Vol.2019-SE-202 No.5
2019/7/12

要求仕様書を対象とした状態遷移表作成支援とその評価

中村 成[1]　吉田 則裕[1,a)]　山本 椋太[1]　高田 広章[1]

概要：要求仕様書は多くの場合，自然言語で記述されるため，記述内容の誤り（抜け漏れや曖昧さ）が存在しうる．要求仕様書の品質と開発の成否は大きく関わっており，開発を成功させるためにも，要求仕様書中の誤りを早い段階で検出し，除去することが望まれる．そこで，開発現場では要求分析の結果をモデリングすることが一般的であり，多くのモデリング技法が提案されている．特に状態遷移表は事象と状態が大規模・複雑化した場合でも取り扱いやすく，ある状態において事象が発生したときの処理についての誤りを発見することができる．しかし，モデリングする要求仕様書の規模は大きいもので 2000 ページ程にもなりうる．このような膨大な規模の要求仕様書から状態遷移表を手作業で作成する場合，多大な時間を要する．そこで，本研究では組込みシステムの要求仕様書から状態遷移表の作成支援を行うための手法を提案する．

Supporting the generation of state transition tables in a requirements specification document and its evaluation

Naru Nakamura[1]　Norihiro Yoshida[1,a)]　Ryota Yamamoto[1]　Hiroaki Takada[1]

1. はじめに

要求仕様書は多くの場合，自然言語で記述される [1][2]．そのため，記述内容の誤り（抜け漏れや曖昧さ）が存在しうる [3]．この誤りを要求定義で低減することができれば，要求定義以降のプロセスでの手戻りや最終的なプロジェクト・コストの超過を回避でき，プロジェクトの成功につながる [4]．つまり，要求仕様書の品質と開発の成否は大きく関わっており，開発を成功させるためにも，要求仕様書中の誤りを早い段階で検出し，除去することが望まれる [5][6][7]．そこで，開発現場では要求分析の結果をモデリングすることが一般的であり，多くのモデリング技法が提案されている [8]．特に状態遷移表は事象と状態が大規模・複雑化した場合でも取り扱いやすく [9]，ある状態において事象が発生したときの処理についての誤りを発見することができる [10]．しかし，モデリングする要求仕様書の規模は大きいもので 2000 ページ程にもなりうる [11]．このような膨大な規模の要求仕様書から状態遷移表を手作業で作成する場合，多大な時間を要する．

本稿では，我々が既存研究において提案した手法 [12], [13] を拡張し，組込みシステムの要求仕様書を対象として状態遷移表の作成を行う開発者を支援するための手法を提案する．具体的には，ユーザに状態遷移表の要素候補（状態名候補やイベント候補）を提示し，ユーザがインタラクティブに入力・選択を行うことで，状態遷移表の作成支援を可能にする．提案手法を実現するため，形態素解析システム JUMAN と日本語構文・格・照応解析システム KNP を用いた自然言語処理を行い，要求仕様を解析する．解析した要求仕様から定義・条件・処理を表す要素を抽出する．ユーザの入力・選択履歴を用いて自動決定可能な要素は自動抽出し，自動決定困難な要素はユーザに状態遷移表の要素候補を提示して入力・選択を行うようにする．状態遷移表の要素が全て決定すると，状態遷移表のベースと状態遷移表の要素の組を提示する．ユーザに状態遷移表の要素の組を確認してもらうことで，状態遷移表のベースに状態遷移表の要素の組が全て書き込まれる．提案手法の有効性を評価するため，手作業と提案手法を実装した支援ツールを用いて対照実験を行った．

[1] 名古屋大学
Nagoya University, Nagoya, Aichi 464–8601, Japan
[a)] yoshida@ertl.jp

2. 提案手法

本章では，提案手法である要求仕様書から状態遷移表の作成支援を行うための手法について述べる．要求仕様書から状態遷移表の作成支援過程を図 1 に示す．紙面の都合上，提案手法の詳細を説明することはできないため，本稿では状態遷移記述の抽出および状態遷移表の作成を中心に主要な点のみを述べる．提案手法の詳細については，文献 [14] を参照されたい．

2.1 節の抽出・分類

節分類，節の定義，節の分類条件および例を表 1 に示す．表 1 に示すように，定義節と処理節は条件節の有無で分類する．条件節を分類するためには品詞パターンを用いることを考えられるが，品詞およびその細分類と見出し語の組み合わせは多岐に渡るため，実装コストが大きい．そこで，KNP が出力する feature 情報を利用した文節への属性付与 [15] を用いる．本研究では [15] で示されている属性のうち，“type”，“parallel”，“attribute” を用いる．節の抽出・分類の詳細については，文献 [14] を参照されたい．

2.2 状態遷移記述の抽出

本節では，状態遷移表を作成する際に必要な要求仕様書中の要素（以下，状態遷移記述と呼ぶ）をインタラクティブに抽出する手法を述べる．状態遷移記述を抽出することで，状態遷移記述の組 STD を作成する．ここで，状態遷移記述の組 STD について以下を定義する．

