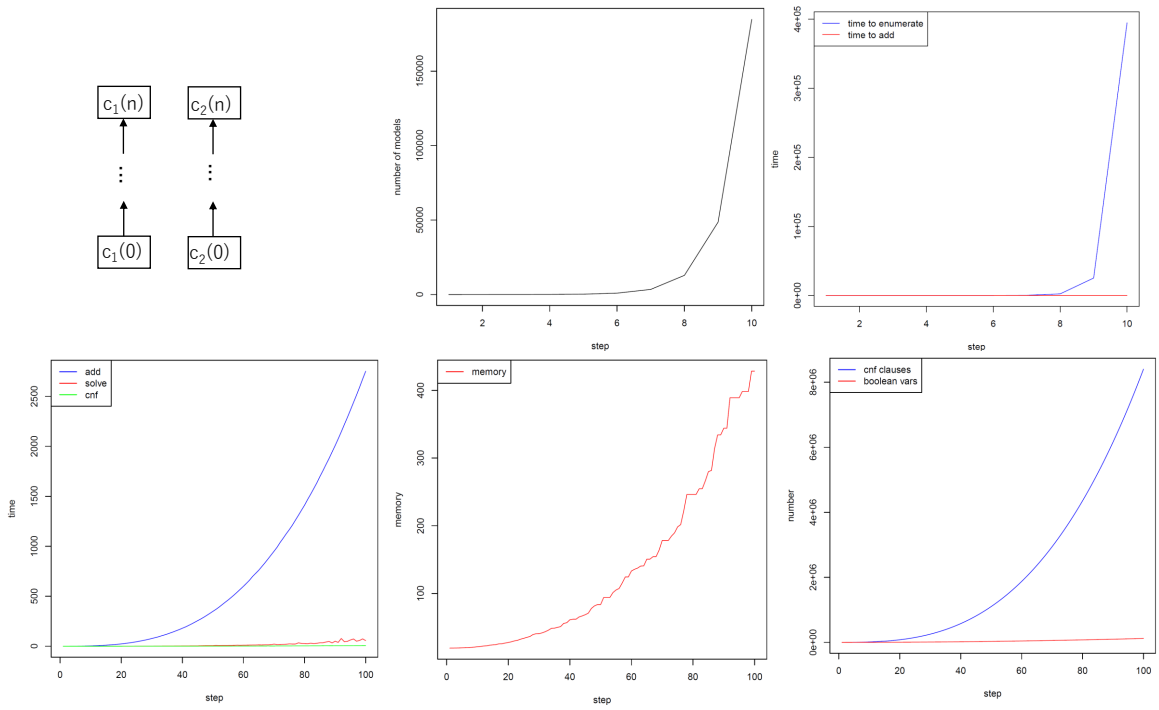

図 6 Performance

8403006 個である．それでも，CNF への変換にかかる時間，および，充足可能性判定にかかる時間は，制約追加にかかる時間に比べて非常に小さい．よって，1 つのシナリオ生成にかかる時間に関しては制約の大きさが主要因であり，シナリオ列挙にかかる時間に関しては，制約追加よりは，シナリオ数の影響が大きいと考えられる．

6. 議論

表 1 に示した通り，VPRD により，大量のシナリオを簡潔に表現できたと言える．自動車が 4 台出現する case 3-1, 3-2, 3-3 では，195〜169560 個のシナリオを 22 個のボックスと 15〜19 個のボックス遷移で表現できている．VPRD では，多くのシナリオをコンパクトに表現できており，視覚的にレビューすることが可能である．それが表現しているシナリオの数は膨大であり，条件遷移の追加 (削除) による影響の分析などには，シナリオを列挙する仕組みが不可欠である．本論文では，SAT ソルバを用いて，網羅的なシナリオを自動的に獲得する手法も提案した．

