車載ソフトウェアに対するレンジ解析手法の提案

石井 良尚[1,a)]　沓名 拓郎[1,b)]

概要：車載ソフトウェアの開発効率化を狙い，モデルベース開発が導入されてきている．車載 ECU では計算リソースが限られているため，開発途中で浮動小数点数型の変数が含まれるモデルを，固定小数点数型のモデルに設計し直す必要がある．固定小数点の設計は多大な労力を要するが，レンジ解析手法を用いモデル中の変数の取りうるレンジを自動的に求め，必要なビット長を導出することで効率化できる．しかし，既存のレンジ解析手法の多くは，変数間に相関がある場合，精度が低下してしまう．一方，高精度な手法も存在するが，計算コストの面で課題がある．そこで，本論文では精度と計算コストの両立を狙い，モデル中で変数間に相関が生じる部分を特定し，その部分にのみ高精度なレンジ解析手法を適用する手法を提案する．また，評価実験により本提案手法の有用性を示す．

キーワード：車載ソフトウェア，レンジ解析，固定小数点

Range Analysis Method for In-vehicle Software

Yoshinao Ishii[1,a)]　Takuro Kutsuna[1,b)]

Abstract:
Model-based development has been introduced for efficient development of in-vehicle software. It is necessary to convert floating point variables into fixed-point in developing models, because resouces of in-vehicle ECUs are limited. However, this task requires great effort to developers. It can be automated by calculating the necessary bit-length based on variable ranges that result from using range analysis methods. However, a precision of most range analysis method degrade when there is correlation between variables. In contrast, highly precise methods have a drawback in calculation cost. In this paper, we propose a method that finds the parts which produce correlation between variables in models, and apply highly precise methods to these parts, to achieve a good balance between high precision and low cost. Furthermore, we show the usability of this proposed method by evaluation experiments.

Keywords: in-vehicle software, range analysis, fixed-point

1. はじめに

近年，自動車機器分野では，車載ソフトウェアの開発効率化を狙い，MATLAB/Simulink[*1]によるモデルベース開発の導入が積極的に進められている．モデルベース開発では，Simulink モデルは自動的に実装コードへと変換される．車載 ECU（Electronic Control Unit）では使用できる計算リソースが限られているため，変換により得られる実装コード中の変数は，計算コストの低い固定小数点数型でなければならないことが多い．一方で，開発の上流工程における仕様モデルは，ほとんどの場合，計算コストが高くメモリ消費量も多い浮動小数点数型で設計されている．浮動小数点数型で設計された Simulimk モデルを実装コードに変換すると，得られるコード中の変数も浮動小数点数型となってしまう．そのため，何れかの開発フェーズで，浮動小数点数型で設計されたモデルを，固定小数点数型のモデルに設計し直す必要がある．この固定小数点設計を人手で行う場合，演算によるオーバーフローが起こらないようモデル中の各信号の取りうるレンジを考慮し，必要最小限のメモリ消費量となるビット長を設定しなければならず，多大な労力がかかる．

こうした背景から，MATLAB/Simulink における固定小数点設計を支援する “ Fixed-Point Designer[1] ” や “ TargetLink[2] ” などのツールが存在する．これらのツールを

[1] 株式会社 豊田中央研究所
41-1, Yokomichi, Nagakute, Aichi 480–1192, Japan
[a)] Y-Ishii@mosk.tytlabs.co.jp
[b)] kutsuna@mosk.tytlabs.co.jp
[*1] Mathworks 社の登録商標である．

用いると，モデル中の各信号が取りうるレンジを自動的に求めることができるが，必ずしも厳密なレンジが得られるわけではない．特に，モデル中に非線形演算が存在する場合，実際に信号が取りうるレンジよりも広いレンジが推定されてしまう（これを over-estimation[3] と呼ぶ）．Over-estimation が生じたレンジを基に固定小数点設計を行うと，変数のビット長が必要以上に長くなってしまう．そのため，over-estimation を起こさないように Simulink モデルの信号のレンジを導出できる手法の開発が望まれている．

一方で，C プログラムなどに対しては，Interval Arithmetic[4] を始めとするレンジ解析手法を用いて，プログラム中の変数が取りうる値を解析する手法が研究されてきた [5],[6]．近年では，非線形演算が含まれる場合でも over-estimation をなるべく抑え，高精度にレンジを求めることができるレンジ解析手法が提案されてきている．こうしたレンジ解析手法の多くは，Simulink モデルに対しても適用できる．すなわち，既存の C プログラムなどに対する高精度なレンジ解析手法を利用すれば，Simulink モデル中の各信号のレンジを高精度に求めることが可能である．しかし，高精度なレンジ解析手法は計算コストが高く，大規模な Simulink モデルに適用すると実用時間内に結果が得られないことが多い．