【定義 1】（状態遷移記述の組 STD）

$STD = (sc, ev, pr, tr)$

$sc \in \mathbf{VD} \cup \mathbf{ST}$：状態値判定または状態値変化に関する条件

$ev \in \mathbf{EV}$：イベント

$pr \in \mathbf{PR}$：処理

$tr \in \mathbf{TR}$：遷移

STD は必ず以下が成り立つ．

- sc と ev のうち少なくとも一方は存在する．
- pr と tr のうち少なくとも一方は存在する．

また，2.1 節で分類した節と箇条書きについて以下を定義する．

【定義 2】（節分類と箇条書きの分類）

$d \in \mathbf{D}$：定義節

$c \in \mathbf{C}$ ：条件節

$p \in \mathbf{P}$ ：処理節

$id \in \mathbf{ID}$：定義節に付属する箇条書き

$ic \in \mathbf{IC}$ ：条件節に付属する箇条書き

$ip \in \mathbf{IP}$ ：処理節に付属する箇条書き

2.2.1 状態名の選択

どの状態名に基づいて状態遷移表を書きたいかは，ユーザ次第である．そのため，以下の【手順 1】を行う．

【手順 1】（状態名の選択）

$d_0, \ldots, d_k$ における主語を抽出する．

$sn = extract\_s(d)$

抽出した主語 $sn_0, \ldots, sn_i$ の集合が状態名候補 **SN** になる．

ユーザは **SN** から状態名を 1 つ選択する．

k は 2.1 節で分類した定義節の数である．i は $d_0, \ldots, d_k$ から抽出した主語の数である．本研究で対象とする要求仕様書には，1 文字の名詞が状態名となる場合が存在しないため，状態名候補から除去する（例：表）．また，‘初期値’ を含む主語も同様に状態名になり得ないため，状態名候補から除去する（例：運転モードの初期値）．

2.2.2 状態値の入力

KNP は確率モデルを用いた構文解析であるため，状態値となる文字列を状態値候補として抽出できない場合が存在しうる．そのため，ユーザに状態値候補を提示し，ユーザが状態値を選択する方法は困難である．そこで，以下の【手順 2】を行う．

【手順 2】（状態値の入力）

sn_k を含む d_k をユーザに提示する．

d_k が箇条書きを付属する場合，id もユーザに提示する．

ユーザは d_k と id を目視で確認して，状態値となる文字列を全て入力する．

ユーザが入力した状態値を **SV** とする．

sn_k は 2.2.1 節でユーザが選択した状態名，d_k は sn_k を含む定義節である．また，**SV** はユーザが入力した状態値の集合である．

2.2.3 状態値判定と状態値変化に関する条件の抽出

状態値判定の抽出は，以下の【手順 3-1】と【手順 3-2】を行う．

【手順 3-1】（状態値判定の抽出）

$c_0, \ldots, c_e$ について以下の全ての条件を満たすものを **VD** とする．

$$\begin{cases} sv \in \mathbf{SV} \text{ を 1 つ以上含む．} \\ sn_k \text{を含む．} \\ ContainsVerb(c) \text{ が偽を返す．} \end{cases}$$

$ic_0, \ldots, ic_f$ について以下の全ての条件を満たすものを **VD** とする．

$$\begin{cases} sv \in \mathbf{SV} \text{ を 1 つ以上含む．} \\ sn_k \text{を含む．} \\ ContainsVerb(ic) \text{ が偽を返す．} \end{cases}$$

e は 2.1 節で分類した c の数，f は 2.1 節で分類した ic の数である．$ContainsVerb(c)$ は c が動詞を含むかを真 or 偽で返す関数である．$ContainsVerb(ic)$ は ic が動詞を含むかを真 or 偽で返す関数である．**VD** の要素に接続詞『ま

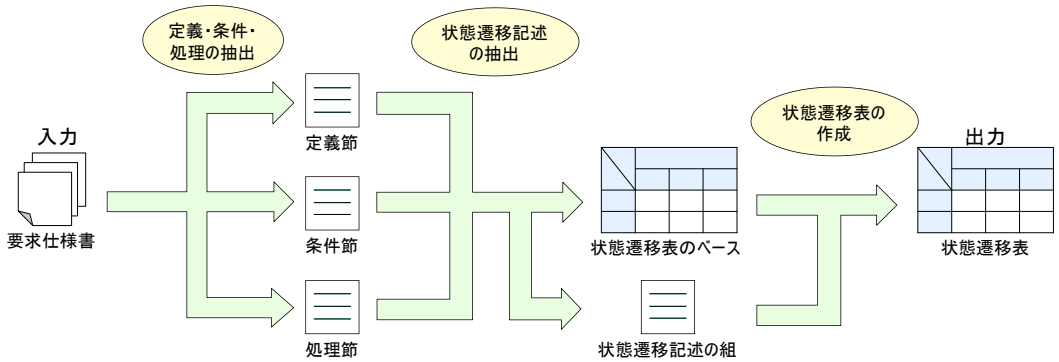

図 1 状態遷移表の作成過程

表 1 分類する節に関する情報

節分類	節の定義	節の分類条件	例
定義節	状態を定義している節	条件節を伴わない	スイッチは「オン」「オフ」のいずれかの値を取る。
条件節	条件を表す節	KNP の出力結果である feature 情報を用いる	モード切替ボタンを「オン」にすると、
処理節	条件を満足した時に実行される節	直前の節が条件節である	＜条件節＞モード選択画面を表示する。

たは』や『かつ』を文頭に持つ条件節が存在する場合，その文頭の接続詞は必要がないため，除去する．次に，**【手順 3-2】**より，同じ意味を表す状態値判定を統合する．

【手順 3-2】（状態値判定の重複記述の統合）

$$\mathbf{VD} \ / \ R_g = \{[vd_a] \mid vd_a \in \mathbf{VD}\ \} (0 \leq g \leq a)$$

文字列完全一致する記述は，ユーザに提示する前に自動的に統合する．その後，ユーザに $vd_0, \ldots, vd_a$ を提示し，ユーザは関係 R_g を用いて，同じ意味を表す状態値判定を統合する．a は抽出した状態値判定（文字列完全一致は統合済み）の数である．

