社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2004－ＩＴＳ－16　(12)
2004／3／2

EVALPSNに基づく鉄道連動装置安全性検証及びシミュレータ

中松和己[1]　木内洋介[2]　鈴木淳之[2]

[1]姫路工業大学　〒670-0092 兵庫県姫路市新在家本町1-1-12 姫路工業大学環境人間学部
[2]静岡大学　〒432-8011　静岡県浜松市3-5-1 静岡大学情報学部情報科学科
E-mail: [1]nakamatu@hept.himeji-tech.ac.jp
[2] {cs9025,suzuki}@cs.inf.shizuoka.ac.jp

概要 – 鉄道連動装置の故障は深刻な問題となるので、できるだけ速く復旧させる必要がある。そのため故障をできるだけ速く検知するために *EVALPSN*(拡張ベクトル真理値付き論理プログラム) と呼ばれるパラコンシステント論理プログラムを使い、鉄道連動装置の安全性検証を行い、列車の安全性を保障させる方法を紹介する。また、この方法を拡張し、乱れた列車のダイヤグラムを早急に直すために、時相区間論理を加えた *EVALPSN* に基づく鉄道ダイヤの復旧支援システムの提案を計画している。このシステムは、鉄道連動装置安全性検証システムとダイヤグラムをシミュレーションするシステムからなる。本論文では、*EVALPSN* に基づいた鉄道連動装置の安全性検証システムと、それに信号制御を加えたシミュレーションシステムのプロトタイプを紹介する。

キーワード： 鉄道連動装置安全性検証、真理値付き論理プログラム、ディフィージブル義務推論、EVALPSN、時相区間論理

Railway Interlocking Safety Verification and its Simulator Based on EVALPSN

Kazumi Nakamatsu[1]　Yosuke Kiuchi[2]　and　Atsuyuki Suzuki[2]

[1]School of H.E.P.T. Himeji Inst. Of Tech　HIMEJI　670-0092,JAPAN

[2]School of Information Shizuoka Univ. HAMAMATSU 432-8011,　JAPAN

E-mail: 1 nakamatu@hept.himeji-tech.ac.jp　2 {cs9025,suzuki}@cs.inf.shizuoka.ac.jp

Abstract Railway operation disorder is a serious problem and expected to be recovered as soon as possible. we introduce an EVALPSN(Extended Vector Annotated Logic Program with Strong Negation) based railway interlocking safety verification. We are planning to provide a railway diagram recovery system based EVALPSN with interval temporal reasoning. The system consists of a railway interlocking safety verification system and a railway diagram simulation system. In this paper, we introduce the ideas of the railway interlocking safety verification system based on EVALPSN and its prototype simulation system with signal control.

Keyword Railway Interlocking Safety verification, Annotated logic program, Defeasible deontic reasoning, EVALPSN, interval temporal logic

1　序論

線路上での事故や、列車の故障等のさまざまな理由での列車工程上の不備は深刻な問題になるので、できるだけ速く復旧させるべきである。そのため、我々はEVALPSNを用いて鉄道連動装置の安全性検証をEVALPSNを用いて行う。また、将来的な展望として、ダイヤグラムを早急に回復させるために時相区間論理を加えたEVALPSNで列車運行ダイヤの復旧を行うためのシステムを構築することを計画している。この論文で、我々はEVALPSNに基づく鉄道連動装置安全性検証システムと、それに信号制御を加えたシミュレーションシステムのプロトタイプを紹介する。

真理値付き論理[2]はパラコンシステント論理に属していて、Blair & Subrahmanian [1]によって、論理プログラミングの観点から発達していった。その応用例として、3種類の真理値付き論理プログラミングに発展していった。第1に、非単調推論を扱うためにALPSN(強否定を伴う真理値付き論理プログラム)を紹介する。それは[5, 6]. で非単調推論等への論理的な基盤となることを示す。第2に、ディフィージブル推論やファジー推論を扱うために、VALPSN(ベクトル真理値付き論理プログラム)と[7, 8, 9]でVALPSNによるディフィージブル推論に基づいた衝突を起こす問題への解決方法を紹介する。最後にディフィージブル義務推論を扱うために、VALPSNをEVALPSN(拡張ベクトル真理値付き論理プログラム)に[10, 11]で拡張する。EVALPSNは競合する複数の候補から1つの候補を選択する能力をもち、さらに義務、禁止、許可といった様相概念を扱える論理プログラムである。我々が行動を決定する場合、世の中の規範(例えば倫理とか法律)にしたがって、行動を決定する場合がある。このような場合、義務、禁止、許可といった様相概念に基づき行為を推論する。また複数の競合(矛盾)する候補が

あり、その中から1つだけ選択し実行したいという場合がある。このようなことを扱う論理として前述したディフィージブル義務論理があり、これを扱うためにEVALPSNを作成した。我々はEVALPSNを次の2つの場合、知的行動制御及び安全性検証 [12, 13, 14]. に適応させる。EVALPをこの2つに適応させる理由として、次のような事柄が挙げられる。安全性の検証は法律やルールなどのような規範を表す記述を必要とする。よって、通常の論理プログラミングと比較して、義務論理は規範をより自然に記述できる。さらに、論理プログラムはその導出過程そのものが証明過程となっているので、検証という分野に対して非常に有効であると考えたからである。

このEVALPSNを用いて、将来的に見てEVALPSNが電子回路として、マイクロチップに実装する事ができ、さらに即時制御 [15] に適している事を示す。

鉄道連動装置安全性検証システムは鉄道システムへの最も重要な部分の一つである。近年、鉄道連動装置の安全性検証を行うための形式的な方法として、論理言語が注目されてきた。本論文において、その方法の元となる考えは、Morley's Ph.D の論文 [4] のイギリス鉄道を例にして紹介している。

Morley's Ph.D は3つの安全特性として、鉄道連動装置の安全性検証を保証できるように提案し、高階論理言語HOL[3] で安全特性に基づいて安全性を検証する形式的な方法を提案した。しかし、その方法は、高階論理言語の不完全な部分も含んでいる。我々は、EVALPSNで表現した同じ安全特性を紹介し、そして、鉄道連動装置の安全性がEVALPSNプログラミングによって検証される事を示す。