そこで，本論文では，Simulink モデルにおいて over-estimation が生じる可能性がある部分を特定し，その部分に対してのみ高コストなレンジ解析手法を適用する手法を提案する．本提案手法を用いることで，over-estimation を抑えながら高速にレンジ解析を行うことが可能となる．

以下において，本論文の構成について述べる．まず，2 章において既存のレンジ解析手法を紹介し，その問題点について述べる．次に，3 章において本提案手法の詳細について述べる．そして，4 章において，簡単な例題モデルを用い，本提案手法に対する評価実験を行う．

2. レンジ解析手法

本章では，Interval Arithmetic をはじめとする，既存のレンジ解析手法について述べる．

2.1 Interval Arithmetic

レンジ解析のための代表的な手法として，Interval Arithmetic(以降，IA と略記) がある．IA では，ある変数 x の取り得る値の範囲を，下限 x^L と上限 x^U によるレンジ

$$X = [x^L, x^U] = \{x \mid x^L \le x \le x^U\} \tag{1}$$

で表現し，このレンジ同士の演算を定義する．例えば，レンジ $X = [x^L, x^U], Y = [y^L, y^U]$ に対する四則演算は，

$$X + Y = [x^L + y^L, x^U + y^U], \tag{2}$$

$$X - Y = [x^L - y^U, x^U - y^L], \tag{3}$$

$$\begin{aligned} X \times Y &= [\min(S), \max(S)] \\ \text{where } S &= \{x^L \cdot y^L, x^L \cdot y^U, x^U \cdot y^L, x^U \cdot y^U\} \end{aligned} \tag{4}$$

のように定義される [7]（ただし，除算に関しては付録中の式 (A.1) を参照のこと）．例として，$X = [-2, 3], Y = [-1, 4]$ が与えられた場合，$2X - Y$ は，

$$\begin{aligned} 2X - Y &= [2, 2] \times [-2, 3] - [-1, 4] \\ &= [-4, 6] - [-1, 4] \\ &= [-8, 7] \end{aligned} \tag{5}$$

のように計算される．

同様に，ブール型変数 b のレンジは，

$$\begin{aligned} B &= [b^L, b^U] \\ &= \begin{cases} \{0\} & \text{if } b^L = b^U = 0, \\ \{1\} & \text{if } b^L = b^U = 1, \\ \{0, 1\} & \text{if } b^L = 0, b^U = 1. \end{cases} \end{aligned} \tag{6}$$

のように表現される．ただし，b^L, b^U は $[b^L, b^U] \neq [1, 0]$ を満たすブール型変数とする．さらに，レンジ $B_1 = [b_1^L, b_1^U], B_2 = [b_2^L, b_2^U]$ に対する論理演算 and, or, not は，

$$B_1 \text{ and } B_2 = [b_1^L \wedge b_2^L, b_1^U \wedge b_2^U], \tag{7}$$

$$B_1 \text{ or } B_2 = [b_1^L \vee b_2^L, b_1^U \vee b_2^U], \tag{8}$$

$$\text{not } B_1 = [\neg b_1^U, \neg b_1^L]. \tag{9}$$

で定義される [8]．以上が IA の演算方法の定義となる．

IA を Simulimk モデルに適用するには，モデルに対する入力信号のレンジを仮定し，それらをモデル内の信号に伝播させていけばよい．伝播の際には，各種 Simulink ブロックごとに，入力されるレンジを基に出力レンジを計算する必要がある．例えば，Relational Operator ブロックのレンジ演算は，例えば比較演算子が $\le$ の場合，入力レンジを $X = [x^L, x^U], Y = [y^L, y^U]$，出力レンジを B とすると，

$$B = \begin{cases} [0, 0] & \text{if } x^L > y^U, \\ [1, 1] & \text{if } x^U \le y^L, \\ [0, 1] & \text{otherwise} \end{cases} \tag{10}$$

と計算できる．また，Switch ブロックは，分岐条件に利用されるブール型入力のレンジを $B = [b^L, b^U]$，条件が TRUE 時に採用される入力レンジを $X = [x^L, x^U]$,FALSE 時の入力レンジを $Y = [y^L, y^U]$，出力レンジを Z とすると，

$$Z = \begin{cases} [x^L, x^U] & \text{if } b^L = 1, \\ [y^L, y^U] & \text{if } b^U = 0, \\ [\min(x^L, y^L), \max(x^U, y^U)] & \text{otherwise} \end{cases} \tag{11}$$

