生涯学習支援のための計算モデル

大矢野　　潤*

概要

インターネットに代表されるコンピュータネットワークは一般家庭にまで急速に普及し，その上で提供されるサービスは電子メールやホームページの検索機能にとどまらない。これまで独立に存在した社会基盤をインターネット上に再構築し，統合することにより新たなサービスを創出し始めている。これらのサービスはVoIP，電子政府，IP放送システムなど多岐にわたり，当然教育インフラもその視野に含まれる。本論文では，現在のe-Learningシステムの状況と問題点について考察する。さらに，生涯学習支援システムをインターネット上に構築するために，すでに一般的であるHTMLに代わる仕組みであるXML，セマンティックWebとその意味論であるRDFモデル理論を紹介する。そして，RDFモデルの述語を状態遷移システムとして解釈することにより時制論理を導入する。さらに単なる参照関係としてのWebリンクを依存関係として再考し，さまざまな教育資源と他のサービスを組み合わせた「コース」を作り出すことを可能にする。そして，自律分散的に存在するコースを統合し，効率よく管理・利用するための枠組みと理論を与える。

1　はじめに

近年の社会情勢の急激な変化に伴い，社会基盤に対する要求も著しく変化してい

*yanpi@cuc.ac.jp

る。高齢化・少子化問題，終身雇用制の崩壊，インターネットの普及によってもたらされた情報化などは，大学が社会に提供する教育資源の変化を要請している。これらは生涯学習や大学院大学，高度専門教育などの18才人口に限定しない学習機会の創出，e-Learning等を応用した教育資源のオンライン化への要求などの形で表面化している。

筆者はこれらの問題に対応するために「ユビキタス・ユニバーシティ(Ubiquitous University)」というパラダイムを提案している。ユビキタス・ユニバーシティとは，直観的には「いつでも，どこでも，だれでも就学可能な大学」という意味である。このいつでも，どこでも，だれでも（anytime，anywhere，anybody）を支援するための技術的な基盤としてのインターネットの利用と，その上で構築されるサービスを効率よく提供するための仕組みであるWWW（World Wide Web）の整備が活発に行われている［2］。

これらの技術の普及と進歩を背景に，生涯学習に対する社会的要求に応えるべく多数の大学や企業が教育コンテンツの提供と新しいビジネスチャンスの創出に向け努力してきたが，今のところ，失敗しているケースが多数見られる。これは，現在の教育インフラでは生涯学習には十分に対応する能力が欠けているが，反面，活路と目されるインターネットを利用した技術は未だ発展段階にあり十分に成熟しているとはいえない。とりわけ，インターネットの技術は高度なヒューマンコミュニケーションである教育活動，および技術伝播を効率よく行えるほどには成熟していない。さらに，ネットワーク上に教育コンテンツを構築することにより失われる現実世界でのコミュニケーション，特に就学者の学習意欲を維持することが難しいこと，そしてオンラインシステム・教材の構築には膨大な費用がかかるため，現状では圧倒的に教材が不足していることも指摘される。良質の教材，カリキュラムなどの学習コンテンツは社会的に共有し利用する仕組みが必要であり，反面，投入した資金や労力を直接学生から回収するというビジネスモデルはもはや成立しない。これらの議論からe-Learningシステムには生涯学習特有とも思われる問題点を内包しており，安易な教育コンテンツのオンライン化は現状を単に悪化させる傾向にあるといえる。

これらの問題点のうち，就学のカリキュラムをインターネット上で実装できていないことがひとつの要因であると考える。カリキュラムを実装する場合には順序を

保証する枠組が必要となる。教材同士の関連を何らかの形で記述したもの，つまり，カリキュラム自体も教材であるという認識が重要である。筆者は，カリキュラムをインターネット上で提供するためにコースの概念を導入することを提案する。また，教育インフラは電子商取引などの他のサービスとの融合が必須であり，コースはそのための手段としても有効であるといえる。

本論文では，教育コンテンツを提供するための基本的な仕組みとしてXML（eX-tensible Markup Language）の利用を仮定する。また，カリキュラムを実装するための仕組みとして，セマンティックWeb（Semantic Web）の枠組みを利用する。セマンティックWebの構成技術であるOWL/RDF-S/RDFはいずれもXMLの枠組みの中で記述され，それらはすべてRDF（Resource Description Format）のモデル理論（RDF Model Theory）上で展開される。RDFモデル理論はプリミティブな状態遷移システムをモデル領域としてもち，その意味論はKIF（Knowledge Interchange Format）によって記述される。ここでKIFの意味論は，本質的には自己適応が行われるため，本質的には高階の理論となり，non-wellfoundedな集合，つまりRussel集合を包含してしまう。これらの不都合を避けるために，KIFは外延の展開を制限すること，つまり，外延とそれに含まれる要素の字句的に区別することで，一階の述語論理に等価な理論を展開している。反面，KIFの適応範囲に高階論理（Higher Order Logic）や，無限論理（Infinitary Logic）を含める活動も行われている。

ここではコースの依存関係に基づく順序を導入するために，KIFの論理体系に時制論理（Temporal Logic）を導入する。具体的には，RDFによって構築される状態遷移システム（Transition System）をモデルとし，そのモデル上での可達性の検証（reachability check）をおこなうことにより，目標とするゴールにたどり着くためのパス（path）もしくはラン（run）の上で「次の」ステップを発見する。ここでは，次の状態が正しいかどうかを判定するためにモデルチェック（Model Checking）の技術を利用する。具体的には時制を含んだ可達性の問題を様相mu計算（modal mu-calculus）の式で記述し，その式を満たすモデルの発見をモデルチェックのアルゴリズムを用いて行う。これは，ネットワークのすべての意味を確定するのではなく，ローカルモデルチェックでの適当なパスを発見するプロセスであ