今回鉄道連動装置安全性検証において、EVALPSNには強否定 (Strong negation) を持たないが、もし強否定を含んでいれば、それは扱いやすい階層化されたプログラムとなる。それゆえ、論理プログラムで安全性検証を実装するのは簡単である。EVALPSNを使うもう一つの理由として、時相区間論理で扱うような別の推論を扱うために拡張する事が容易にできるという事である。

実際、列車ダイヤグラムシミュレーションシステムは、インターバルタイムというもう一つの真理値を持つEVALPSNになる。更に、そのシステムにおいてEVALPに基づいた即時処理進行が可能となり、電子回路として、ハードウェアとしてマイクロチップに実装できる [15]。　この論文で、我々は信号制御を含むEVALPSNに基づいた鉄道連動装置安全性検証を行うシミュレータのプロトタイプを紹介する。本論文の構成は以下のようになっている。最初にEVALPSNを簡潔に解説し、それに基づく安全性検証の基本的な例を紹介する。次に、区分内進路開放要求 (サブルートリリース) や、進路確保要求 (ルートリクエスト) の単純な例を使って、EVALPSNに基づく鉄道連動装置の安全性検証の方法を記述する。次に、EVALPSNを使った信号制御を含む鉄道連動装置の安全性検証を示すシミュレーションシステムのプロトタイプを紹介する。最後に結論と将来的なシステムの展望について述べる。

2 EVALPSN

一般的に**アノーテーション**と呼ばれる真理値が真理値付き論理プログラムに明示的に取り付けられている。例えば、p リテラルで, μ はアノーテーションである。その時、$p{:}\mu$ は**真理値付きリテラル**と呼ばれる。

アノーテーションの集合は完備束で構成されている。VALPSNのアノーテーションは非負の整数で構成され、それはベクトルアノーテーションと呼ばれる2次元ベクトルになっている。そして、ベクトルアノーテーションの完備束 $\mathcal{T}_v$ は次のように定義される。

$$\mathcal{T}_v = \{\ (x,y)\ \mid\ 0 \le x \le n, 0 \le y \le n, \\ x, y \text{ and } n \text{ are integers }\}.$$

束 $\mathcal{T}_v$ の順序関係は $\preceq$ の記号で定義され、次のように定義される。

ベクトル $\vec{v_1} = (x_1, y_1)$ かつ、 $\vec{v_2} = (x_2, y_2)$ の時、

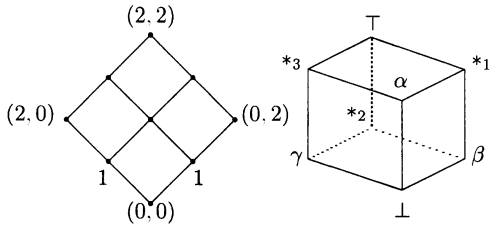
$$\vec{v_1} \preceq \vec{v_2} \iff x_1 \le x_2 \ \text{ and } \ y_1 \le y_2.$$

ベクトル真理値付きリテラル $p{:}(i,j)$ の中でベクトル真理値の構成要素 i, j はそれぞれリテラル p の正の情報量、負の情報量を表す。例えば、整数 $n = 2$ とするとベクトル真理値付きリテラル $p{:}(2,1)$ は「リテラル p は正の強さ2で負の強さ1であることが知られている」と解釈される。次に我々はVALPSNをEVAPSNに拡張する。EVALPSNのアノーテーションは**拡張ベクトル真理値**と呼ばれ、$[(i,j),\mu]$ の形で表される。最初の構成要素 (i,j) はVALPSNのベクトル真理値のような2次元ベクトルで、2番目の構成要素

$$\mu \in \mathcal{T}_d = \{\bot, \alpha, \beta, \gamma, *_1, *_2, *_3, \top\},$$

は義務概念か矛盾を表す。拡張ベクトル真理値の完備束 $\mathcal{T}_e$ は $\mathcal{T}_v \times \mathcal{T}_d$ として定義される。$\mathcal{T}_v$ はベクトル真理値束を表し、$\mathcal{T}_d$ は義務論理の真理値束を表す。束 $\mathcal{T}_d$ の順序関係は $\preceq_d$ の記号で表され、Figure 1 のような形で定義される。束 $\mathcal{T}_d$ の構成要素のそれぞれの意味は次の通りである。真理値 α はそれを持つベクトル真理値付き論理が事実について述べていることを表す。真理値 β はそれを持つベクトル真理値付き論理が義務について述べていることを表す。真理値 γ はそれを持つベクトル真理値付き論理が義務ではない事について述べていることを表す。又、$*_1$ は α, β の最小上界、$*_2$ は β, γ の最小上界、$*_3$ は γ,α の最小上界、$\bot$ は α, β, γ の最大下界、$\top$ は α, β, γ の最小上界をそれぞれ表す。

$\mathcal{T}_d$ の真理値束は、ベクトル真理値付き論理式が義務を表すのか、事実を表すのかを示すために用いられる。$\overrightarrow{\gamma\beta}$ 方向がの義務真理値量を表し、$\overrightarrow{\bot *_2}$ 方向が義務知識の強さを表し、$\overrightarrow{\bot\alpha}$ 方向が事実の強さを表している。それゆえ、例えば真理値 β は真理値 γ よりも義務の観点から見てより真であると解釈される。そして、真理値 $\bot$ と $*_2$ は義務という知識の観点から見ると、中立



ベクトル真理値束　　義務論理の真理値束

Figure 1: 束 $\mathcal{T}_v$ と $\mathcal{T}_d$

であり義務でも非義務でもない。束 $\mathcal{T}_v$ の順序関係は $\preceq$ の記号で表され、拡張ベクトル真理値 $[(i_1, j_1), \mu_1]$、$[(i_2, j_2), \mu_2]$ があった時、次のように定義される。

$$[(i_1, j_1), \mu_1] \preceq [(i_2, j_2), \mu_2] \iff (i_1, j_1) \preceq_v (i_2, j_2) \text{ and } \mu_1 \preceq_d \mu_2.$$