と計算できる．同様な考え方で，他の Simulink ブロックについてもレンジを伝播できる．

IA はその手法の単純さから，非常に高速にレンジ解析が行えるが，必ずしも厳密なレンジを導出できるわけではな

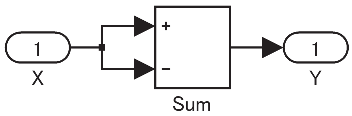

図 1 over-estimation が起こるモデルの例
Fig. 1 over-estimation が起こるモデルの例

い．以下の式 (12) で与えられるレンジ Y の演算を考える．

$$Y = X \times X - X, \quad \text{where } X = [-1, 2]. \tag{12}$$

Y の厳密な解は $[-0.25, 2]$ であるが，IA を用いた場合，

$$\begin{aligned} Y &= X \times X - X \\ &= [-1, 2] \times [-1, 2] - [-1, 2] \\ &= [-2, 4] - [-1, 2] \\ &= [-4, 5] \end{aligned} \tag{13}$$

となり，over-estimation が生じる．これは，IA が変数間の相関性を考慮できないことが原因で起こる．

Simulink モデルにおいては，**図 1** にように，ある信号線が分岐し再び合流する部分（以降，再収斂部分と呼ぶ）において相関性が生じるため，over-estimation が起こり得る．車載ソフトウェアを表す Simulink モデルには，こうした再収斂部分が多く存在する．そのため，車載ソフトに対して IA を用いてレンジ解析を行うと，ほとんどの場合において over-estimation の問題が生じる．

この IA を拡張した手法として，Affine Arithmetic[9] や Taylor Series Methods[10], [11] がある．これらの手法は，変数間にある程度の相関があっても高精度なレンジを導出できるが，以下の問題点を持つ．

1. IA よりも計算コストが高く，また，変数間の相関性が非常に高い場合，レンジ解析の精度が悪化する．
2. モデルがブール型変数を含む場合，適用が難しい．

特に 2 の問題点は，ブール型変数が多く存在する車載ソフトの Simulink モデルに対して適用する上では大きな障害となる．

2.2 SMT ソルバーを用いる手法

変数間の相関性を考慮しながらブール型変数も扱えるレンジ解析手法（以下，高精度なレンジ解析手法と表現する）の 1 つとして，SMT（Satisfiability Modulo Theories problem）ソルバーを用いる手法 [12] が挙げられる．なお，SMT とは制約充足問題の充足可能性を判定する問題のことである．SMT ソルバーを用いる手法は，入力変数にレンジ制約が与えられた非線形物理演算を主な対象としており，こうした演算の出力値が取りうるレンジを高精度に求めることができる．ただし，非線形演算に対応した SMT ソルバーを用いる必要がある．この手法の具体的なレンジ導出法は以下のとおりである．

1. 入力変数のレンジ制約を基に，IA を用いて出力レンジ $Z = [L, U]$ を推定する．
2. 対象とする演算を SMT に変換する．
3. 推定された L の精度をあげるため，以下の処理を行う．
   3-1. $L_1 = L, L_2 = U$ とする．
   3-2. $L' = (L_1 + L_2)/2$ とする．
   3-3. L' より小さい解の有無を SMT ソルバーで判定．解がないと判定された場合，$L_1 = L'$ とする．解があると判定された場合，$L_2 = L'$ とする．
   3-4. $L_2 - L_1$ が閾値 δ 以下になるまで処理 3-1 から 3-4 までを繰り返し，得られた L_2 をレンジの下限として出力する．
4. U に関しても同様の処理を行い，得られた上限を出力する．

上記のように，この手法は，出力が特定の値以下（または、以上）を取り得るかどうか SMT ソルバーに判定させる処理を繰り返すことで，徐々に実際に取り得るレンジに漸近させていく．この処理は，実際に取り得るレンジからの誤差が閾値以内に収まるまで続けられるため，閾値を小さくすれば，ほとんど over-estimation の無いレンジを得ることができる．また，この SMT ソルバーを用いる手法は，非線形演算を扱えるだけでなく，演算中の変数にブール型が存在する場合でも問題なく適用できる．さらに，多くの場合 Simulink モデルは SMT として表現できる [13] ので，この SMT ソルバーを用いる手法は，Simulink モデルのレンジ解析にも適用可能である．

しかし，SMT ソルバーを用いるこの手法は，変換された SMT のサイズに対するスケーラビリティに乏しいという問題点がある．すなわち，Simulink モデルのサイズが大きくなると，変換により得られる SMT のサイズも大きくなり，実用時間内にレンジが得られない可能性がある．

SMT ソルバーを用いる手法以外の高精度なレンジ解析手法として，レンジが取りうる範囲を求めるという問題を最適化問題に変換して解く手法などが存在するが，何れも Simulink モデルのサイズに対するスケーラビリティが乏しいという課題がある [12]．

そこで，次章において，Simulink モデル中で over-estimation の原因となる再収斂部分を特定し，その限られた部分にのみ高精度なレンジ解析手法を適用することで，精度とスケーラビリティを両立する手法を提案する．