り，ここで重要な点は妥当性（validity）ではなく充足可能性（satisfiability）にあることに注意する。

最後に，コースの抽象化に対する提案を行う。これはインターネットの空間は本質的には無限であり，有限モデルの性質を利用したとしても，検証する状態が多くなる場合には全体のパフォーマンスに重大な影響をもたらすことが知られている。このため，全体のモデルを抽象化し，より扱いやすいモデルを構築することを目標とする。

2　インターネット上の生涯学習支援環境

2.1　ユビキタス・ユニバーシティ

情報技術，いわゆるIT（Information Technology）の発達に伴ったハードウェアの価格の急激な低下やユーザインタフェイスの進歩にともない，コンピュータの利用形態は大きな転換期を向かえた。わずか十数年ほど前，一部の研究者に利用されていた大型コンピュータの，ゆうに数百倍ものデータ処理速度をもつパソコン（以下，PC）が，現在ではWindowsやMac OSなどの基本ソフトウェアを搭載して，一般家庭へ普及している。このコンピュータの利用形態の変化をもたらしたITの進歩は今のところ停滞する気配をみせず，我々をさらなる新しい世界へ誘おうとしている。この，大型コンピュータ，PCにつづく第3の波と目されているのがユビキタス・コンピューティング（Ubiquitous Computing）というパラダイムである。

ユビキタスという聞きなれない言葉は「遍在」を意味し，いつでもどこでも必要な情報を得るための環境について語っている。ユビキタス・コンピューティングはユーザが空間を移動している状態での通信，いわゆる移動体通信（Mobile Computing）をその技術的な基礎としている。しかし，ユビキタス・コンピューティングというパラダイムは決して情報技術のみに注目するものではなく，むしろ，コンピュータの持つ情報処理能力を現実世界の中で活用するための概念である。

ITを積極的に利用することにより，情報へのアクセス可能性の飛躍的な向上を期待することができる。卑近な例では，専門分野に関する論文を検索する場合に，研

究室の隣の建物にある図書館に出向くよりも，英国の大学のWebサイトで公開されている論文データにアクセスし，自室のプリンタに出力する方がはるかに速い場合がほとんどである。また，地図検索，航空券やホテルの予約，天気予報など日常生活においてもインターネットは欠かせない存在になっている。この情報へのアクセス可能性が拡大することへの期待が人々にPCを購入させ，インターネットへの参加を促している。そして情報のユビキタス化はその傾向をさらに加速させていくと見られている。

本論文では，この情報という言葉を「生涯教育」という範疇に特化して議論を進めていく。つまり，いつでも，どこでも，誰でも大学等に就学できる環境の構築を目指すこと，ユビキタス・ユニバーシティ（Ubiquitous University）というパラダイムを提案する。

2.2 コンピュータネットワークによる生涯学習支援

コンピュータを教育支援に応用しようという試み，いわゆるCAI（Computer Assisted Instruction）は1980年代より積極的に研究され，応用ツールも盛んに開発されてきた。CAIの研究が盛んになってきた当時は，コンピュータ利用者の大半は研究者や技術者であったため，CAIの対象も暗黙のうちに専門職の卵を仮定している場合が多く見られた。近年のコンピュータネットワークの一般家庭への普及に伴い，CAIの主な対象は一般市民へと遷移している。そして，コンピュータを教師の代わりとする色彩の強いものから，自習や遠隔学習（Distance Learning）を行なうための道具へと変化して来ている。コンピュータ・ネットワークの遠隔学習に関する応用は，近年e-Learningとして知られ，研究開発のみならず盛んに実践されている。この試みは，主に時間や地理的理由により，授業が行なわれている時間にその教室で授業に参加することができない人への就学支援，いわゆる学習に関する情報格差（デジタルデバイド）の解消を目的としている。

2.3 e-Learningの現状

ここでe-Learningの現状について考察することにしよう［3,4,5］。米国の広大な国土と進んだIT環境はe-Learningの普及の条件を満たしているように見え，当然のようにe-Learningをベースにしたビジネスへの取り組みが行われてきた。とりわけ，2001年4月4日，MIT（マサチューセッツ工科大学）が打ち出したOCW（Open Course Ware）計画は各方面に衝撃を与えた。OCW計画とは今後10年間で8,500万ドルを投じ，約2000コースに上るMITの全てのコースのシラバス，講義資料，参考文献，評価問題などをWeb上で無料で公開するものである。この計画の社会的意義が非常に高い反面，ここにかかるコストの大きさにも着目しなければならない。このコストは単に教師が普段行っている授業コースをWeb化するだけでも教師に対して高い負荷をかけることを物語っている。また，コース自体は無償で提供するため，費用は別の部門で捻出する必要があること，また，コースに投入される知的所有権の取り扱いなど運用上の問題が多い。

このような問題を含みつつ，米国のオンライン関連の知識市場に関する強気の見通しは各大学のe-Learningへの参入を積極的に促した［4］。ある予想では1994年の94億ドルから2003年533億ドルと年率54%での成長を見込んだものもあり，e-Learningの分野はITバブル経済の中にあって格好のターゲットとなったといえる。しかし，カリフォルニアバーチャル大学は1997年に制定されたものの1999年4月に廃止，ウェスタン・ガバナーズ大学1996年6月制定，2001年9月方向転換，バーチャルテンプルは1999年11月創設，2001年7月に廃止されるなど［3］苦戦が強いられている。

ここで注意したいことは，これらの米国の例は決してe-Learning自体の可能性を否定するものではないく，むしろ，ニーズの高さを象徴しているということである。反面，現状の授業を単純にオンラインコース化したものはe-Learningに対する本質的な要求を反映できないことを物語っており，このことが採算われを起こす原因になっていることを強調しておきたい。