状態値変化に関する条件の抽出は，以下の**【手順 4-1】**と**【手順 4-2】**を行う．

【手順 4-1】（状態値変化に関する条件の抽出）

$c_0, \ldots, c_h$ について以下の全ての条件を満たすものを **ST** とする．

$$\begin{cases} sv \in \mathbf{SV} \text{ を 1 つ以上含む．} \\ sn_k \text{を含む．} \\ ContainsVerb(c) \text{ が真を返す．} \end{cases}$$

$ic_0, \ldots, ic_j$ について以下の全ての条件を満たすものを **ST** とする．

$$\begin{cases} sv \in \mathbf{SV} \text{ を 1 つ以上含む．} \\ sn_k \text{を含む．} \\ ContainsVerb(ic) \text{ が真を返す．} \end{cases}$$

h は 2.1 節で分類した c の数から箇条書きでない状態値判定を除いた数，j は 2.1 節で分類した ic の数から箇条書きの状態値判定を除いた数．**ST** の要素に接続詞『または』や『かつ』を文頭に持つ条件節が存在する場合，その文頭の接続詞は必要がないため，除去する．

次に，**【手順 3-2】**より，同じ意味を表す状態値判定を統合する．

【手順 3-2】（状態値判定の重複記述の統合）

$$\mathbf{VD} \ / \ R_g = \{[vd_a] \mid vd_a \in \mathbf{VD}\ \} (0 \leq g \leq a)$$

文字列完全一致する記述は，ユーザに提示する前に自動的に統合する．その後，ユーザに $vd_0, /dots, vd_a$ を提示し，ユーザは関係 R_g を用いて，同じ意味を表す状態値判定を統合する．a は抽出した状態値判定（文字列完全一致は統合済み）の数である．

状態値変化に関する条件の抽出は，以下の**【手順 4-1】**と**【手順 4-2】**を行う．

【手順 4-1】（状態値変化に関する条件の抽出）

$c_0, /dots, c_h$ について以下の全ての条件を満たすものを **ST** とする．

$$\begin{cases} sv \in \mathbf{SV} \text{ を 1 つ以上含む．} \\ sn_k \text{を含む．} \\ ContainsVerb(c) \text{ が真を返す．} \end{cases}$$

$ic_0, /dots, ic_j$ について以下の全ての条件を満たすものを **ST** とする．

$$\begin{cases} sv \in \mathbf{SV} \text{ を 1 つ以上含む．} \\ sn_k \text{を含む．} \\ ContainsVerb(ic) \text{ が真を返す．} \end{cases}$$

h は 2.1 節で分類した c の数から箇条書きでない状態値判定を除いた数，j は 2.1 節で分類した ic の数から箇条書きの状態値判定を除いた数．**ST** の要素に接続詞『または』や『かつ』を文頭に持つ条件節が存在する場合，その文頭

の接続詞は必要がないため，除去する．次に，【手順 4-2】より，同じ意味を表す状態値変化に関する条件を統合する．

【手順 4-2】（状態値変化に関する条件の重複記述の統合）

$$\mathbf{ST} \ / \ R_l = \{[st_b] \ — \ st_b \in \mathbf{ST} \ \} \ (0 \leq l \leq b)$$

文字列完全一致する記述は，ユーザに提示する前に自動的に統合する．その後，ユーザに $st_0, /dots, st_b$ を提示し，ユーザは関係 R_l を用いて，同じ意味を表す状態値変化に関する条件を統合する．b は抽出した状態値変化に関する条件（文字列完全一致は統合済み）の数である．

2.2.4 イベントの抽出・選択

イベントには自動決定可能なイベントと，ユーザに選択させるイベントの 2 種類がある．まず，自動決定可能なイベント（以下，自動決定イベントと呼ぶ）について述べる．自動決定イベントは，状態値判定もしくは，状態値変化に関する条件に伴う条件節を指す．自動決定イベントの抽出には，以下の【手順 5】を行う．

【手順 5】（自動決定イベントの抽出）

$cell_0, /dots, cell_m$ について，$contain(cell, ele)$ が真を返す．この時，$ele \in \mathbf{D} \cup \mathbf{C} \cup \mathbf{P} \cup \mathbf{ID} \cup \mathbf{IC} \cup \mathbf{IP}$ である．m は要求仕様書のセルの数である．『$contain(cell, ele)$ が真を返す』とは，要求仕様書の各セルに書かれている文のうち，ele に分類可能である文を指す．$c_0, /dots, c_n$ について以下の全ての条件を満たすものを $\mathbf{V}$ とする．

$$\begin{cases} ele \in \mathbf{VD} \cup \mathbf{ST} \\ \text{ele の前後いずれかに存在する．} \\ c \in \overline{\mathbf{VD}} \cap \overline{\mathbf{ST}} \end{cases}$$

c が箇条書きを付属する場合は，その箇条書き ic も $\mathbf{V}$ とする．

$\mathbf{V}$ は自動決定イベントの集合である．n は状態値判定でない，かつ状態値変化に関する条件でない条件節の数である．この時，状態値判定や状態値変化に関する条件と $\mathbf{V}$ の関係によって，各文で現れる状態遷移記述の組の数が異なる．状態遷移記述の組数を決定するためのフローの概要を図 2 に示す．図 2 の入力は状態値判定もしくは状態値変化に関する条件である．状態値判定と状態値変化に関する条件は 5 つに分類することができる．5 つの分類について詳しく説明する．まず，（分類 1）は ‘OR 箇条書きへの参照’ に関する分類である．‘OR 箇条書きへの参照’ とは，箇条書きに状態値判定もしくは状態値変化に関する条件を持ち，文頭が『以下のいずれかの』である条件節を指す．詳しい分類フローを図 3 に示す．図 3 の “$None$” は該当する情報がない，“??” はまだ抽出プロセスに達していない処理と遷移である．