3. 提案手法

3.1 再収斂部分の特定

本提案手法では，まずはじめに，Simulink モデル中で over-estimation の原因となる再収斂部分を特定する．再収斂部分の特定は，Simulink モデルをグラフとみなし，グラフの探索を行うことで実現する．このグラフ探索の際には，

高精度なレンジ解析手法の計算コストを抑えるため，特定される再収斂部分の大きさを制限する．なお，Simulink モデルに対応するグラフの作成方法は，文献 [14],[15] などを参照していただきたい．以下において，特定方法の詳細を述べる．

1. Simulink モデルに対応する有向グラフにおいて，2 本以上の枝を出す頂点をすべて抽出する．抽出された n 個の頂点を $v_i(i=1,\cdots,n)$ とする．ただし，$n=0$ の場合，再収斂部分が存在しないとして終了．
2. 深さ優先探索 [16] を用いて，各頂点 $v_i(i=1,\cdots,n)$ からパスの長さが P 以内で到達できる頂点群 $D_i(i=1,\cdots,n)$ を求める．
3. 各 i に対して以下を行う．D_i を無向グラフとみなし，2 重連結成分を求める．求めた各 2 重連結成分の中で，すべての頂点が同 2 重連結成分内の頂点に対して 2 本以上の辺を持つものを，D_i における再収斂部分グラフ $Dp_i(i=1,\cdots,n)$ とする．
4. Dp 中の部分グラフ間において，共通する頂点が存在する場合，それらの部分グラフをマージする．ただし，マージ後の部分グラフの最長有向パスが長さ P を超える場合はマージしない．マージ後に残った各部分グラフに対応する Simulink モデル部分を再収斂部分とする．

ただし，2 重連結成分とは極大な 2 重連結グラフのことを指し，2 重連結グラフとは任意の 1 つの頂点を除いた場合でも連結であるグラフのことを指す．2 重連結成分の導出方法は，文献 [16] などを参照していただきたい．また，有向グラフ作成の際には，UnitDelay ブロックをはじめとする，メモリー要素を持ち，同ステップ中に入力信号と出力信号間で情報のやり取りを行わないブロック（以下，内部状態ブロックと表現する）は，入力信号線が切断されているものと見なす必要がある．

次に，手順 4. においてマージを行う理由を以下で述べる．手順 3. において求めた各部分グラフ間に共通する頂点があるということは，その共通する頂点によって部分グラフ間に相関が生じていることになる．これら部分グラフをマージすることで部分グラフ間の相関性も考慮できるようになり，高精度なレンジが得られやすくなる．この際，マージ後の部分グラフの最長有向パスが P を超える場合にマージしない理由は，高精度なレンジ解析手法によってレンジ解析を行うモデルサイズを制限することで，スケーラビリティを確保するためである．

以下において，**図 2** を用い，2 重連結成分を用いて再収斂部分を特定する過程を例示する．今，$P=6$ として手順 1,2 で得られた d_i において，ある無向グラフが**図 2(a)** であったとする．このグラフの 2 重連結成分は，**図 2(b)** の細線で囲んだそれぞれの部分となる．この各 2 重連結成分の中で，すべての頂点が同 2 重連結成分内の頂点に対して 2 本以上の辺を持つものは，**図 2(c)** の細線で囲った 2 つの部分グラフのみであり，これが Dp_i となる．今回，この他に Dp_i が存在しなかったと仮定し，手順 4 に移る．今回，得られた 2 つの部分グラフ間に共通する頂点が 1 つ存在する．この場合，マージしても最大有向パス長は P 以下なのでマージを行い**図 2(d)** となり，図 2(d) の細線で囲った 1 つの部分グラフが再収斂部分として求まる．