ユーザは通常就学するほどの金額を払ってまではe-Learningは利用しないという点，また具体的にその知識を必要とする社会人であっても就学意欲の継続が非常に

難しいことが指摘されている。筆者はコンスタントな就学意欲の維持を前提条件にするということや，大学のように各授業が密に連携をとったコースの体系は生涯学習の考え方に沿っていないと考える。言い換えると，時間の空いたときに自分の状況にあわせていつでも就学を可能とする仕組みが必要であると考える。また，学習は教材だけではなくカリキュラム，つまり提供する知識相互の関係を示す情報自体も学習に必要な知識であるため，このカリキュラム自体を教材として提供する仕組みが必要である。

千葉県には「千葉県私立大学・短期大学および放送大学間における単位互換認定」制度が平成10年度から発足している。これは千葉県の24私立大学・14短期大学間での単位互換協定に基づき，一定基準の範囲内で他大学の授業の聴講を可能とし，自身の所属する大学の単位として認定できる制度であり，千葉商科大学も参加している。しかし，実際に利用している学生は非常に少なく，必ずしも制度がうまく機能しているとは言えない。反面，千葉商科大学の姉妹校であり，同一キャンパス内に存在する千葉短期大学との単位互換，および編入時の単位認定は非常に活発に行われている。この一見当然と思える結果は，就学の地理的制限の強さ，および単位互換にはお互いのカリキュラムを熟知していなければ単位互換制度を有意義なものにすることができないことを物語っている。e-Learningには地理的，時間的制約の解消がその効果として期待されるが，コース相互のカリキュラムの連携がない場合には全体のシステムが活性化することは難しい。

たとえば，CISCO CCNA 検定を受験しようとする場合に，システムアドミニストレータ初級合格程度の知識と英検3級程度の英文読解力，そして千葉商科大学のネットワークシステム論（大矢野）を習得した程度の学力が必要であるとしよう。このとき，例えば英検はTOEFLEのスコアと，千葉商科大学の授業は他大学の適当な授業と読み替えたり，また，前提条件にかけている部分を指摘する仕組み，つまり，ネットワーク上でコースを再度構築し維持していく仕組みが必要である。

3 Semantic Webの応用

本節では，インターネット上にコースを構築する場合に利用するセマンティック

Web（Semantic Web）の枠組みについて説明する。

HTML（Hyper Text Markup Language）で記述されたWeb上のコンテンツを検索することを考えてみよう。ユーザがあるインターネット上に存在するであろうコンテンツを検索する場合には，Google（http://www.google.com/）などの検索サイトを利用して検索するのが一般的である。これは文書の意味によって検索しているのではなく，実際には「文字列」による検索が行われる。ここで，千葉商科大学の教員である“大矢野”の専門分野を検索したいが，名前がはっきり思い出せず，とりあえず，千葉商科大学の“先生”で検索をかける場合を考える。ここで，大学のWebページに“大矢野助教授”と言う記載しかない場合にはこの問い合わせは当然失敗する。“先生”と“助教授”は文字列的に包含関係が成立しないからである。同様のケースとして“教諭”，“講師”などの文字列を用いた場合にも意味のある検索を行うことができない。

もちろん，YAHOO（http://www.yahoo.co.jp/），GOO（http://www.goo.ne.jp/）など文書の内容によって検索範囲を限定したり，ディレクトリサービスを提供するシステムは存在し，成功している。しかしこれらの意味づけは通常サイトやページ単位で行われ，かつ第三者の人間が行うために膨大なWeb空間の中の細かい文書の意味づけを行うことができない。

セマンティックWebとは，通常人間が読んで理解していたHTMLの文章を，XML（eXtensible Markup Language）やRDF（Resource Description Format）などの書式に従って記述することにより，機械が読んで理解できるようにする仕組みである。正確な意味を厳密に定義することにより，「助教授は先生である」という機械的な推論が可能になり，「意味による」Webの検索が可能になる。

まず，セマンティックWebを記述する基本的な書式であるXMLについて簡単に説明する。

3.1 XML

B2B（Business to Business）やB2C（Business to Customer），さらにC2C（Customer to Customer）のいわゆる異業種間コミュニケーションを促進しようとした場合に

はコミュニケーションを構成するデータフォーマットの統一化は必須である。このため，ISO（International Organization for Standardization：国際標準化機構）は1986年に標準化された文書フォーマットとしてSGML（Standard Generalized Marckup Language）を制定した。HTML（Hyper Text Markup Language）はこのSGMLに基づき，W3C（World Wide Web Consortium）により定められたインターネット上での文書の公開を目的としたフォーマットである。しかし，HTMLの普及はHTML自身の持つ機能的欠点を明らかにし，さらに共有，交換などに関して高機能な文書のフォーマットが必要となってきた。特に，HTMLではできなかったタグ（tag）と呼ばれる文書のMarkupを行う要素を自由に定義したり，文書の構造を自由に適する機能，そして文書に意味を記述することなどを目的としてXML（eXtensible Markup Language）が1998年にW3Cにより制定された。

XML文書は構成要素として次の3つの要素をもつ。

1．XML宣言（XMLの文書であるという宣言，およびXMLのバージョン，文字コードの指定）

2．DTD（Document Type Definition: XML文書要素の文書型定義）

3．XML文書要素（要素名，要素の内容，意味構造などを定義）

ここで，DTDが文書の要素と別に定義できることに注意する。この定義により機械的に文書の論理構造の処理が可能になる。たとえば図1の例では文書構造の中に「千葉学園（chibagakuen）/千葉商科大学（cuc）/政策情報学部（pi）」という階層構造が機械処理可能な形で記述できることがわかる。