次に，（分類 2）は ‘AND 箇条書きへの参照’ に関する分類である．‘AND 箇条書きへの参照’ とは，箇条書きに状態値判定もしくは状態値変化に関する条件を持ち，文頭が『以下の全ての』である条件節を指す．詳しい分類フローを図 4 に示す．図 4 の “$None$” は該当する情報がない，“??” はまだ抽出プロセスに達していない処理と遷移である．

次に，（分類 3）は ‘接続関係（OR/AND）を持たない条件’ である．‘接続関係（OR/AND）を持たない条件’ とは，箇条書きでなく，他の条件節との接続関係（OR/AND）を持たない状態値判定と状態値変化に関する条件を指す．詳しい分類フローを図 5 に示す．図 5 の “$None$” は該当する情報がない，“??” はまだ抽出プロセスに達していない処理と遷移である．

次に，（分類 4）は ‘AND 条件節’ に関する分類である．‘AND 条件節’ とは，箇条書きでなく他の条件節との接続関係（AND）を持つ状態値判定と状態値変化に関する条件を指す．詳しい分類フローを図 6 に示す．図 6 の “??” はまだ抽出プロセスに達していない処理と遷移である．具体的な分類手法は以下に説明する．

次に，（分類 5）は ‘OR 条件節’ に関する分類である．‘OR 条件節’ とは，箇条書きでなく他の条件節との接続関係（OR）を持つ状態値判定と状態値変化に関する条件を指す．詳しい分類フローを図 7 に示す．図 7 の “$None$” は該当する情報がない，“??” はまだ抽出プロセスに達していない処理と遷移である．具体的な分類手法は以下に説明する．

自動決定困難なイベントは，まず以下の【手順 6-1】を行う．

【手順 6-1】（自動決定困難なイベントの抽出）

$c_0, /dots, c_q$ について以下の全ての条件を満たすものを $\mathbf{CDV}$ とする．

$$\begin{cases} \text{状態値判定と状態値変化に関する条件を伴わない．} \\ c \in \overline{\mathbf{VD}} \cap \overline{\mathbf{ST}} \cap \overline{\mathbf{V}} \end{cases}$$

c が箇条書きを付属する場合は，その箇条書き ic も $\mathbf{CDV}$ とする．

$\mathbf{CDV}$ は自動決定困難なイベントの集合である．q は状態値判定でない，状態値変化に関する条件でない，かつ自動決定イベントでない条件節の数である．

次に，同じ意味を表す $\mathbf{CDV}$ の要素と $\mathbf{V}$ の要素を統合する【手順 6-2】を行う．

【手順 6-2】（重複記述の統合）

$$\mathbf{OV} \ / \ R_s = \{[ov_r] \ — \ ov_r \in \mathbf{OV} \ \} \ (0 \leq s \leq r)$$

文字列完全一致する記述は，ユーザに提示する前に自動的に統合する．その後，ユーザに $\mathbf{CDV}$ と $\mathbf{V}$ の和集合 $\mathbf{OV}$ を提示し，ユーザは関係 R_s を用いて，同じ意味を表す条件節を統合する．この時，ユーザによって，$\mathbf{CDV}$ の要素が $\mathbf{V}$ の要素と同じ意味を表す記述であると判断された場合，その $\mathbf{CDV}$ の要素は $\mathbf{V}$ に移動する．r は $\mathbf{OV}$ の要素（文字列完全一致は統合済み）の数である．