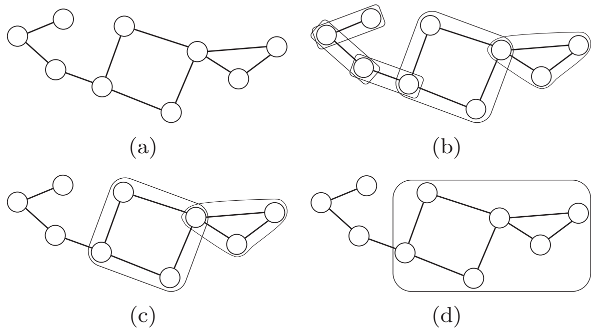

図 2 再収斂部分特定の過程
Fig. 2 A process to specify a re-convergence part

3.2 レンジの伝播方法

Over-estimation が起こらない再収斂部分以外の部分では，モデル最上位の Inport ブロックや Constant ブロックなどのソースブロックを起点として，IA を用いて高速にレンジを伝播させる．ただし，再収斂部分に関しては over-estimation を避けるため，高精度なレンジ解析手法を用いる．各再収斂部分に対して，その再収斂部分に対するすべての入力のレンジが確定した時点で高精度なレンジ解析手法を適用し，その再収斂部分の出力レンジを求める．得られた再収斂部分の出力レンジを基に，再び IA を用いてレンジを伝播させる．ただし，内部状態ブロックに到達した場合は，伝播をやめる．これを伝播が終わるまで繰り返すことで，周期実行されるモデルの 1 ステップ分の信号のレンジが得られる．ただし，用いる高精度なレンジ解析手法によっては，ある再収斂部分の出力レンジを計算する際に，その再収斂部分の出力以外のレンジが得られないことがあり，SMT ソルバーを用いる手法はこれに含まれる．もし，その部分のレンジが必要な場合は，IA などを用いて別途計算する必要がある．

3.3 内部状態ブロックが存在するモデルに対するレンジ解析方法

ほとんどの車載モデルは周期実行され，カウンタやフィードバックループなどを構成する目的で，内部状態ブロックが用いられている．内部状態ブロックが存在する場合，ステップを跨いでレンジが発展していく可能性がある．しかし，3.1 節と 3.2 節の処理を 1 度行うだけでは，モデルの初期状態から 1 ステップ分のレンジしか導出できない．そ

こで，内部状態ブロックが存在する場合でも高精度にレンジを導出するために，以下の処理を行う．

1. モデルの初期条件を基にして，3.2 節の処理を行い，1 ステップ分のレンジを求める．
2. 得られた 1 ステップ分のレンジのうち，内部状態ブロックへの入力レンジを，次ステップにおける内部状態ブロックの出力レンジに設定する．
3. 手順 2 で設定された内部状態ブロックの出力レンジを基にして，3.2 節の処理を行い，次ステップのレンジを求める．
4. 手順 2, 3 を，すべての内部状態ブロックのレンジの変化が収束する，あるいは指定されたステップ数 K まで繰り返す．詳細は以下で述べる．

まず，手順 4 におけるレンジの収束判定方法について述べる．上記処理において計算される n 個の各内部状態ブロック $S_i(i=1,\cdots,n)$ における k ステップ目（ただし，$k \leq K$）の出力レンジを $R_i^k(i=1,\cdots,n)$ とする．このとき，すべての i において

$$R_i^k = R_i^{k-1} \tag{14}$$

が成立した場合，全信号のレンジ計算が収束したと言えるため，この時点で計算を終了する．ここで，収束したステップ数を K' とし，各ステップ k におけるモデル内の各信号 j のレンジを $R_j^k(j=1,\cdots,N,\ k=1,\cdots,K')$ とすると，各信号が取りうる値のレンジ R_j は

$$\begin{aligned} R_j &= [R_j^L, R_j^U] \\ &= [\min_k({R_j^k}^L),\ \max_k({R_j^k}^U)] \end{aligned} \tag{15}$$

によって導出することができる．ただし，N はモデル内における全信号の個数を表す．

次に，K ステップ目までに，すべての i において式 (14) が成立しなかった場合におけるレンジの計算方法について述べる．K ステップ目において，式 (14) が成立しない S_i の集合を，S_p とする．ここで，Simulink モデルに対する有向グラフを考える．ただし，3.1 節のときと違い，内部状態ブロックの入力信号線は接続されているものとする．この有向グラフにおいて，S_p から到達可能な S_i の集合を S_r，S_r に含まれない S_i の集合を S_u とする．このとき，S_r に含まれる内部状態ブロックの出力レンジは，収束していないと考えられる．そこで，S_r の各出力レンジ R_r をすべて $(-\infty,\infty)$（ブール型の場合は [0,1]），S_u の各出力レンジ R_u は収束したレンジに設定した上で 3.2 節の処理を行い，得られたレンジを出力することとする．これにより，得られるレンジが実際のレンジよりも狭くなることを回避する．なお，有向グラフにおける到達可能性の導出方法は，文献 [16],[17] などを参照していただきたい．

上記処理により，周期実行されるモデルにおいて，内部状態ブロックによりステップを跨いでレンジが発展していく場合でも，高精度にレンジ解析を行うことができる．さらに，各信号ごとにおける K ステップ目までのステップごとの高精度なレンジの推移も同時に得ることができるため，固定小数点設計の際の大きな助けになると考えられる．