一方，シナリオの数が膨大であると，列挙するだけでは，意図しないシナリオの存在などの確認が困難である．case 2-2 では，非存在条件付遷移により，衝突する場所に RCar が存在する場合には EgoCar は右車線に車線変更を行わないようにしている．左車線への車線変更も同様である．しかしながら，この場合，66 個の衝突が発生するシナリオが含まれる．このような衝突は，同期遷移を用いて EgoCar と RCar のボックスの組みを同時に遷移させることにより回避できる．これは，case 2-1 に相当する．case 2-1 では，150 個のシナリオを表現しており，個々のシナリオが妥当であるか人手で確認するには多すぎる量である．case 3-1, case 3-2 では，さらに，その数が多くなっている．VPRD では，SAT ソルバを用いて列挙を行っているため，柔軟に制約を追加することができる．そこで，シナリオに対して期待する性質や期待しない性質を表現する制約を適宜追加することが考えられる．これにより，モデルの検証やシナリオの絞り込みが可能になると考えている．このような性質を記述するための言語の提案は今後の課題である．

5.2 節に示したとおり，パフォーマンスに関しては，シナリオの数が主要因であると考えれる．シナリオの数は，基本的には，ボックス遷移の数に関して階乗的に増えるが，制約を追加することにより，その数を抑制することができる．表 1 に示したとおり，同じ自動車の台数では，(nc) が一番少なく，(c) が次に少なく，(n) が一番多い．よって，同期遷移や非存在条件付遷移を用いて制約を追加することにより，シナリオの数を減らすことができる．他にも，相対的距離に関する制約を追加することなどが考えられる．例えば，図 4 の例では，EgoCar(4) と RCar(5) の組み合せは重要であるが，EgoCar(0) と RCar(5) の組み合せは，それほど重要ではない．そこで，ボックスの距離が一定の範囲内のシーンのみ列挙することもできる．図 6 の例で，それぞれの隣り合うボックスの距離を 1 とする．そして，ボックスの距離が 3 未満のシーンのみに限定すると，$n = 10$ の場合，39366 個となる．限定が無い場合が 184756 個であるので，約 79%削減される．このように，様々な制約を追加することにより，シナリオの数の抑制が可能である．

7. まとめ

本論文では，自動運転システムを対象としたシナリオ開発のためのグラフィカルな記法である VPRD を提案した．VPRD は，L4 の Logical シナリオを分析設計する際に用いることを想定している．そのようなシナリオの抽象度は高いが，その数は膨大となりがちである．シナリオの分析設計の際には，複数のシナリオの選択肢を見比べて，試行錯誤しながらモデル化することになる．よって，複数のシナリオの全体像が見た目で把握できることが望ましく，VPRD を用いると，それが可能である．さらに，SAT ソルバを用いて，VPRD のモデルが表現するシナリオを列挙することができる．これにより，網羅的なシナリオを獲得するだけでなく，モデルが意図したものになっているか確認することも可能である．現在は，高い抽象度のシナリオを対象にしているが，今後は，地図データ [11], [12] や交通データ [26] などの具体的なデータと対応づける方法について検討していきたい．これにより，実際の道路や交通を対象としたモデル化と分析が可能になると考えている．また，複雑な道路ネットワークトポロジへの対応についても，今後，検討する．