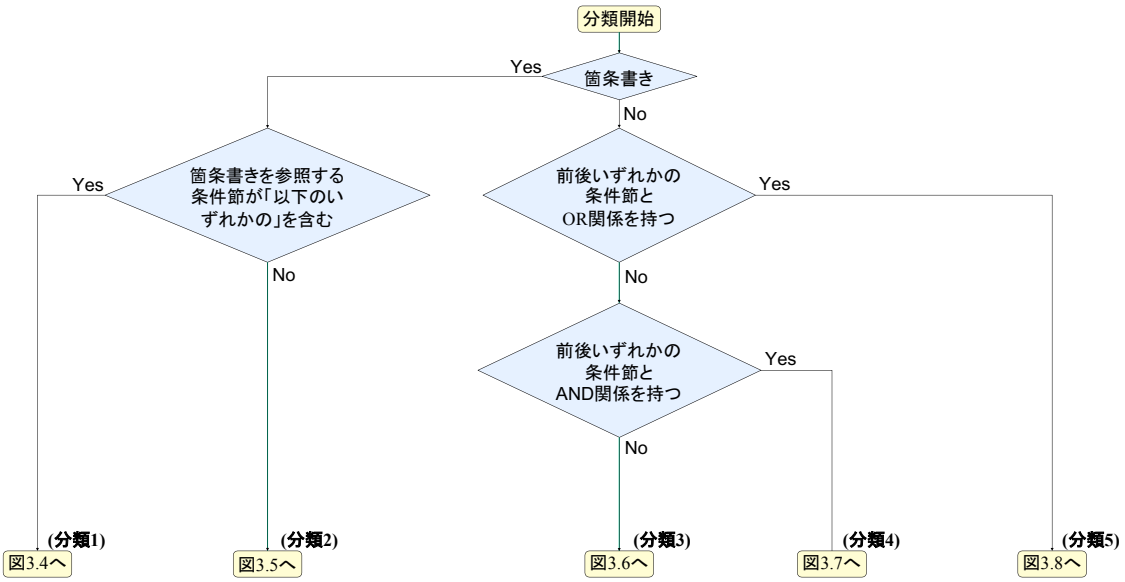

図 2　状態遷移記述の組数決定フローの概要

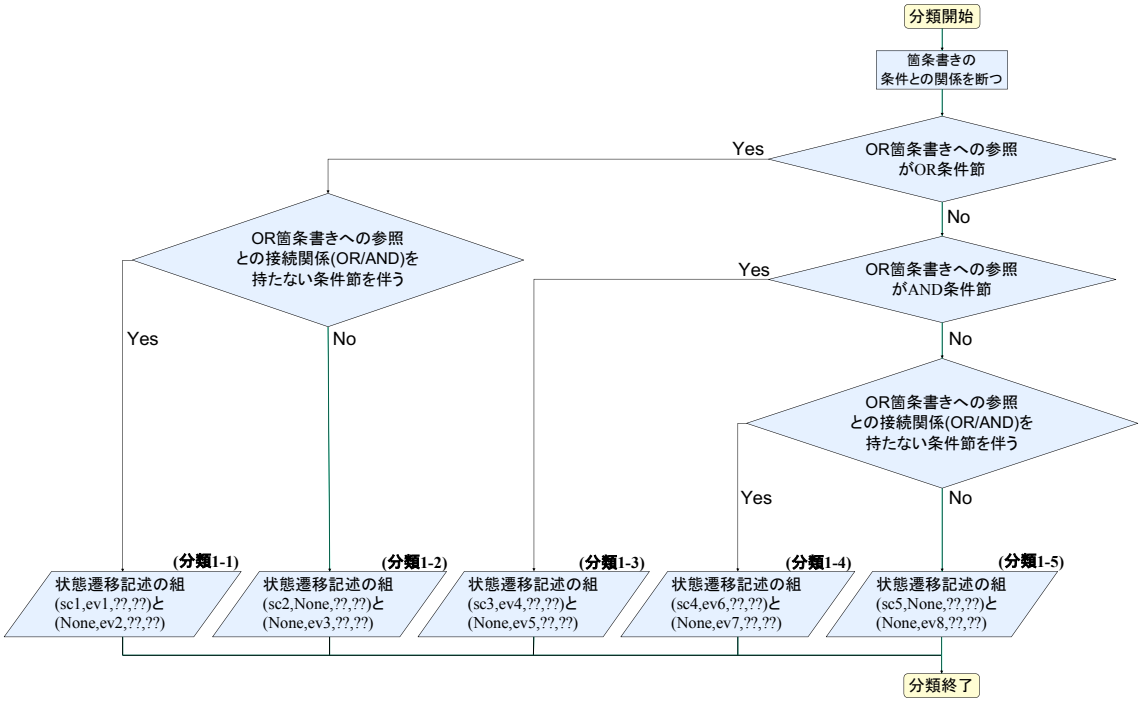

図 3　‘OR 箇条書きへの参照’ に関する状態遷移記述の組

重複記述の統合後，【手順 6-3】を行う．

【手順 6-3】（自動決定困難なイベントからイベントの選択）

ユーザに $pv_0, /dots, pv_u$ をランキング形式で提示する．

ユーザがランキングを目視で確認して，選択した条件節を **VS** とする．

u は **CDV** から重複記述を除いた要素の数である．ランキング作成には，2 つの文字列の類似度を測るレーベンシュタイン距離 [16] を用いる．このランキングは状態値判定と関連性の高い条件節が上位に現れるランキングである．**VS** はユーザが選択したイベントの集合である．**V** と **VS** の和集合が **EV** となる．イベントを提示する時に，2.2.5 節で後述する処理・遷移も提示する．これは本研究において，制約・定義・理由のいずれかを伴う条件節はイベントになり得ないと考えており，その判断をユーザが行う際に必要な情報であるため，提示する．