次章において，本章で提案した手法に対する評価実験を行う．

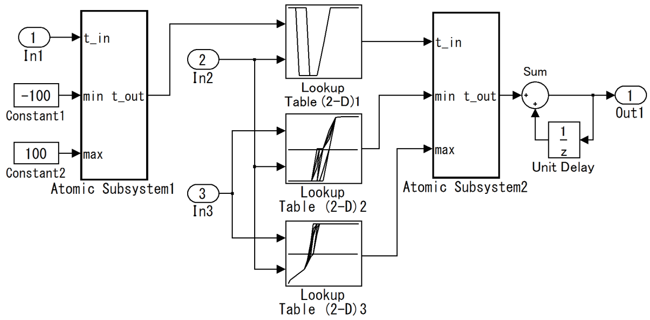

図 3　比較実験用の例題モデル

Fig. 3　The example model for comparison experiments

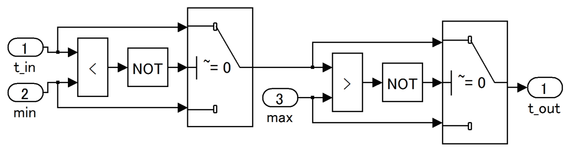

図 4　例題モデル中の Atomic Subsystem の内部構造

Fig. 4　The internal structure of Atomic Subsystems in the example model

4. 評価実験

本章では，2 章で述べた Interval Arithmetic，SMT ソルバーを用いる手法と，3 章で述べた提案手法の精度と計算コストを比較する実験を行う．

4.1 実験方法

今回の実験には，**図 3** に示す例題モデルを用いる．ただし，図 3 中の 2 つの Atomic Subsystem の内部は，共に**図 4** のような構造をしている．この例題モデルは，3 つの 2-D Lookup Table ブロック（テーブルのサイズは 15×10 が 1 つ，10×8 が 2 つで，補間方法はすべて外挿）と，Switch, Logic, Relational Operator ブロックを用いて信号の値に上下限制約を与えるロジック（図 4）により構成されており，固定ステップで周期実行される．そして，入力を部分的に共有する 3 つの 2-D Lookup Table の出力値によって得られる値の積算値を出力する．このような再収斂部分を持つモデル構造は，車載制御モデル中で頻繁に用いられる．

このモデルに対して各手法を適用し，K ステップ目におけるモデルの出力レンジ（図 3 の Out1 に入力される信号のレンジ）を求め，得られたレンジにおける正解値からの誤差と，レンジ導出にかかった計算時間で比較評価を行う．一方で，このモデルの内部状態ブロックの出力レンジ

表 1 既存手法と提案手法における正解値からの誤差の比較
Table 1 Comparison between existing and proposed method by error from correct values

手法	誤差 ϵ				
	K=1	K=2	K=3	K=5	K=10
IA	4.11×10^{1}	8.22×10^{1}	1.23×10^{2}	2.06×10^{2}	4.11×10^{2}
SMT	5.42×10^{-3}	N/A	N/A	N/A	N/A
提案手法	1.01×10^{-2}	2.02×10^{-2}	3.03×10^{-2}	5.06×10^{-2}	1.01×10^{-1}

表 2 レンジ解析に要した計算時間の比較
Table 2 Comparison of the calculation time required for range analysis

手法	計算時間 [s]				
	K=1	K=2	K=3	K=5	K=10
IA	1.19×10^{-2}	2.31×10^{-2}	3.36×10^{-2}	5.77×10^{-2}	1.12×10^{-1}
SMT	1.53×10^{2}	T.O.	T.O.	T.O.	T.O.
提案手法	2.62×10^{1}	5.22×10^{1}	7.82×10^{1}	1.30×10^{2}	2.61×10^{2}

は収束しないため，提案手法を適用した結果のレンジは $(-\infty,\infty)$ となってしまうことは明らかである．そのため，今回の実験においては収束判定を行わず，K ステップ目のレンジをそのまま出力することとする．ただし，誤差 ϵ は，正解値 $Z_e=[z_e^L,z_e^U]$，得られたレンジ $Z=[z^L,z^U]$ を用い，

$$\epsilon=|z_e^L-z^L|+|z^U-z_e^U| \tag{16}$$

で計算する．また，入力のレンジは全手法で共通とし，In1 に関しては In1= $[-256,255]$，In2, In3 に関しては各 2-D Lookup Table ブロックのテーブルデータを元にしたレンジを与えることとする．なお，IA における内部状態ブロックへの対応方法は 3.3 節と同様の方法を用い，SMT ソルバーを用いる手法における SMT ソルバーは“ Z3 ” [18] を用い，閾値 $\delta=10^{-2}$ とした．また，提案手法のパラメータは $P=10$ とし，高精度なレンジ解析手法として，SMT ソルバーを用いる手法（設定は同上）を利用した．なお，実験に使用したマシンのスペックは Windows 7，Core i7 3.47GHz CPU，24GB Memory である．