このように記述されたXML文書は，XPathなどの技術を用いて自由な要素指定が可能になることからデータベースとしての利用が可能になり，またSOAP（Simple Object Access Protocol）を用いてデータの交換が可能になる。また，XSL（eXtensible Stylesheet Language）を用いて自由な書式に変換して表示することができる。

3.2 Semantic Webの意味階層

セマンティックWebはXMLの文書形式を利用して定義される。XMLが文書（オ

```
< ?xml version = "1.0" encoding = "SJIS_JIS">
< !DOCTYPE chibagakuen [
< !ELEMENT chibagakuen (cuc|cjc|cuchs)
< !ELEMENT cuc (ec|pi)>
< !ELEMENT cjc (bc)>
< !ELEMENT ec (#PCDATA)>
< !ELEMENT pi (#PCDATA)>
< !ELEMENT bc (#PCDATA)>
< !ELEMENT cuchs (#PCDATA)>
]>

<chibagakuen>
    <cuc>
        <ec> 商経学部 </ec>
        <pi> 政策情報学部 </pi>
    </cuc>
    <cjc>
        <bc> ビジネスコミュニケーション学科 </bc>
    </cjc>
    <cuchs> 付属高校 </cuchs>
</chibagakuen>
```

図１：XMLの記述例

ブジェクト)[1]のデータ形式と具体的な内容を定義するのに対し，セマンティックWebはオブジェクトの満たす性質や，オブジェクト間の関係を定義していく。セマンティックWebはそれぞれの定義している内容のレベルにより９階層に分類されている（図２参照)。この階層は，たとえばRDF（Resource Description Format）はXMLのみを，RDF-S（RDF Schema）はRDFとXMLのみを用いるという純粋な参照関係によって生成される。

セマンティックWebの意味論はRDFモデル理論［８］を利用して構築されるため，セマンティックWebの一般的なモデル理論はRDFモデル理論を理解すれば十分である。また，実際に規定する意味はKIF（Knowledge Interchange Format）を用いて記述されている。

RDFのモデルは$S-P-O$の三つ組みで表現される。ここで，それぞれSは主語

(1) XMLではすべてのものはオブジェクトとして参照される

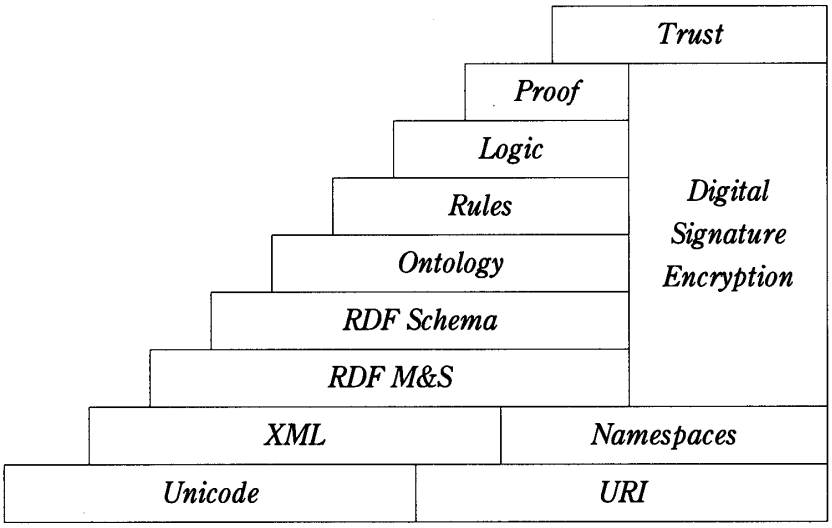

図2：セマンティック *Web* の階層構造

(Subject)，Pは述語（Predicate)，Oは目的語（Object）の意味である。これをセマンティックネットワークのような図的表現に変換するとすると次のようになる。

$$S \xrightarrow{P} O$$

また，これを通常の述語論理の範囲で記述すると次のようになる。

$$(S, O) \in P,\ P(S, O)$$

たとえばPをex: MemberOf（所属している)，Sをyanpi@cuc.ac.jp，OをCUC（千葉商科大学）とすると，それぞれ次のように記述できる。

$$\text{yanpi@cuc.ac.jp} \xrightarrow{\text{ex: MemberOf}} \text{CUC}$$

$$(\text{yanpi@cuc.ac.jp},\ \text{CUC}) \in \text{ex: MemberOf}$$

3.3 Knowledge Interchange Formatの意味論

前述のとおり，セマンティックWebの意味論は基本的にRDFモデル理論上の $S-P-O$ 三つ組みで表現される。この三つ組みはXML/RDFパーサ（parser：構文解析機）によって抽出される。この $S-P-O$ のトリプルを $(S, O) \in P$ もしくは $P(S, O)$ という述語に変換し，KIF上の述語体系にマップする。このことによりKIFで規定する意味論との対応をとることができる。

KIF［6］とは知識工学や人工知能の分野で広く用いられているフォーマットで

あり，さらに現在ではSUO（Standard Upper Ontology）において活発に開発が進んでいる。KIF（SKIF）の記述力は基本的には一階の述語論理でありLISPと互換性を持つ形式で記述された文法を持ち，この意味で，他の定理証明機との連携を容易にしている。

またKIFの特徴としては機械の文法チェックの負担を軽減するために自由文法（free syntax）になっている。これはオブジェクト，定数，関数記号，関係記号の間に区別がなく，すべてを項（term）として記述する。このため，KIFは本質的に自己適応な文の記述が可能となるため，ユーザは整礎でない（non-wellfounded）な記述を避けるために注意をする必要がある。

3.4 KIFの解釈

直観的にはKIFの意味論は非常に単純であり，解釈は容易である。ここではKIFの解釈［7］について基本的な部分を紹介する。

まず，解釈（interpretation）とは意味関数iであり，KIFの定数と概念（conceptualization）を結合する。意味関数iが解釈を行うためには，次のように定数を適当な型の概念にマップする。

1．σがオブジェクト定数のとき，$i(\sigma)\in O$

2．σ関数定数のとき$i(\sigma):O^{*}\longrightarrow O$

3．σ関係記号のとき$i(\sigma)\subseteq O^{*}$

4．σ論理記号のとき$i(\sigma)\in\{true,\ false\}$.