EVALPSN には 2 種類の認識論否定 $\neg_1$ と $\neg_2$ があり、$\mathcal{T}_v$ と $\mathcal{T}_d$ でそれぞれ次のように定義される。

定義 1 (認識論否定、$\neg_1$ 、$\neg_2$)

$$\begin{aligned}
\neg_1([(i,j),\mu]) &= [(j,i),\mu], \quad \forall \mu \in \mathcal{T}_d \\
\neg_2([\lambda,\bot]) &= [\lambda,\bot], & \neg_2([\lambda,\alpha]) &= [\lambda,\alpha], \\
\neg_2([\lambda,\beta]) &= [\lambda,\gamma], & \neg_2([\lambda,\gamma]) &= [\lambda,\beta], \\
\neg_2([\lambda,*_1]) &= [\lambda,*_3], & \neg_2([\lambda,*_2]) &= [\lambda,*_2], \\
\neg_2([\lambda,*_3]) &= [\lambda,*_1], & \neg_2([\lambda,\top]) &= [\lambda,\top],
\end{aligned}$$

λ はベクトル真理値 (i, j) を表す。それらの認識論否定 $\neg_1$,$\neg_2$ は上記の工程を追って除去される。一方、EVALPSN の存在論的否定 (強否定 $\sim$) は、認識論的否定 $\neg_1$,$\neg_2$ で解釈されて定義される。

定義 2 (強否定 (*StrongNegation*))

F を任意の論理式とすると次のように表される

$$\sim F =_{def} F \to ((F \to F) \wedge \neg(F \to F)).$$

ここで、$\neg$ は $\neg_1$, または $\neg_2$ のどちらかである。

定義 3 (well extended vector annotated literal)

p をリテラルとする。$p : [(i, 0), \mu]$ や $p : [(0, j), \mu]$ は *well extended vector annotated literals*(略して weva-literals) と呼ぶ。また、ここでは $i, j \in \{1, 2\}$, and $\mu \in \{\alpha, \beta, \gamma\}$ とする。

定義 4 (EVALPSN)　$L_0, \cdots, L_n$ が weva-literals, ならば

$$L_1 \wedge \cdots \wedge L_i \wedge \sim L_{i+1} \wedge \cdots \wedge \sim L_n \to L_0$$

は強否定を含む拡張ベクトル真理値付き論理プログラム節 *(Extended Vector Annotated Logic Program clause with Strong Negation)* であり、略して EVALPSN 節と呼ばれる。*Extended Vector Annotated Logic Program with Strong Negation* は EVALPSN 節の有限集合である。EVALPSN が強否定を含まないときは EVALP と呼ぶ。　義務概念や事実は拡張ベクトル真理値で次のように表現できる。"事実" は拡張ベクトル真理値 $[(m, 0), \alpha]$ で表す；"義務" は拡張ベクトル真理値 $[(m, 0), \beta]$ で表す；"禁止" は拡張ベクトル真理値 $[(0, m), \beta]$ で表す；and "許可" は拡張ベクトル真理値 $[(0, m), \gamma]$ で表す。$m = 1, 2$ とする。例えば、weva-literals $p : [(2, 0), \alpha]$ and $q : [(0, 1), \beta]$ はそれぞれ、"リテラル p は強さ 2 の事実である事が知られている"、"リテラル q は強さ 1 の禁止である事が知られている" と解釈される。

3 EVALPによる鉄道連動装置安全性検証

3.1 EVALPに基づいた安全性検証

一般的に、安全性検証システムの大部分は検証する時にこのようになれば安全性が保障されるという基準を持っている。そして、今回の安全性検証は検証するための入力情報と、安全である基準の間に矛盾が存在するかを調べることで行う。安全な基準は本論文では *safety properties for railway interlocking*(鉄道連動装置安全性検証) と呼ぶ。例えば、「2 台以上の列車が同時に同じトラックをロックする事は禁止である」のような *safety properties* を用いた文をは義務論理を用いて表現できるので、EVALP を使って表現できる。　一方で、検証する入力情報は許可のリクエストとして見なす。例えば「トラック T_0 をロックする事を許可できるか?」というように表現する。EVALP は義務概念を扱い、検証する安全特性と入力情報 (リクエスト) は EVALP に変換できる。

鉄道連動装置における安全性検証システムでは、基本的にパネルルートリクエスト (PRR) とサブルートリリース (SRR) という 2 つのオペレータからの要求がある。連動装置の安全性を検証する方法として、進路確保要求、区分内進路開放要求が安全性を表現する規則に反しないかどうかを判定することによって行う。PRR はルートをロック (確保) するための要求であり、SRR はロックしたルートを開放するための要求である。これらは一般的に次のような形をしている。

$$\texttt{Qx} \quad \texttt{if} \quad A_1, \ldots, A_m \quad \texttt{then} \quad B_1, \ldots, B_n \backslash .$$

これらを EVALP で検証する手順は後述する。

3.2 鉄道における基本用語

鉄道連動装置とは、駅構内の信号機やポイント (転轍機) の制御を行う装置の事である。まず、Figure 2 で表されるような簡単な路線図を想定する。Figure 2 の路線上にある物理的及び論理的な制御の状態は次のように表現される。

$$\text{track sections} \quad \mathcal{T} = \{T_0, T_1, \ldots, T_7\},$$
$$\text{points} \quad \mathcal{P} = \{P_1, P_2, P_3\}.$$

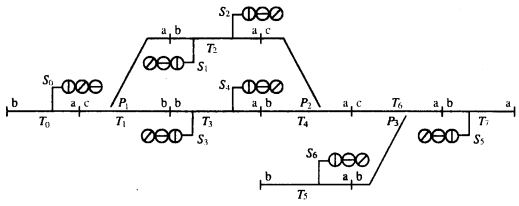


Figure 2: Signaling Schema Example

$$\begin{aligned}&\text{signals} \quad \mathcal{S} = \{S_0, S_1, \ldots, S_6\},\\ &\text{routes} \quad \mathcal{R} = \{R_{02}, \ldots, R_{53}, R_6\},\\ &\text{sub-routes} \quad \mathcal{U} = \{T_0^{ab}, T_0^{ba}, T_1^{ca}, \ldots, T_7^{ba}\}.\end{aligned}$$