4.2 実験結果

$K=\{1,2,3,5,10\}$ のそれぞれに対して各手法を適用した結果を**表 1**, **表 2** に示す．ただし，各表における“ SMT ”は SMT ソルバーを用いる手法を意味し，表 2 における T.O. は，10,000 秒以内に結果が得られずタイムアウトさせたことを意味する．なお，提案手法において得られた再収斂部分は 2 箇所で，1 つは図 3 の In1 に対して上下限制約を与える左側の Atomic Subsystem 内のモデル部分，もう 1 つは 3 つの 2-D Lookup Table ブロックと右側の Atomic Subsystem 内のブロックを合わせたモデル部分となった．また，この再収斂部分を求めるためにかかった計算時間は，表 2 中の値に含まれている．

今回の実験結果より，以下のことがいえる．

1. 提案手法は IA よりも高精度なレンジを求めることができる．
2. 計算時間の面において，SMT ソルバーを用いる手法はステップ数 K に対するスケーラビリティが低いが，提案手法は線形時間に収まる．
3. 提案手法の誤差は，K が大きくなるにつれ徐々に増えていく．

結果 1 のようになった理由は，変数間の相関性が考慮できない IA に対して，SMT ソルバーを用いる手法を利用した提案手法は，各再収斂部分における変数間の相関性を考慮することができたからだと言える．なお，IA における誤差の生じ方は，以下の通りである．まず，左側の Atomic Subsystem 内の相関性が考慮できず，その出力レンジは，上下限制約による正解値 $[-100,100]$ よりも広い $[-256,255]$ が得られてしまい，これにより，一番上のテーブルの出力レンジも正解値より広くなる．さらに，右側の Atomic Subsystem 内の上下限制約も考慮できず，その出力レンジは，正解値よりも広めのレンジとなる．この結果が積算されて誤差が積み重なっていく．

また，結果 2 に関して，SMT ソルバーを用いる手法がステップ数 K に対してスケーラビリティが低くなった理由は，この手法を用いる場合，K ステップ分のモデル情報をすべて SMT に変換する必要があるため，K が増えると SMT のサイズが大きくなり，ソルバーの計算時間が指数的に増えてしまうためであると考えられる．同様に，Simulink モデルのサイズを大きくした場合においても SMT のサイズが大きくなるため，SMT ソルバーを用いる手法はモデルサイズに対するスケーラビリティも低いと考えられる．一方で，SMT ソルバーを用いる手法を利用しているにも関わらず，提案手法の計算時間が K に対して線形になっている理

由は，3.3 節で述べたように各ステップごとに分割してレンジを伝播させているため，SMT のサイズが K にほぼ依存せず，SMT を解く回数のみが K に比例して増えるためであると考えられる．加えて，提案手法は，パラメータ P によってサイズが限定された再収斂部分のみに高精度なレンジ解析手法を適用するため，モデルサイズが大きくなった場合でもスケーラビリティを確保できるという利点があると言える．また，高いスケーラビリティにより，大きい K に設定しても短時間でレンジを導出できるため，収束した高精度なレンジを得やすいという利点もあると言える．

これに対して，結果 3 のようになった理由は，提案手法が内部状態ブロックに関して各ステップごとに分割してレンジを伝播させている関係上，内部状態ブロックにおいて生じる相関性を考慮できず，積分器を模すモデル部分などにおいて，K が増えるにつれ誤差が積み重なっていくためであると考えられる．もし，SMT ソルバーを用いる手法の結果が $K > 1$ でも得られたとすると，提案手法のように誤差が K に依存せず，常に $2\delta = 2 \times 10^{-2}$ 内に収まる結果となっていたはずである．また，提案手法では，再収斂部分のサイズに制約をかけている関係上，そのサイズより大きな再収斂部分によって生じる変数間の相関性を考慮できないため，こうした部分で over-estimation が生じる可能性がある．ただし，今回の実験結果において，提案手法は他の比較手法と違い，精度とスケーラビリティを高水準で両立できており，十分に実用的であると言える．

5. おわりに

本論文では，既存のレンジ解析手法の課題である over-estimation と計算コストの高さを解決するために，新たな手法を提案した．本提案手法は，over-estimation の原因となる再収斂部分を特定し，その部分にのみ高精度なレンジ解析手法を適用することで，精度とスケーラビリティの両方を確保する．また，実験により，提案手法は既存手法と違い，精度とスケーラビリティの両立ができていることを確認した．今後の展望としては，以下が挙げられる．

1. 提案手法で利用する高精度なレンジ解析手法として，SMT ソルバーを用いる手法以外の手法を利用した場合における精度とスケーラビリティの評価を行う．
2. 実際に開発されている車載モデルに対して提案手法を適用し，実用的な精度，計算時間となるかどうかの評価を行う．
3. 提案手法における内部状態が存在するブロックのレンジが収束しなかったときの出力レンジを，無限大ではなく，ループ不変量の解析手法などを取り入れることで，より厳密に推定できるようにする．