参考文献

[1] E. Thorn, S. Kimmel and M. Chaka: A Framework for Automated Driving System Testable Cases and Scenarios, NHTSA, No. DOT HS 812 623, 2018.
[2] 自動運転の安全性評価フレームワーク，日本自動車工業会，2020.
[3] Unified Modeling Language Specification version 2.5.1, Object Management Group, 2017.
[4] OMG System Modeling Language Specification version 1.6, Object Management Group, 2019.
[5] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt and M. Maurer: Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving, pp. 982–988, International Conference on Intelligent Transportation Systems, 2015
[6] T. Menzel, G. Bagschik and M. Maurer: Scenarios for Development, Test and Validation of Automated Vehicles, International Conference on Intelligent Vehicles Symposium, pp. 1821–1827, 2018.
[7] F.Schuldt: Towards testing of automated driving functions in virtual driving environments, Ph.D. dissertation, Technische Universität Braunschweig, 2017.
[8] G. Bagschik, T. Menzel and M. Maurer: Ontology based Scene Creation for the Development of Automated Vehicles, International Conference on Intelligent Vehicles Symposium, pp. 1813–1820, 2018.
[9] ASAM OpenSCENARIO V2.0, https://www.asam.net/standards/detail/openscenario/
[10] ASAM OpenDrive, https://www.asam.net/standards/detail/opendrive/
[11] OSM (Open Street Map): https://www.openstreetmap.org/
[12] P. Bender, J. Ziegler and C. Stiller, Lanelets: Efficient map representation for autonomous driving, Intelligent Vehicles Symposium Proceedings, pp. 420-425, 2014.
[13] X. Zhang, S. Khastgir and P. Jennings, Scenario Description Language for Automated Driving Systems: A Two Level Abstraction Approach, International Conference on Systems, Man, and Cybernetics, pp. 973-980, 2020.
[14] R. Queiroz, T. Berger and K. Czarnecki, GeoScenario: An Open DSL for Autonomous Driving Scenario Representation, Intelligent Vehicles Symposium, pp. 287-294, 2019.
[15] F. Bock, C. Sippl, A. Heinz, C. Lauer and R. German, Advantageous Usage of Textual Domain-Specific Languages for Scenario-Driven Development of Automated Driving Functions, International Systems Conference, pp. 1-8, 2019.
[16] W. Damm, R. Galbas: Exploiting Learning and Scenario-Based Specification Languages for the Verification and Validation of Highly Automated Driving, SEFAIAS@ICSE, pp.39-46, 2018.
[17] J.Bach, S. Otten and E. Sax, Model based scenario specification for development and test of automated driving functions, Intelligent Vehicles Symposium, pp. 1149-1155 2016.
[18] M. Althoff, M. Koschi and S. Manzinger: CommonRoad: Composable benchmarks for motion planning on roads, Intelligent Vehicles Symposium, pp. 719-726, 2017.
[19] R. Ben Abdessalem, S. Nejati, L. C. Briand and T. Stifter: Testing advanced driver assistance systems using multi-objective search and neural networks, International Conference on Automated Software Engineering, pp. 63-74, 2016.
[20] M. Althoff and S. Lutz: Automatic generation of safety-critical test scenarios for collision avoidance of road vehicles, Intelligent Vehicles Symposium, pp. 1326–1333, 2018.
[21] H. Beglerovic, M. Stolz, and M. Horn. Testing of autonomous vehicles using surrogate models and stochastic optimization. International Conference on Intelligent Transportation Systems, pp. 1–6, 2017.
[22] F. Hauer, A. Pretschner, and B. Holzmüller: Fitness functions for testing automated and autonomous driving systems. International Conference on Computer Safety, Reliability, and Security, pp. 69-84. Springer, 2019.
[23] S. Khastgir, G. Dhadyalla, S. Birrell, S. Redmond, R. Addinall, and P. Jennings: Test scenario generation for driving simulators using constrained randomization technique. Technical report, SAE Technical Paper, 2017.
[24] A. Calo, P. Arcaini, S. Ali, F. Hauer and F. Ishikawa: Generating Avoidable Collision Scenarios for Testing Autonomous Driving Systems, ICST, pp. 375-386, 2020.
[25] S. Ogata, T. Tsuchiya, T. Kikuno: SAT-Based Verification of Safe Petri Nets, ATVA, pp.79-92, 2004.
[26] V. Punzo, M. T. Borzacchiello, and B. Ciuffo: On the assessment of vehicle trajectory data accuracy and application to the Next Generation SIMulation (NGSIM) program data, Transportation Research Part C, vol. 19, pp. 1243.1262, 2011.
[27] L. de Moura and N. Bjorner, Z3: an efficient smt solver, International Conference on Tools and Algorithms for the Construction and Analysis of Systems, pp. 337–340, 2008.
[28] G. C. Tseitin: On complexity of derivations in propositional calculus. In Studies in Constructive Mathematics and Mathematical Logic, part II, pp.466–483, 1970.
[29] https://github.com/toshiaki-jaist/rprd