例えば、サブルート T_0^{ab} はトラック T_0 が a から b に向かって進むことを意味する。ルート R_{02} は信号機 S_0 から S_2 までのルートを示し、サブルート $T_0^{ba}, T_1^{ca}, T_2^{ba}$ から構成されている事を意味する。各々の実体は次に述べるような論理的、物理的状態から成る。

サブルートは2つの状態 `locked(l)` と `free(f)` から成る。`locked` は、「サブルートがロックされた」という事を表し、サブルートが列車によってオキュパイ (占有) された事を意味する。`free` はロックされていない事を意味する。T_0^{ba} `1` はトラック T_0 が ba 方向にロックされたことを示す。

ルートは2つの状態 `set(s)` と `unset(xs)` から成る。`set` は「ルートがセットされた」という事を表し、そのルートの全てのサブルートがロックされた事を意味する。`unset` はサブルートがセットされていない状態である。　トラックは `occupy(o)` と `cleared(c)` から成る。`occupy` は「トラックがオキュパイされた」という事を表し、列車がそのトラックに存在することを示す。`cleared` は列車が既に通過してしまったことを示す。

ポイントは4つの状態から成る。ポイント P_1 を元にして説明する。`controlled normal(cn)` は P1 が順方向である事を示す。つまり、P_1 `cn` は P_1 があるトラック T_1 を列車は cb か bc 方向に進むという事である。`controlled reverse(cr)` は P_1 が逆方向である事を示す。P_1 `cr` は P_1 上のトラック T_1 を列車は ca か ac 方向に進むという事である。`controlled normal or free to move(cnf)` は順方向になっているか、逆方向に鎖錠されていても、P_1 `cnf` 順方向が $free$ であるならば P_1 は順方向へ動かせるという意味である。`controlled reverse or free to move(crf)` も同じ意味である。

3.1 節で述べたルートリクエストによる表記で表現すると、　PRR の集合 $\mathcal{Q}_{PRR}$ は次のように宣言される。例えば、次の PRR`Q02` はルート R_{02} を確保するための PRR である。

$\mathcal{Q}_{PRR} = \{\texttt{Q02}, \texttt{Q04}, \ldots, \texttt{Q6}, \ldots\}$.

$$\begin{array}{lll}\texttt{Q02} & \text{if} & P_1\texttt{crf},\ T_1^{ac}\texttt{f},\ T_2^{ab}\texttt{f}\\ & \text{then} & R_{02}\texttt{s},\ P_1\ \texttt{cr},\ T_1^{ca}\texttt{1},\ T_2^{ba}\texttt{1}\end{array}$$

SRR の集合 $\mathcal{Q}_{SRR}$ は次のように宣言される。例えば、次の SRR`SR02` はルート R_{02} を開放するための SRR である。

$\mathcal{Q}_{SRR} = \{\texttt{SR02}, \texttt{SR04}, \ldots, \texttt{SR6}, \ldots\}$.

$$\begin{array}{lll}\texttt{SR02} & \text{if} & R_{02}\texttt{xs},\ T_1\texttt{c},\ T_2\texttt{c}\\ & \text{then} & T_1^{ca}\texttt{f},\ T_2^{ba}\texttt{f}\end{array}$$

安全性検証における範囲を駅構内から路線全体に拡張した。そのため、安全性検証は即時性を持ち、常に行う必要がある。

SRR においては列車が現在いるトラックを走り終わった後に導出させて、トラック上のサブルートを開放させる。

これらについての安全性検証の方法は後述する。

3.3 EVALP の安全特性

安全特性 *MX, RT, PT* [4] は EVALP で表現できる。

MX　同一トラック内に属するサブルートのうち、2つ以上のサブルートが同時にロックされることは禁止である。

RT　ルートの構成要素の全てのサブルートがロックされれば、そのルートはセットされるポイントを含むサブルートがロックされたならば、トラック中のポイントは対応するサブルートの方向に鎖錠されなければならない。

PT　ポイントを含むサブルートがロックされたならば、トラック中のポイントは対応するサブルートの方向に鎖錠されなければならない。

各状態を表現する記号が次のようにベクトル真理値 (2次元ベクトル) の代わりに拡張ベクトル真理値として使われる：$\{\texttt{l}, \texttt{f}, \texttt{s}, \texttt{xs}, \texttt{cn}, \texttt{cnf}, \texttt{cr}, \texttt{crf}, \texttt{o}, \texttt{c}\}$,

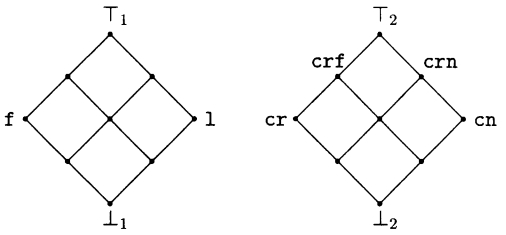
それから、拡張ベクトル真理値で定義される認識論否定 $\neg_1$ は次のように定義される。

$$\begin{aligned}\neg_1(\bot_i) &= \bot_i, & \neg_1(\top_i) &= \top_i,\\ \neg_1([\texttt{l}, \mu]) &= [\texttt{f}, \mu], & \neg_1([\texttt{f}, \mu]) &= [\texttt{l}, \mu],\\ \neg_1([\texttt{s}, \mu]) &= [\texttt{xs}, \mu], & \neg_1([\texttt{xs}, \mu]) &= [\texttt{s}, \mu],\\ \neg_1([\texttt{cn}, \mu]) &= [\texttt{cr}, \mu], & \neg_1([\texttt{cr}, \mu]) &= [\texttt{cn}, \mu],\\ \neg_1([\texttt{cnf}, \mu]) &= [\texttt{crf}, \mu], & \neg_1([\texttt{crf}, \mu]) &= [\texttt{cnf}, \mu],\\ \neg_1([\texttt{o}, \mu]) &= [\texttt{c}, \mu], & \neg_1([\texttt{c}, \mu]) &= [\texttt{o}, \mu],\end{aligned}$$