参考文献

[1] MathWorks: Fixed-Point Designer Documentation - MathWorks, MathWorks(online), available from <http://www.mathworks.co.jp/jp/help/fixedpoint/index.html> (accessed 2014-06-05).
[2] dSPACE：dSPACE - 変数の自動スケーリング，dSPACE（オンライン），入手先 <https://www.dspace.com/ja/jpn/home/products/sw/pcgs/targetli/scaling.cfm>（参照 2014-06-23）.
[3] Fang, C. F. et al.: *Toward efficient static analysis of finite-precision effects in DSP applications via affine arithmetic modeling*, In Proceedings of the 40th annual Design Automation Conference, pp. 496–501 (2003).
[4] Moore, R. E., Kearfott, R. B. and Cloud, M. J.: *Introduction to interval analysis*, Siam (2009).
[5] Cousot, P. and Cousot, R.: *Abstract interpretation: a unified lattice model for static analysis of programs*, In Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages, pp. 238–252 (1977).
[6] 関文貴ほか：制御ソフトウェアの固定小数点演算化ツールの設計と実装，情報処理学会研究報告，ソフトウェア工学研究会報告，No. 27, pp. 1–9 (2010).
[7] Hickey, T., Ju, Q. and Van Emden, M. H.: *Interval arithmetic: From principles to implementation*, Journal of the ACM (JACM), Vol. 48, No. 5, pp. 1038–1068 (2001).
[8] Benhamou, F. and Older, W. J.: *Applying interval arithmetic to real, integer, and boolean constraints*, The Journal of Logic Programming, Vol. 32, No. 1, pp. 1–24 (1997).
[9] De Figueiredo, L. H. and Stolfi, J.: *Affine arithmetic: concepts and applications*, Numerical Algorithms, Vol. 37, No. 1–4, pp. 147–158 (2004).
[10] Brez, M. and Hoffstätter, G.: *Computation and application of Taylor polynomials with interval remainder bounds*, Reliable Computing, Vol. 4, No. 1, pp. 83–97 (1998).
[11] Nedialkov, N. S., Kreinovich, V. and Starks, S. A.: *Interval arithmetic, affine arithmetic, taylor series methods: Why, what next?*, Numerical Algorithms, Vol. 37, No. 1–4, pp. 325–336 (2004).
[12] Kinsman, A. B. and Nicolici, N.: *Finite precision bit-width allocation using SAT-modulo theory*, In Proceedings of the Conference on Design, Automation and Test in Europe, pp. 1106–1111 (2009).
[13] Herber, P., Reicherdt, R. and Bittner, P.: *Bit-precise formal verification of discrete-time MATLAB/Simulink models using SMT solving*, In Proceedings of the Eleventh ACM International Conference on Embedded Software, IEEE Press, Article No. 8, pp. 1–10 (2013).
[14] Florian, D. et al.: *Clone detection in automotive model-based development*, In Proceedings of the 30th International Conference on Software Engineering, pp. 603–612 (2008).
[15] Reicherdt, R. and Glesner, S.: *Slicing MATLAB Simulink models*, In Proceedings of the 34th International Conference on Software Engineering, pp. 551–561 (2012).
[16] 石畑 清：アルゴリズムとデータ構造, 岩波書店 (1989).
[17] 小澤 孝夫：コンピュータ・アルゴリズム 基礎からプログラミングへ, 昭晃堂 (1990).
[18] De Moura, L. and Bjørner, N.: *Z3: An efficient SMT solver*, In Conference on Tools and Algorithms for the Construction and Analysis of Systems, pp. 337–340

(2008).

付　　録

IA における除算は以下のように定義される．

$$X\,/\,Y = \begin{cases} [x^L, x^U] \times [1/y^U, 1/x^L] \\ \qquad \text{if } 0 \notin [y^L, y^U], \\ (-\infty, \infty) \\ \qquad \text{if } 0 \in [x^L, x^U] \wedge 0 \in [y^L, y^U], \\ [x^U/y^L, \infty) \\ \qquad \text{if } x^U < 0 \wedge y^L < y^U = 0, \\ (-\infty, x^U/y^U] \cup [x^U/y^L, \infty) \\ \qquad \text{if } x^U < 0 \wedge y^L < 0 < y^U, \\ (-\infty, x^U/y^U] \\ \qquad \text{if } x^U < 0 \wedge 0 = y^L < y^U, \\ (-\infty, x^L/y^L] \\ \qquad \text{if } 0 < x^L \wedge y^L < y^U = 0, \\ (-\infty, x^L/y^L] \cup [x^L/y^U, \infty) \\ \qquad \text{if } 0 < x^L \wedge y^L < 0 < y^U, \\ [x^L/y^U, \infty) \\ \qquad \text{if } 0 < x^L \wedge 0 = y^L < y^U, \\ \emptyset \\ \qquad \text{if } 0 \notin [x^L, x^U] \wedge y^L = y^U = 0. \end{cases} \tag{A.1}$$