2.2.5　処理・遷移の抽出

処理・遷移の抽出には，以下の【手順 7-1】と【手順 7-2】を行う．

【手順 7-1】（処理・遷移の抽出）

$p_0, /dots, p_w$ について，状態値判定，状態値変化に関

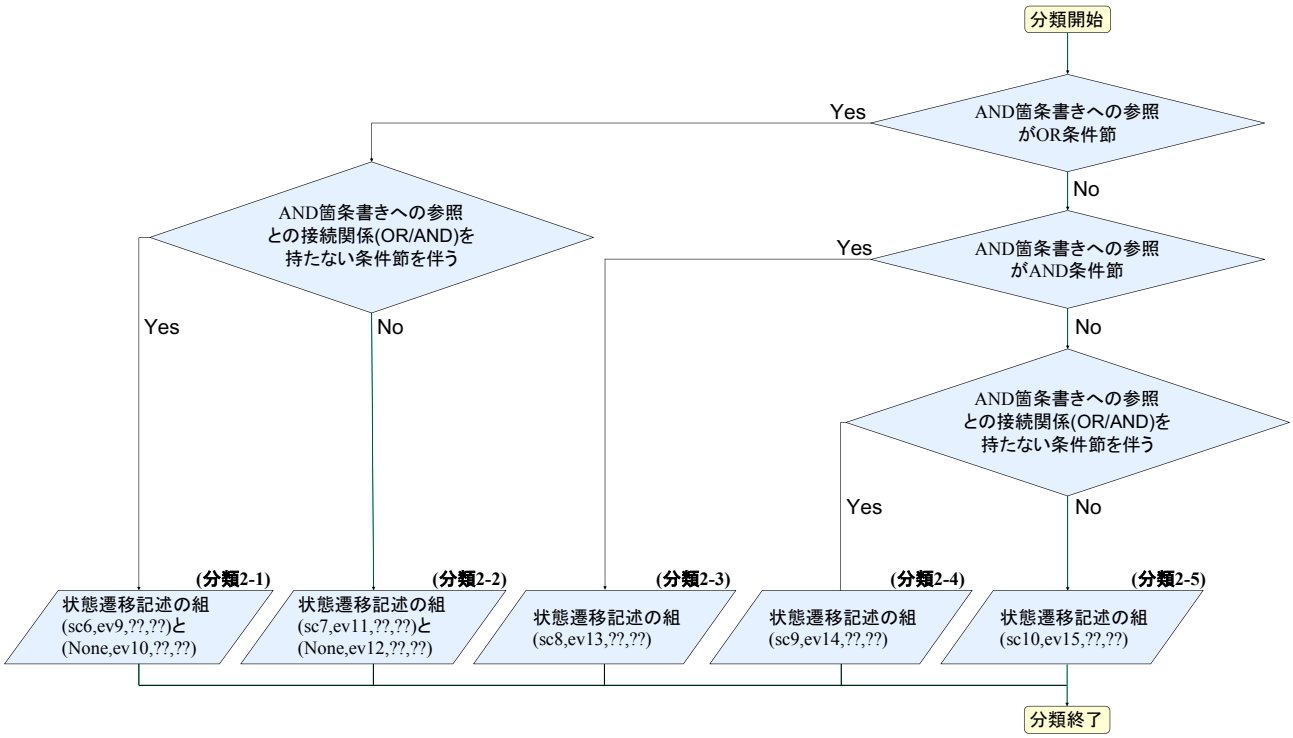

図 4 ‘AND 箇条書きへの参照’ に関する状態遷移記述の組

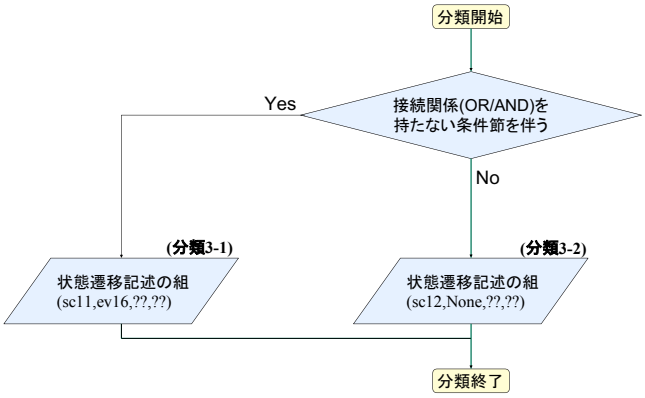

図 5 ‘接続関係（OR/AND）を持たない条件’ に関する状態遷移記述の組

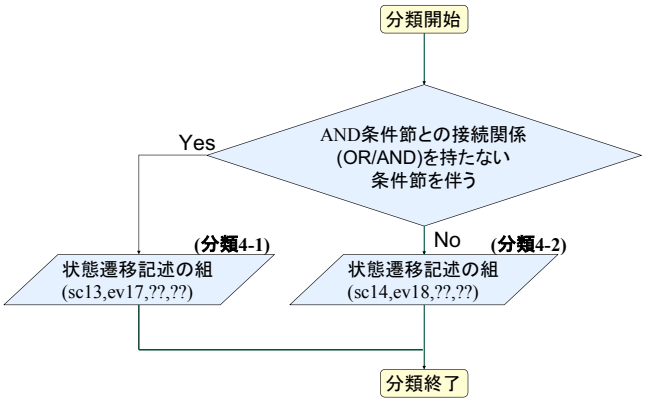

図 6 ‘AND 条件節’ に関する状態遷移記述の組

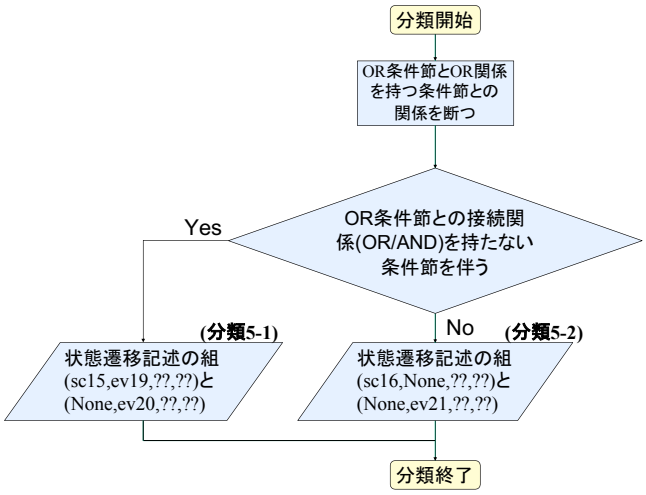

図 7 ‘OR 条件節’ に関する状態遷移記述の組

する条件，イベントのうち少なくとも 1 つを伴うものを **PT** とする．

p が箇条書きを付属する場合は，その箇条書き ip も **PT** とする．**PT** のうち，sn_k を含まないものを **PR**，sn_k を含むものを **TR** とする．1 文に複数の処理が存在する場合は，1 つに結合したものを処理と見る．

w は 2.1 節で分類した処理節の数である．次に，【手順 7-2】より，同じ意味を表す処理・遷移を統合する．

【手順 7-2】（処理・遷移の重複記述の統合）

$$\mathbf{PT} \ / \ R_x = \{[pt_y] \ — \ pt_y \in \mathbf{PT} \ \} \ (0 \leq x \leq y)$$