ここで、$\mu \in \mathcal{T}_d$ で、i=1,2,3,4 となる。この束構造をサブルートに関する構造と、ポイントに関する構造を Figure 3 に挙げる。トラックとルートに関する構造はサブルートと同じ構造となる。

EVALP 節でどのように鉄道の状態を表現するかを述べる。例えば、$T(0, ab) : [\texttt{f}, \alpha] \to T(0, ba) : [\texttt{f}, \gamma]$ は "サブルート T_0^{ab} が `free` であることが事実ならば、サブルート T_0^{ba} が `lock` される事を許可する" となる。

サブルートへの安全特性 *MX* は同じトラック内で2台以上の列車からのロックを禁止する事を意味する。



サブルートの真理値束　　ポイントの真理値束

Figure 3: 鉄道状態における真理値束

それゆえ、次の条件 $MX[T_0^{ab}, T_0^{ba}]$ は“サブルート T_0^{ab} と T_0^{ba} のどちらかが free ならば、別のサブルートはロックを許可する”ことを示し、EVALP 節で次のように変換される。

$$T(0,ab):[\mathtt{f},\alpha] \rightarrow T(0,ba):[\mathtt{f},\gamma], \quad (1)$$
$$T(0,ba):[\mathtt{f},\alpha] \rightarrow T(0,ab):[\mathtt{f},\gamma]. \quad (2)$$

同様に条件 $MX[T_2^{ab}, T_2^{ba}]$ は EVALP 節に変換される。

$$T(2,ab):[\mathtt{f},\alpha] \rightarrow T(2,ba):[\mathtt{f},\gamma], \quad (3)$$
$$T(2,ba):[\mathtt{f},\alpha] \rightarrow T(2,ab):[\mathtt{f},\gamma], \quad (4)$$

トラック T_1 はポイント P_1 を含む、その時の条件は、$MX[T_1^{ac}, T_1^{ca}, T_1^{bc}, T_1^{cb}]$ となり、“順方向 (逆方向) 側のサブルート T_1^{bc}、T_1^{cb} (T_1^{ac}、T_1^{ca}) のどちらかが free で、ポイント P_1 が順方向 (逆方向) に制御されることを許可するなら、順方向 (逆方向) 側のサブルートの残った方はロックを許可する”と解釈する。そして、EVALP 節で変換する。

$$T(1,cb):[\mathtt{f},\alpha] \wedge P(1):[\mathtt{cr},\gamma] \rightarrow T(1,bc):[\mathtt{f},\gamma], \quad (5)$$
$$T(1,bc):[\mathtt{f},\alpha] \wedge P(1):[\mathtt{cr},\gamma] \rightarrow T(1,cb):[\mathtt{f},\gamma], \quad (6)$$
$$T(1,ca):[\mathtt{f},\alpha] \wedge P(1):[\mathtt{cn},\gamma] \rightarrow T(1,ac):[\mathtt{f},\gamma], \quad (7)$$
$$T(1,ac):[\mathtt{f},\alpha] \wedge P(1):[\mathtt{cn},\gamma] \rightarrow T(1,ca):[\mathtt{f},\gamma]. \quad (8)$$

安全特性 RT はあるルート上の全てのサブルートがロックされることを許可するなら、そのルートがセットされる事を許可する事を意味する。条件 $RT(R_{02}, [T_1^{ca}, T_2^{ba}])$ は“サブルート T_1^{ca} と T_2^{ba} の両方がロックを許可されるなら、ルート R_{02} はセットを許可する”となり、EVALP 節で次のように変換する。

$$T(1,ca):[\mathtt{f},\gamma] \wedge T(2,ba):[\mathtt{f},\gamma] \rightarrow R(02):[\mathtt{xs},\gamma]. \quad (9)$$

安全特性 PT はポイント制御とサブルートの連動の関係を表現している。条件式 PT $\mathtt{cn}(P_1, [T_1^{bc}, T_1^{cb}])$、$PT$ $\mathtt{cr}(P_1, [T_1^{ac}, T_1^{ca}])$ は“順方向 (逆方向) 側のサブルート T_1^{bc},T_1^{cb} (T_1^{ac},T_1^{ca}) が free で、ポイント P_1 が controlled normal (reverse) か、free to move ならば、ポイント $P1$ は順方向 (逆方向) に制御することが許可される”となり、EVALP 節で次のように表現できる。

$$T(1,bc):[\mathtt{f},\alpha] \wedge P(1):[\mathtt{cnf},\alpha] \rightarrow P(1):[\mathtt{cr},\gamma], \quad (10)$$
$$T(1,cb):[\mathtt{f},\alpha] \wedge P(1):[\mathtt{cnf},\alpha] \rightarrow P(1):[\mathtt{cr},\gamma], \quad (11)$$
$$T(1,ac):[\mathtt{f},\alpha] \wedge P(1):[\mathtt{crf},\alpha] \rightarrow P(1):[\mathtt{cn},\gamma], \quad (12)$$
$$T(1,ca):[\mathtt{f},\alpha] \wedge P(1):[\mathtt{crf},\alpha] \rightarrow P(1):[\mathtt{cn},\gamma]. \quad (13)$$

次にサブルートリリース (SRR) におけるルートのアンセット条件を考える。SRR についても、EVALP 節を用いて、表現できる。SRRs は列車が通った後のロックされたトラックを順に開放し、ルートをアンセット状態にしていく。

例えば、サブルート T_0^{ab} がリリース (開放) されるための条件を考えてみる。そのためには、サブルート T_1^{ac} と T_1^{bc} が free 状態で、トラック T_0 が列車のいない cleared 状態である必要がある。それによって、T_0^{ab} が free になる事が許可される。これを EVALP で表すと、次のようになる。

$$T(1,ac):[1,\gamma] \wedge T(1,bc):[1,\gamma] \wedge T(1):[\mathtt{c},\alpha] \rightarrow T(0,ab):[1,\gamma], \quad (14)$$