次にiはいくつかの命題（axiom）を充足（satisfy）させる。ここではKIFは部分的（partially interpreted）な言語であることに注意したい。つまり，残りの部分はユーザが定義できるようになっている。

ここで，項（term）の意味は項の記号に意味値（semantic value）を対応させることによって与えられる。Tを項の集合，Oをオブジェクトの集合，vを変数に対する付値関数とすると項の意味関数s_{iv}は次のようになる。

$$s_{iv}:T\longrightarrow O$$

$$s_{iv}(v)=v(v),\ s_{iv}(\sigma)=i(\sigma)$$

また，同様に項に真偽値を対応させる関数 t_{iv} は次のようになる。

$$t_{iv} : S \longrightarrow \{true,\ false\}$$

$$t_{iv}(\lambda) = i(\lambda)$$

ここで，それぞれの項の型によって前述の型に従って再帰的に意味付けが行われる。ここでは典型的な意味関数の適応ルールを例としてあげる。

関数：

$$s_{iv}((\pi\ \tau_1 \ldots\ \tau_n)) = i(\pi)[s_{iv}(\tau_1), \ldots, s_{iv}(\tau_n)]$$

関係：

$$t_{iv}((\rho\ \tau_1 \ldots\ \tau_n)) = \begin{cases} true & \langle s_{iv}(\tau_1), \ldots, s_{iv}(\tau_n)\rangle \in i(\rho) \\ false & \text{otherwise} \end{cases}$$

論理：

$$t_{iv}((\textbf{not}\ \ \phi)) = \begin{cases} true & t_{iv}(\phi) = false \\ false & \text{otherwise} \end{cases}$$

$$t_{iv}((\textbf{and}\ \ \phi_1 \ldots\ \phi_n)) = \begin{cases} true & t_{iv}(\phi_j) = true \text{ for all } j \quad 1 \le j \le n \\ false & \text{otherwise} \end{cases}$$

$$t_{iv}((\textbf{exists}\ \ (v_1 \ldots\ v_k \omega)\ \phi)) = \begin{cases} true & \exists v' t_{iv'}(\phi) = true \\ false & \text{otherwise} \end{cases}$$

$$t_{iv}((\textbf{forall}\ \ (v_1 \ldots\ v_k \omega)\ \phi)) = \begin{cases} true & \forall v' t_{iv'}(\phi) = true \\ false & \text{otherwise} \end{cases}$$

4　時制論理

哲学や数学の領域で盛んに研究されてきた論理体系と計算との密接な関連性が明らかになるにつれ，論理体系で取り扱う「正しさ」をプログラムに対して応用する技術が盛んに開発されてきた。プログラムは時間と共に動的にその性質を変化させるものであり，逆に言うとその時間と共に変わる性質こそがプログラムに意味を与えているともいえる。このとき，時制（tense）に関連したプログラムに対する言明の正当性は通常の命題論理（Propositional Logic），一階述語論理（First Order Predicate Logic）では取り扱うことはできない。たとえば，倉庫に格納されている

商品の数量を管理するプログラムを考え，変数Xにその個数が格納されるとしよう。このとき，最初は$X=0$つまり在庫数が0であるとする。また在庫を一つ増やす操作は$X:=X+1$で記述することができる。この操作が行われた直後には初期値が0であった変数Xの値は1になっているはずである。つまり，プログラムの中で$X=0$の状態と$X=1$つまり$X\neq 0$，もしくは$\neg(X=0)$の状態が存在する。これを通常の命題論理や，一般の述語論理の体系の中で表すと次のように矛盾（$\perp$）が導出される。

$$\frac{X=0 \quad \neg(X=0)}{\perp}$$

これは変数のように変化する内容を持つものを単一の式で表現しようとすることに起因する。このため，ある時間tにおける変数Xの値つまり，$X_t=0$，$X_{t+1}\neq 0$のように記述するか変数Xを時間の関数として記述し$X(t)=0$，$X(t+1)\neq 0$と記述することにより回避することができる。しかし，この場合命題論理では可算無限の項が必要になる。また，述語論理においても例えば「変数Xはずっと100よりも小さい（倉庫があふれない）」という言明を有限の長さの論理式で記述すること[2]はできない。一方，今まで見てきたように，プログラムがこのような性質を満たすことを証明することに対する要求はきわめて自然である。

これらの問題点を克服するために，可算無限の長さの論理式の導入する方法[3]（Infinitary Logic）や，時間のモデルを論理体系の外に置き，時間の経過を様相（mode）ととらえ様相論理の枠組みの中で性質を記述する方法などがある。プログラムの挙動を取り扱う様相論理のうち代表的なものにはLTL（Linear Time Logic），PLTL（Propositional Linear Time Logic），BTTL（Branching Time Temporal Logic），CTL（Computational Tree Logic），CTL*などがある。ここではこれらを包含する記述力を持つ命題様相mu計算（Propositional Modal mu-Calculus，以下mu計算）と，その検証系としてmu計算に基づいたモデルチェックを行うことを考える。