文字列完全一致する記述は，ユーザに提示する前に自動的に統合する．その後，ユーザに $pt_0, /dots, pt_y$ を提示し，

図 8　作成した状態遷移表の枠組み（状態値が 3 つ，イベントが 2 つの場合）

ユーザは関係 R_x を用いて，同じ意味を表す処理・遷移を統合する．y は抽出した処理・遷移（文字列完全一致は統合済み）の数である．

【手順 1】から**【手順 7-2】**を行うことで，状態遷移記述を全て抽出することが可能になる．したがって，状態遷移記述の組 STD を作成することが可能になる．

2.3　状態遷移表の作成

2.2 節で得た状態遷移記述の組 STD を用いて状態遷移表を作成する．まず，状態遷移表の行と列を自動作成する．状態遷移表の行は，状態名を書き込む行，状態値を書き込む行，イベントに伴う処理を書き込む行，イベントに伴う遷移を書き込む行が必要であるため，行数は $2 + |EV| * 2$ である．状態遷移表の列は，イベントを書き込む列と各状態値を書き込む列が必要であるため，列数は $1 + |SV|$ である．以上より，図 8 のような状態遷移表の枠組みを作成することが可能になる．

次に，処理・遷移を書き込む．処理・遷移には以下の場合が存在する．

(i)　イベントのみ伴う．
(ii)　状態値判定を伴うが，イベントを伴わない．
(iii)　状態値判定とイベントを伴う．
(iv)　状態値変化に関する条件を伴うが，イベントを伴わない．
(v)　状態値変化に関する条件とイベントを伴う．

(i) はどの状態値の時でも起こる処理・遷移であるため，伴うイベントが書かれている行全てに書き込む．**(ii)** と **(iii)** は状態値判定を伴うため，その状態値判定がどの状態値に対応するか知る必要がある．意味解析を行えば，ある程度対応づけを自動で行うことが可能かもしれないが，確実性高めるため，本手法では意味解析を行わない．そこで，2.2.2 節で得た **SV** と，2.2.3 節で得た **VD** をユーザに提示し，状態値判定がどの状態値を表す条件か，ユーザが対応づけを行う．

この対応づけ結果を用いると，**(ii)** は状態値判定が表す状態値の列全てに書き込む．**(iii)** は状態値判定が表す状態値の列と，その状態値判定に伴うイベントの行が交差するセルに書き込む．この時，1 つのセルに同じ要素が書き込まれることはない．**(i)**，**(ii)**，**(iii)** を対応するセル全てに書き込み終えた場合，その表と状態遷移記述の組を出力する．**(iv)** と **(v)** は，ユーザに状態値変化に関する条件を含む状態遷移記述の組を提示し，ユーザが対応するセルを目視で確認して書き込む．これは対応づけと同様に意味解析を行わないため，ユーザによる書き込みで実現する．

3.　評価実験

本章では，2 章で述べた提案手法の有効性を評価するための実験と，その結果に対する考察について述べる．本実験の対象を以下に示す．

対象 1　キッチンタイマの要求仕様書
対象 2　車載システムの要求仕様書

対象 1 の規模はテキストファイル 41 個，**対象 2** の規模はテキストファイル 243 個である．各テキストファイルには箇条書き 1 つ，または 1 文が格納されている．**対象 1** と**対象 2** は共に企業が作成した要求仕様書である．2 章で述べた提案手法の有効性を評価するため，**対象 1** と**対象 2** の要求仕様書に対して，6 名の被験者による手作業および支援ツールによる評価実験を行う．被験者 6 名は状態遷移表と無関係の研究を取り組む，本学の情報システム学専攻の学生である．これらの被験者を 3 名ずつ 2 グループに分ける．具体的な実験手順を説明する．本実験では，支援ツールを用いて状態遷移表を作成する場合と，手作業で状態遷移表を作成する場合の 2 フェーズがある．この 2 フェーズは 2 日に分けて実施する．

実験の前段階として，状態遷移表と支援ツールの使用法，および**対象 1** と**対象 2** のドメイン知識の説明を行った．被験者の目標は制限時間内に**対象 1** と**対象 2** それぞれにおいて，妥当と考えられる状態遷移表を作成することである．制限時間は支援ツールを使用する場合，手作業の場合共に 4 時間とする．制限時間を超えた場合，その時点で作業を打ち切る．支援ツールを使用する場合は，**??**章で示した**ステップ 1** から**ステップ 12** を行い，状態遷移表を作成する．手作業の場合は，要求仕様書を目視で参照して状態遷移表を作成する．被験者が作成する状態遷移表の書式は 2.3 節で述べた図 8 の通りである．実験中は，要求仕様書を適宜見ても良く，エディタでの検索機能の使用を許可する．

本実験では，作成する状態遷移表を 1 つに指定するため，各フェーズにおいて状態名を実験実施者が指定した．被験者は自身が妥当な状態遷移表を作成できたと判断した時点で，作成した状態遷移表を実験実施者に提出する．提出した状態遷移表に誤りがある場合，実験実施者が誤りを指摘し，被験者が修正する．この修正作業は，実験実施者が考案した基準を満たすまで繰り返される．実験実施者が考案した基準は以下の通りである．

- 2.3 節で述べた図 8 のフォーマットを満たす．
- 状態値が過不足なく存在する．
- 状態値判定を伴う処理・遷移が不足なく存在する．

表 2　状態遷移表作成の所要時間

	対象 1	対象 2
グループ 1	支援ツールなし	支援ツールあり
被験者 A	72 分	216 分
被験者 B	127 分	131 分
被験者 C	171 分	122 分
グループ 2	支援ツールあり	支援ツールなし
被験者 D	78 分	240 分
被験者 E	66 分	240 分
被験者 F	92 分	225 分