次に異なるルートに対して、共通したサブルートを含んでおり、そのサブルートをリリースする場合を考える。リリースするためには、そのサブルートを含む全てのルートがアンセットされ、トラックがクリアされて列車がそのトラック上に存在しない必要がある。そのような状態であれば、目的とする共通のサブルートがフリーになることが許可される。または、全ての共通するサブルートの後にあるサブルートがフリーで、トラックに列車がいないクリアの状態ならば、共通するルートにあるサブルートはフリーである事が許可される。例を挙げて説明する。R_{53} と R_{51} が両方ともアンセット状態でトラック T_6 がクリアされているならば、サブルート T_6^{ac} はフリーを許可する。

これを EVALP で表現すると次のようになる。

$$R(53):[\mathtt{xs},\alpha] \wedge R(51):[\mathtt{xs},\alpha] \wedge T(6):[\mathtt{c},\alpha] \rightarrow T(6,ac):[1,\gamma], \quad (15)$$

次節の例題で SRR SR02 について検証する。そこで R_{02} をリリースするために必要な論理式を同様 (14),(15) に EVALP を用いて定義する。

$$T(1, ca):[1, \gamma] \wedge T(2):[\mathtt{c}, \alpha]$$
$$\rightarrow T(2, ba):[1, \gamma], \quad (16)$$
$$R(02):[\mathtt{xs}, \alpha] \wedge T(1):[\mathtt{c}, \alpha]$$
$$\rightarrow T(1, ca):[1, \gamma]. \quad (17)$$

これらを用いて次節で SRR の安全性検証を行う。

3.4 例題

例題を用いて、EVALP を用いた安全性検証を行う。3.1 節で示したように、PRR,SRR は次のような形をしている。

***Q** **if** $A_1, \ldots, A_m$ **then** $B_1, \ldots, B_n \backslash$.

PRR,SRR を EVALP を用いて検証するための基本的な考え方とは、3.3 節で説明したような安全特性は、事実、義務、禁止、許可といった記述を含む規則としてみなすことができる。全ての要求は **if** 部分（条件部）と **then** 部分（結論部）から成っている。そして、条件部が仮定された時に、結論部が安全特性に反しないのならば、その要求の安全性は保証されたと言える。つまり、条件部が事実として与えられ、結論部が安全特性に対して許可されるならば、その要求の安全性は保証されたと解釈することができる。従って、これらの要求を EVALP によって表現し、EVALP プログラミングシステムの中で、線路図の安全特性に対してチェックをすることにより、与えられた要求の安全性を検証することができるということである。次に、線路図の安全特性に対する、要求の安全性のチェックの方法について示す。

まず与えられた線路図で必要とされる安全特性を EVALP EP_1 に変換する。全ての条件部 A_i $(1 \leq i \leq m)$ を事実として EVALP EP_2 に変換し、結論部 B_j $(1 \leq j \leq n)$ をそれぞれ許可として EVALP 節に変換する。その EVALP 節を、EVALP $EP = EP_1 \cup EP_2$ に対して質問する。そして、**yes** が返ってくれば、要求の安全性は保証されたことになり、**no** が返ってくれば、要求は安全ではないということを意味している。実際の検証例として、PRR Q02 を用いて、EVALP を用いた安全性検証を行う。まず、PRRQ02 に必要とされる安全特性 $\{(1), \ldots, (13)\}$ の EVALP 節を EVALP EP_1 に変換する。次に PRR Q02 の条件部を EVALP 節に変換し、それを事実として EVALP EP_2 に加える。

$$P(1):[\mathtt{crf}, \alpha], \quad (18)$$
$$T(1, ac):[\mathtt{f}, \alpha], \quad (19)$$
$$T(2, ab):[\mathtt{f}, \alpha], \quad (20)$$

結論部となる EVALP 節は

$$T(2, ba):[\mathtt{f}, \gamma], \quad (21)$$
$$P(1):[\mathtt{cn}, \gamma]. \quad (22)$$
$$T(1, ca):[\mathtt{f}, \gamma]. \quad (23)$$

それぞれ $\{(3), (20)\}$, $\{(18), (12), (19)\}$, $\{(8), (19), (22)\}$, から導出される。最後に、ルート R_{02} をセットする事を許可する表現 $R(02):[\mathtt{xs}, \gamma]$ は次の EVALP 節 $\{(9), (21), (23)\}$. から導出される。よって、PRR Q02 の安全性は保証された。

次に、SRR SR02 の安全性を検証する。PRR と同様に、SRR SR02 に必要とされる条件式 (16)、(17) を EP_1 に加える。SRR の条件式を EVALP 節に変換し、それを事実として EVALP EP_2 に加える。条件部となる EVALP 節は

$$R(02):[\mathtt{xs}, \alpha], \quad (24)$$
$$T(1):[\mathtt{c}, \alpha], \quad (25)$$
$$T(2):[\mathtt{c}, \alpha], \quad (26)$$

となる。次に結論部となる EVALP 節は

$$T(1, ca):[1, \gamma], \quad (27)$$
$$T(2, ba):[1, \gamma], \quad (28)$$

となる。この安全性検証は、それぞれ $\{(24), (25), (17)\}$, $\{(26), (27), (16)\}$ から導き出される。これによって、SRR SR02 による安全性は保障され、ルートの開放が成される。

4 安全性検証シミュレータ

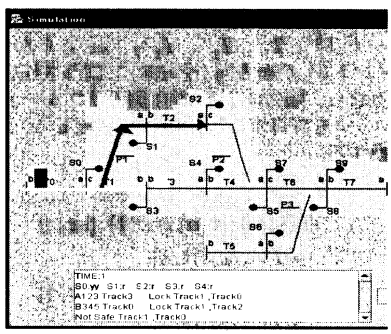
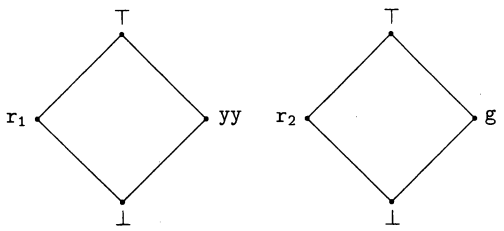