---

(2) $X(t)<100\wedge X(t+1)<100\wedge X(t+2)<100\ldots$

(3) $\wedge_{0\leq t} X(t)<100$

4.1 mu-Calculus

mu計算はKozenらによって導入され，その証明体系の論理的な健全性（Soundness）や完全性（Completeness）に関する理論的な研究，その応用であるシステムの時間的な性質が活発に行われている［11］。特にモデルとして有限状態機械（FSM, Finite State Machine）をとり，そのモデルの正当性の証明をもちいて電気回路やVLSIなどの検証を行う方法が盛んに行われ，実用化されている［10］。

モデルチェックの方法は主にmu計算のモデルに直接基づく方法と，ROBDD（Reduced Ordered Binary Decision Diagrams）などを利用したシンボリックモデル検証（Symbolic Model Verifier, SMV），タブロー（tableau）を利用した方法などがある。

ここでは直観的に理解しやすいモデル理論に基づく検証法と，タブローを利用した方法を紹介する。タブローによるモデル検証は特に局所モデル検証（Local Model Checking）と呼ばれ，インターネット上のサービスなど本質的に無限の状態を扱う場合に有利である。無限の状態を扱うタブローはBradfield［12］によって健全かつ完全な体系が開発されたが，その無限のもつ性質により証明が非決定的（non-deterministic）になるため，ユーザの証明への介入が必要となり，インターネット上のサービスに組み込むことは難しい。このため，ここでは有限の状態を扱ったCleaveland［13］の方法を紹介する。

4.2 mu計算のシンタックス

mu計算の式Φは変数$X, Y, Z, \ldots$の集合Varとラベル$a, b, \ldots$の集合$\mathcal{L}$から以下の規則によって作られる。

$$\Phi ::= Z \mid \neg \Phi_1 \mid \Phi_1 \wedge \Phi_2 \mid [K]\Phi_1 \mid \nu Z.\Phi_1$$

ここでKの範囲は$\mathcal{L}$の部分集合（$K \subseteq \mathcal{L}$）であり，$\nu Z.\Phi_1$はΦ_1中のZの自由な出現は偶数個の否定記号$\neg$のスコープ内に制限される。また，通常の双対（dual）のオペーレータの略記として以下のような記法を採用する。

$$\mathbf{tt} \stackrel{\text{def}}{=} \nu Z.Z$$

$$\mathbf{ff} \stackrel{\text{def}}{=} \neg \mathbf{tt}$$

$$\Phi_1 \vee \Phi_2 \stackrel{\text{def}}{=} \neg \Phi_1 \wedge \neg \Phi_2$$

$$\langle K \rangle \Phi \stackrel{\text{def}}{=} \neg [K] \neg \Phi$$

$$\mu Z.\Phi \stackrel{\text{def}}{=} \neg \nu Z. \neg \Phi [\neg Z/Z]$$

$$[-K] \Phi \stackrel{\text{def}}{=} [\mathcal{L} - K] \Phi$$

$$[a_1, \ldots, a_n] \Phi \stackrel{\text{def}}{=} [\{a_1, \ldots, a_n\}] \Phi$$

$$[-] \Phi \stackrel{\text{def}}{=} [\mathcal{L}] \Phi$$

ここで，mu計算の式はその意味関数の単調性（monotonicity）を保障するためにpositiveであるという制限がつくことに注意する。これらの論理式の詳しい意味は参考文献［12，13］を参照されたい。

4.3 mu計算のセマンティックス

ここではmu計算の式Φへモデル$\mathcal{M}(\mathcal{T}, \mathcal{V})$に関して意味を与える関数$\| \Phi \|_{\mathcal{V}}^{\mathcal{T}}$を導入する。ここでいう意味づけとは式$\Phi$に式$\Phi$を満たす$\mathcal{M}(\mathcal{T}, \mathcal{V})$上の状態の集合を対応付けることである[4]。

$\mathcal{L}$をラベルの集合，Sを状態の集合，ラベルaについてそれぞれ$\stackrel{a}{\longrightarrow}$を$S$上の二項関係とする。このときソート$\mathcal{L}$を持つラベルつき遷移システム$\mathcal{T}$は$\mathcal{T}=(S, \{\stackrel{a}{\longrightarrow} | a \in \mathcal{L}\})$で与えられる。

mu計算のモデルはソートLのラベルつき遷移システムTと評価関数Vの組$(\mathcal{T}, \mathcal{V})$である。ただし，評価関数$\mathcal{V}$は$\mathcal{V}$: Var$\rightarrow 2^S$で与えられているものとする。以下，特に指定する場合がない場合には$\| \Phi \|_{\mathcal{V}}^{\mathcal{T}}$の$\mathcal{T}$は省略する。

モデル$\mathcal{M}(\mathcal{T}, \mathcal{V})$において式$\Phi$について，$\| \Phi \|_{\mathcal{V}}^{\mathcal{T}}$は以下の規則によって与えられる。

$$\| Z \|_{\mathcal{V}} = \mathcal{V}(Z)$$

$$\| \neg \Phi \|_{\mathcal{V}} = S - \| \Phi \|_{\mathcal{V}}$$

---

(4) $\| \Phi \|_{\mathcal{V}}^{\mathcal{T}} \stackrel{\text{def}}{=} \{s \in S | \mathcal{T}, \mathcal{V}, s \models \Phi\}$

$$
\begin{aligned}
\| \Phi_1 \wedge \Phi_2 \|_\nu &= \| \Phi_1 \|_\nu \cap \| \Phi_2 \|_\nu \\
\| [k] \Phi \|_\nu &= \{ s \in S \mid \forall s' \in S. \forall a \in K. \\
&\qquad s \xrightarrow{a} s' \Rightarrow s' \in \| \Phi \|_\nu \} \\
\| \nu Z. \Phi \|_\nu &= \cup \{ s \subseteq S \mid \| \Phi \|_{\nu[z:=s]} \supseteq S \} \\
\| \Phi_1 \vee \Phi_2 \|_\nu &= \| \Phi_1 \|_\nu \cup \| \Phi_2 \|_\nu
\end{aligned}
$$