表 3　支援ツールの有無による比較

	平均	最大	最小
支援ツールあり	118 分	216 分	66 分
支援ツールなし（時間切れ含む）	179 分	240 分	72 分
支援ツールなし（時間切れ含まない）	149 分	225 分	72 分

- 状態値判定を伴うイベントが不足なく存在する．
- 処理・遷移が状態とイベントの組み合わせに対応するセルに書かれている．

一度に伝える誤りの数は，提出時点で実験実施者が発見した全ての修正箇所数である．

各被験者が**対象 1** と**対象 2** において，妥当と考えられる状態遷移表を作成するまでの時間を表 2 にまとめる．表 2 の被験者 D と被験者 E の 240 分 00 秒は制限時間内に妥当な状態遷移表を作成することが叶わなかったため，作業を打ち切った時間である．

支援ツールの使用有無ごとに集計した表 3 にまとめる．表 3 より，支援ツールありの方が支援ツールなしと比べて，平均・最大・最小時間のすべてについて小さい値になっていることがわかる．この実験結果は，支援ツールを使用すると効率的に状態遷移表を抽出できることを示している考えられる．

4. おわりに

本研究では，組込みシステムの要求仕様書から状態遷移表の作成支援を行うための手法を提案した．

実験において，提案手法の有効性を評価するため，被験者に手作業の場合と支援ツールを用いた場合それぞれで状態遷移表を作成する作業を行わせた．実験の結果，支援ツールを用いると，手作業の場合と比較して妥当な状態遷移表の作成時間を短縮することが確認できた．また，状態遷移表の作成経験が浅いユーザであっても，支援ツールをインタラクティブに操作することで状態遷移表の作成が可能であることが確認できた．

謝辞　本研究について，貴重なご意見をくださった名古屋大学 大学院情報学研究科 価値創造研究センター　笹野遼平 准教授に感謝いたします．

参考文献

[1] 位野木万里，近藤公久：省略と修飾パターンを用いた用語不一致検証による要求仕様の一貫性検証支援ツールの実現と適用評価，コンピュータソフトウェア，Vol. 35, No. 3, pp. 109–127 (2018).
[2] 大森洋一，荒木啓二郎：自然言語による仕様記述の形式モデルへの変換を利用した品質向上に向けて，情報処理学会論文誌プログラミング， Vol. 3, No. 5, pp. 18–28 (2010).
[3] 佐藤芳信，高橋　弘，西田廣治：フィードバック制御用組込みソフトウェアにおける要求仕様書記述方式と開発環境，第 69 回全国大会講演論文集，Vol. 2007, No. 1, pp. 181–182 (2007).
[4] 渡邊由香，武田善行：要求仕様書の文書構造における抜け漏れ防止対策としての非機能要件・曖昧性抽出，プロジェクトマネジメント学会研究発表大会予稿集 2013 年度秋季，pp. 238–239 (2013).
[5] 山田さつき, 大森隆行, 大西　淳：要求文書からの信頼性要求の抽出と検証, 情報処理学会研究報告, Vol. 2018-SE-199, No. 28, pp. 1–8 (2018).
[6] 喜多義弘，鈴木三紀夫，秋山浩一，片山徹郎，西　康晴：ソフトウェア要求仕様書に基づいたテスト項目作成手法の提案，ソフトウェアエンジニアリングシンポジウム 2011 論文集，pp. 1–6 (2011).
[7] 有賀　顕，林　晋平，佐伯元司：構文と文章構造に基づく要求仕様書の問題点発見支援，情報処理学会研究報告，Vol. 2013, No. 4, pp. 1–8 (2013).
[8] 岸　知二：高信頼性組み込みソフトウェア開発-最新技術動向と取り組み: 3. モデル検査技術による UML 設計検証，情報処理， Vol. 47, No. 5, pp. 498–505 (2006).
[9] 渡辺政彦：状態遷移ベースのソフトウェア開発環境の現状と動向，計測と制御，Vol. 41, No. 2, pp. 117–121 (2002).
[10] 山本椋太，吉田則裕，竹田彰彦，館　伸幸，高田広章：組込みソフトウェアを対象とした状態遷移表抽出手法，電子情報通信学会技術研究報告，Vol. 116, No. 127, pp. 13–18 (2016).
[11] Boutkova, E. and Houdek, F.: Semi-Automatic Identification of Features in Requirement Specifications, *Proc. of RE 2011*, pp. 313–318 (2011).
[12] 中村　成，山本椋太，吉田則裕，高田広章：組込みシステムの要求仕様書を対象とした状態遷移モデル作成支援，電子情報通信学会技術研究報告，Vol. 118, No. 230, pp. 25–30 (2018).
[13] 中村　成，山本椋太，吉田則裕，高田広章：組込みシステムを対象とした要求仕様書からの状態遷移記述の抽出，情報処理学会研究報告，Vol. 2018-SE-198, No. 5, pp. 1–8.
[14] 中村　成：組込みシステムの要求仕様書を対象とした状態遷移表作成支援，名古屋大学大学院情報学研究科 情報システム学専攻 修士論文 (2019). https://sites.google.com/site/yoshidaatnu/naru.pdf.
[15] 村上響一，青山裕介，村上神龍，久代紀之，牧　茂，田畑一政，神代　勉，中村　潤：自然言語仕様書からの試験ケース生成のための条件・動作の同定手法，情報処理学会研究報告，Vol. 2018-SE-198, No. 7, pp. 1–7 (2018).
[16] Levenshtein, V. I.: Binary codes capable of correcting deletions, insertions, and reversals, *Soviet physics doklady*, Vol. 10, No. 8, pp. 707–710 (1966).