Figure 4: 安全性検証シミュレーション

この章では、Figure4 のような単純な例を用いて、信号制御も含む EVALP を導入した鉄道連動装置安全性検証シミュレータのプロトタイプを紹介する。最初に、Figure4 の 5 つの信号 S0,S1,S2,S3,S4 の制御について説明する。まず Figure3 は駅構内を示していて、そして T_2,T_3 は駅のプラットホームである。信号 S0 は場内信号と呼ばれていて、yy(黄黄：時速 25k m/h に減速する) と $\mathtt{r}_1$((赤：停止) の 2 つの状態を示す。他の 4 つの信号 S1,S2,S3,S4 は出発信号と呼ばれ、g((青：出発) ,$\mathtt{r}_2$(赤：停止) の 2 つの状態を示す。これらの状態も拡張ベクトル真理値を用いて表現できる。その束構造を Figure 5 で述べる。



場内信号の真理値束　　出発信号の真理値束

Figure 5: 信号機の状態の束

この束構造を認識論否定 $\neg_1$ を用いて表すと次のようになる。

$$\neg_1(\bot_j) = \bot_j, \quad \neg_1(\top_j) = \top_j,$$
$$\neg_1(\mathtt{yy}) = \mathtt{r}_1, \quad \neg_1(\mathtt{r}_1) = \mathtt{yy},$$
$$\neg_1(\mathtt{g}) = \mathtt{r}_2, \quad \neg_1(\mathtt{r}_2) = \mathtt{g}.$$

信号制御は EVALPSN 節で形式化されるが、EVALPSN は一階論理プログラムである。そこで強否定 (Strong Negation) は論理式の存在を否定するものとして、論理式の反対の意味を示す否定として扱う。信号機 S0,S1,S2,S3,S4 の論理的な解釈は次のような形になる。

[S0]　ルート R_{02},R_{04} のどちらかがセットされ、トラック T_0 がオキュパイされれば、信号機 S_0 は yy になる。そうでなければ $\mathtt{r}_1$ である。

[S1]　ルート R_1 がセットされ、トラック T_2 がオキュパイされれば、信号機 S1 は g になる。そうでなければ $\mathtt{r}_2$ である。

[S2]　ルート R_{29} がセットされ、トラック T_2 がオキュパイされれば、信号機 S2 は g になる。そうでなければ $\mathtt{r}_2$ である。

[S3]　ルート R_3 がセットされ、トラック T_3 がオキュパイされれば、信号機 S3 は g になる。そうでなければ $\mathtt{r}_2$ である。

[S4]　ルート R_4 がセットされ、トラック T_3 がオキュパイされれば、信号機 S4 は g になる。そうでなければ $\mathtt{r}_2$ である。

$$R(02):[\mathtt{s},\alpha] \rightarrow S(0):[\mathtt{yy},\beta], \quad (29)$$
$$R(04):[\mathtt{s},\alpha] \rightarrow S(0):[\mathtt{yy},\beta], \quad (30)$$
$$\sim S(0):[\mathtt{yy},\beta] \rightarrow S(0):[\mathtt{r}_1,\beta], \quad (31)$$
$$R(1):[\mathtt{s},\alpha] \rightarrow S(1):[\mathtt{g},\beta], \quad (32)$$
$$\sim S(1):[\mathtt{g},\beta] \rightarrow S(1):[\mathtt{r}_2,\beta], \quad (33)$$
$$R(29):[\mathtt{s},\alpha] \rightarrow S(2):[\mathtt{g},\beta], \quad (34)$$
$$\sim S(2):[\mathtt{g},\beta] \rightarrow S(2):[\mathtt{r}_2,\beta], \quad (35)$$
$$R(3):[\mathtt{s},\alpha] \rightarrow S(3):[\mathtt{g},\beta]. \quad (36)$$
$$\sim S(3):[\mathtt{g},\beta] \rightarrow S(3):[\mathtt{r}_2,\beta], \quad (37)$$
$$R(49):[\mathtt{s},\alpha] \rightarrow S(4):[\mathtt{g},\beta], \quad (38)$$
$$\sim S(4):[\mathtt{g},\beta] \rightarrow S(4):[\mathtt{r}_2,\beta]. \quad (39)$$

Figure 4 について説明する：トラック T_0 には列車 $B345$(黒四角) がいて、その列車からのルートリクエストによって、サブルート T_1^{ca}、T_2^{ba} をロックしたルート R_{02} が黒矢印で表示されている。そして、T_3 に別の列車 $A123$(白四角) がいて、その列車からのルートリクエストによって、サブルート T_1^{bc},T_0^{ab} をロックしたルート R_3 が白矢印で表示されている。この状況下で、PRR によって R_3 の安全性を EVALPSN で検証する。見た通り、トラック T_0 は列車 $B345$ にオキュパイされていて、サブルート T_1^{ca} がその列車によってロックされている。これらの事実から、R_3 の PRR との間に衝突が起こる。それゆえ、R_3 の安全性は保障されない。現在の鉄道連動装置の状態と、安全性検証の結果がシミュレータの下のフレームに表示されている。そのフレームの 2 行目に信号機の状態が表示されている。S0 は R_{02} がセットされているので赤 (r_1, r_2) を示していて、他の信号 S0,S1,S2,S3,S4 はルートがセットされていない、列車の安全性が認められない、といった理由から赤になっている。

補足事項　EVALP に基づいた安全性検証について簡単に記述する。本論文では記述していないが、本来のシミュレーションシステムでは、時間の概念を考慮に入れる必要がある。

5　結論及び展望

今回の Dr.Morley によって提案された鉄道連動装置安全性検証の方法は駅構内においてのやりとりであった。我々はそれを EVALP を用いて路線全体を走る列車や信号制御の安全性検証を行えるようにする。そのため、他の列車との衝突を防ぐために、常に安全性検証を行いながら進む即時安全性検証システムを考えた。そして、それが実際に正しい事を実証するために、その EVALP に基づいたシミュレータを作成し、即時安全性検証が正しい事を実証した。