略記に対するルールは次のように導き出される。

$$
\begin{aligned}
\| \langle k \rangle \Phi \|_\nu &= \{ s \in S \mid \exists a \in K. \exists s' \in S. \\
&\qquad s \xrightarrow{a} s' \wedge s' \in \| \Phi \|_\nu \} \\
\| \mu Z. \Phi \|_\nu &= \cap \{ s \subseteq S \mid \| \Phi \|_{\nu[z:=s]} \subseteq S \}
\end{aligned}
$$

意味関数$\| \Phi \|_\nu^{\mathcal{T}}$は単調であることが文の構成に関する構造帰納法により証明できる。このことにより，Knaster-Tarskiの重要な定理の適応が可能となり，意味関数$\| \Phi \|_\nu^{\mathcal{T}}$は不動点を持つことが保障される。

[Knaster-Tarski]

$$
\begin{aligned}
\| \mu Z. \Phi \|_\nu^{\mathcal{T}} &= \bigcup_{\alpha \in \mathrm{Ord}} \| \mu^\alpha Z. \Phi \|_\nu^{\mathcal{T}} \\
\| \nu Z. \Phi \|_\nu^{\mathcal{T}} &= \bigcap_{\alpha \in \mathrm{Ord}} \| \nu^\alpha Z. \Phi \|_\nu^{\mathcal{T}}
\end{aligned}
$$

ここで，Ordはすべての順序数のクラスであるとする。

モデル(T, V)での近似式は以下の超限帰納法により定義される。これはモデルチェックで使うΦを満たす状態の集合を求めるのに直接利用することができる。

$$
\begin{aligned}
\| \mu^0 Z. \Phi \|_\nu &= \emptyset \qquad \| \nu^0 Z. \Phi \|_\nu = S \\
\| \sigma^{\alpha+1} Z. \Phi \|_\nu &= \| \Phi \|_{\nu[z:=\| \sigma^\alpha Z \cdot \Phi \|_\nu]} \\
\| \mu^\lambda Z. \Phi \|_\nu &= \bigcup_{\alpha < \lambda} \| \mu^\alpha Z. \Phi \|_\nu \\
\| \nu^\lambda Z. \Phi \|_\nu &= \bigcap_{\alpha < \lambda} \| \nu^\alpha Z. \Phi \|_\nu
\end{aligned}
$$

ここで，λは極限順序数とし，σはμもしくはνであるとする。

このモデル理論の意味論に基づき，あるモデル$\mathcal{M}(\mathcal{T}, \mathcal{V})$上の状態$s$が式Φを満たすことは$\mathcal{M}, s$がΦのモデルになっていることを表している。このことをチェックすることはsが$\| \Phi \|_\nu^{\mathcal{T}}$の示す状態の集合に入っているかどうかをチェックすることで行われる。

$$
\mathcal{M}, s \models \Phi \Longleftrightarrow s \in \| \Phi \|_\nu^{\mathcal{T}}
$$

FSMにおいて$\| \Phi \|_\nu^{\mathcal{T}}$を求めるアルゴリズムは図3のようにあたえられることは

function eval(Φ, e)
if $\Phi = p$ **then return** $\{s \mid p \in \mathrm{L}(s)\}$;
if $\Phi = Q$ **then return** $e(\mathrm{Q})$;
if $\Phi = \Phi_1 \wedge \Phi_2$ **then return** eval$(\Phi_1, e) \cap$ eval(Φ_2, e) ;
if $\Phi = \Phi_1 \vee \Phi_2$ **then return** eval$(\Phi_1, e) \cup$ eval(Φ_2, e) ;
if $\Phi = \langle a \rangle \Phi'$ **then return** $\{s \mid \exists \mathrm{t}[s \xrightarrow{a} t \wedge t \in \mathrm{eval}(\Phi', e)]\}$;
if $\Phi = [a] \Phi'$ **then return** $\{s \mid \forall t[s \xrightarrow{a} t \Rightarrow t \in \mathrm{eval}(\Phi', e)]\}$;

if $\Phi = \mu Q.\Phi(Q)$ **then**
  $Q_{\mathrm{val}} := False$;
  repeat
    $Q_{\mathrm{old}} := Q_{\mathrm{val}}$;
    $Q_{\mathrm{val}} :=$ eval$(\Phi, e[Q \leftarrow Q_{\mathrm{val}}])$;
  until $Q_{\mathrm{val}} = Q_{\mathrm{old}}$;
  return Q_{val};
end if

if $\Phi = \nu Q.\Phi(Q)$ **then**
  $Q_{\mathrm{val}} := True$;
  repeat
    $Q_{\mathrm{old}} := Q_{\mathrm{val}}$;
    $Q_{\mathrm{val}} :=$ eval$(\Phi, e[Q \leftarrow Q_{\mathrm{val}}])$;
  until $Q_{\mathrm{val}} = Q_{\mathrm{old}}$;
  return Q_{val};
end if
end function