更に、我々は鉄道連動装置安全性検証システムに基づいた EVALP を列車ダイヤグラム復旧システムに使うように発展させていく。そのシステムは今までのようなルートリクエスト、サブルートリリースによる列車の行動制御だけでなく、列車のスケジューリングの検証を行う。そのため、インターバルタイムとして、時間の概念を扱うための論理である時相区間論理が必要となるだろう。

そして、EVALPSN はインターバルタイムを表現するための第二の束を加えることで時相区間論理を扱えるように拡張していく。

現在考えている時相区間論理を加えた EVALP の形としてはトラックを例にした EVALP を用いて考えると

$$T(2, ba) : [\mathfrak{f}, \gamma] : [t_1, t_2]$$

のような形になる。t_1,t_2 が時間の束となる。

このような時間の束を EVALP に拡張する事で、列車の遅れの検出に役立てることができると考える。実際、列車に遅れが生じて、運行に支障が出てしまう場合がある。その時に、実際の鉄道の世界ではその時の鉄道員の判断で、ダイヤの復旧を行う。このような時に、鉄道員のその場の行為だけでは対処できなくなってしまうような時 (どちらに判断したら良いか決めかねる時) に列車のダイヤの大幅な乱れや、運休等が起こってしまい、乗客に対しての混乱が起こってしまう場合があるだろう。

それだけではなく、列車の遅れによって、列車の運転士や車掌が次にいるべきはずの列車にいないといった場合も生じるかもしれない。そうなれば、もちろん列車にいても発車する事は不可能である。

このような時に EVALP を使って全てのあらゆる自体を解決できるようにしておけば、良い。EVALP はどちらに判断したら良いか決めかねるディフィージブル推論のような事象を解決できる手段であり、推論によって、正確な理論により、正しい答えを導き出せる。又、何分遅れで復旧が可能になるかといったことも人間の勘ではない正しい結論を導く事ができるので、このような場合に対して用いる事ができるという利点がある。現在はこのような時相区間論理を含めた EVALP を用いたダイヤグラム検証システムを計画中である。

文献

[1] Blair,H.A. and Subrahmanian,V.S., "Paraconsistent Logic Programming", *Theoretical Computer Science*, Vol. 68, pp.135–154, 1989.

[2] da Costa,N.C.A., Subrahmanian,V.S., and Vago,C., "The Paraconsistent Logics $P\mathcal{T}$", *Zeitschrift für Mathematische Logic und Grundlangen der Mathematik*, Vol. 37, pp.139–148, 1991.

[3] Gordon,M.J.C. and Melham,T.F., *Introduction to HOL*, Cambridge Univ. Press, 1993.

[4] Morley,J.M., "Safety Assurance in Interlocking Design", *Ph.D Thesis*, University of Edinburgh, 1996.

[5] Nakamatsu,K. and Suzuki,A., "Annotated Semantics for Default Reasoning", *Proc. 3rd Pacific Rim International Conference on Artificial Intelligence*, pp.180–186. International Academic Publishers, 1994.

[6] Nakamatsu,K. and Suzuki,A., "A Nonmonotonic ATMS Based on Annotated Logic Programs, Agents and Multi-Agents Systems", LNAI Vol. 1441, pp.79–93, Springer-Verlag, 1998.

[7] Nakamatsu,K. and Abe,J.M., "Reasonings Based on Vector Annotated Logic Programs", *Computational Intelligence for Modelling, Control & Automation*, Concurrent Systems Engineering Series Vol. 55, pp.396–403. IOS Press, 1999.

[8] Nakamatsu,K., Abe,J.M., and Suzuki,A., "Defeasible Reasoning Between Conflicting Agents Based on VALPSN", *Proc. AAAI Workshop Agents' Conflicts*, pp.20–27. AAAI Press, 1999.

[9] Nakamatsu,K., Abe,J.M., and Suzuki,A., "Defeasible Reasoning Based on VALPSN and its Application", *Proc. The Third Australian Commonsense Reasoning Workshop*, pp.114–130, 1999.

[10] Nakamatsu,K., Abe,J.M., and Suzuki,A., "A Defeasible Deontic Reasoning System Based on Annotated Logic Programming", *Proc. the Fourth International Conference on Computing Anticipatory Systems*, AIP Conference Proceedings Vol.573, pp.609–620. American Institute of Physics, 2001.

[11] Nakamatsu,K., Abe,J.M., and Suzuki,A., "Annotated Semantics for Defeasible Deontic Reasoning", *Proc. the Second International Conference on Rough Sets and Current Trends in Computing*, LNAI Vol.2005, pp.432–440, Springer-Verlag, 2001.

[12] Nakamatsu,K., Abe,J.M., and Suzuki,A., "Defeasible Deontic Robot Control Based on Extended Vector Annotated Logic Programming", *Proc. the Fifth International Conference on Computing Anticipatory Systems*, AIP Conference Proceedings Vol.627, pp.490–500, American Institute of Physics, 2002.

[13] Nakamatsu,K., Suito,H., Abe,J.M., and Suzuki,A., "Paraconsistent Logic Program Based Safety Verification for Air Traffic Control", *Proc. 2002 IEEE International Conference on Systems, Man and Cybernetics* (CD-ROM), IEEE, 2002.

[14] Nakamatsu,K., Seno,T., Abe,J.M., and Suzuki,A., "Intelligent Real-time Traffic Signal Control Based on a Paraconsistent Logic Program EVALP", *Proc. the 9th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing*, LNCS Vol.2639, pp.719–723. Springer-Verlag, 2003.

[15] Nakamatsu,K., Mita,Y., Shibata,T., and Abe,J.M., "Defeasible Deontic Action Control Based on Paraconsistent Logic Program and its Hardware Implementation". *Proc. 3rd International Conference on Computational Intelligence for Modelling Control and Automation* (CD-ROM), 2003.