図3 ：mu-calculusのモデルチェックアルゴリズム

容易にチェックすることができる。

このmu計算の意味論はその不動点を求めるときにモデル内のすべての状態を走査する必要があることに注意する。また，プログラムやハードウェアの検証では一般に状態数が爆発的に増加すること，いわゆる状態爆発（state explosion）が問題となる。このため，すべての状態を操作するアルゴリズムは大規模なシステムに適応することができない。これに対して，調べようとする状態とモデル上でのその周辺の状態のみを局所的に調べることをローカルモデルチェックという。タブローによる方法はある状態が適当な性質を持つことを，モデル上を横断（traverse）して調べる。つまり，タブローの証明はそのアルゴリズムを表現しており，通常の証明

$$\text{(R1)}\ \frac{H \vdash s \in \neg\neg\Phi}{H \vdash s \in \Phi}$$

$$\text{(R2)}\ \frac{H \vdash s \in \Phi_1 \vee \Phi_2}{H \vdash s \in \Phi_1}$$

$$\text{(R3)}\ \frac{H \vdash s \in \Phi_1 \vee \Phi_2}{H \vdash s \in \Phi_2}$$

$$\text{(R4)}\ \frac{H \vdash s \in \neg(\Phi_1 \vee \Phi_2)}{H \vdash s \in \neg\Phi_1, H \vdash s \in \neg\Phi_2}$$

$$\text{(R5)}\ \frac{H \vdash s \in \langle a \rangle \Phi}{H \vdash s' \in \neg\Phi}(s' \in \{s' \mid s \xrightarrow{a} s'\})$$

$$\text{(R6)}\ \frac{H \vdash s \in \neg\langle a \rangle \Phi}{H \vdash s_1 \in \neg\Phi, H \vdash s_2 \in \neg\Phi \ldots}(\{s_1, s_2, \ldots\} = \{s' \mid s \xrightarrow{a} s'\})$$

$$\text{(R7)}\ \frac{H \vdash s \in \nu X.\Phi}{H' \cup \{s : \nu X.\Phi\} \vdash s \in \Phi[\nu X.\Phi/X]}(s : \nu X.\Phi \notin H)$$

$$\text{(R8)}\ \frac{H \vdash s \in \neg\nu X.\Phi}{H' \cup \{s : \nu X.\Phi\} \vdash s \in \neg\Phi[\nu X.\Phi/X]}(s : \nu X.\Phi \notin H)$$

where $H' = H - \{s' : \Gamma \mid \nu X.\Phi < \Gamma\}$

図4 ：mu計算のタブロー規則 [13]

と異なり後ろ向きの方向で健全性（backward sound）が保障されている。また，ここで紹介する方法は健全かつ完全であることが証明されている。

タブローによる方法のもうひとつの利点は，タブローの探査プロセスを直接プログラム化することができる点である。以下，タブロー規則（図4参照）とそれをチェックするためのアルゴリズム（図5参照）をあわせて紹介し，類似性をみながらこの節を終了する。

5 ネットワーク上のコース構築

本節では，いよいよネットワーク上でのWebオブジェクトとその間の参照関係を依存関係とみなしコースを構築する方法と，具体的にそのコース上で現在遷移可能な状態（オブジェクト）を発見する方法について述べる。

```
fun check1'(H ⊢ s ∈ Φ)
  case Φ is
  A ∈ 𝒜   ⟶ return(s ∈ V(A))
  X ∈ 𝒱   ⟶ error
  ¬Φ'     ⟶ return not check1'(H ⊢ s ∈ Φ')
  Φ1 ∨ Φ2 ⟶ return check1'(H ⊢ s ∈ Φ1) or check1'(H ⊢ s ∈ Φ2)
  ⟨a⟩Φ'   ⟶ foreeach s' ∈ {s' | s -a-> s') do
                 if check1'(H⊢s' ∈ Φ) then return true;
               return false
  νX.Φ'   ⟶ let H' = {s' : Γ | Φ ≮ Γ) in
               return(check1'(H' ∪ {s : Φ} ⊢ s ∈ Φ'[Φ/X])
  esac
end

fun check1(s ∈ Φ) = check1'(∅⊢ s ∈ Φ)
```

図5：タブローを使ったモデルチェックアルゴリズム［13］

5.1 目標と次のステップ

ここではユーザがWeb上で何らかの目的を充足させることを考える。一般にユーザは充足させたいもの，たとえばレストランを開業したいなどの目標は明確に持っている。反面，その目標に達するために必要な条件に関する理解は必ずしも自明ではない。たとえば調理師の免許，もしくはそれを保持しているスタッフの充足，資金，簿記会計等の知識等の前提条件を充足する必要があるとする。これらの知識は，すでに開業している人には自明であると思われるが，今からレストランの開店を志す初心者にこの知識を前提条件として課すことはできないとするのが適当であろう[5)]。この状態のユーザに対して目標に通じる適切な次のステップを提供する知識そのものも重要な知識体系である。本論文ではこの知識体系をカリキュラムと呼ぶこととし，カリキュラムを実現するための依存関係を状態遷移システムで表現する。そして，この依存関係のある状態遷移システムをコースと呼ぶことにする。

(5) 生涯学習の対象者の持つ目標は一般に大学教育の枠外にあるとしたほうが自然であり，このようなユースケースも生涯学習には必要であると考える。

5.2 コースとしての意味ネットワーク

セマンティックWebの参照関係は理論的に意味ネットワークを形成する。ここでは論理的に作られる意味ネットワークを発見しコースとして具体的に構築する方法について述べる。これは具体的にはセマンティックWebで構築されたネットワークをmu計算の枠組みで取り扱うための方法論を意味することに注意したい。

このため，これらの方法論は次の項目を表現している。

1．遷移システム $\mathcal{T}^*$の構築

- ●XML文書（オブジェクト）の状態としての解釈
- ●オブジェクトの参照関係に基づく状態遷移関係の構築

2．オブジェクトを表現する解釈を導入した評価関数 $\mathcal{V}^*$の導入

以下，これらについて説明していく。

5.3 遷移システム$\mathcal{T}^*$

5.3.1 XML文書と状態

まず，XML文書はXMLの名前空間（XML Name Space）の存在によりそれぞれがインターネット上で一意に定まることに注意したい。このため，指定したXML文書であるかどうかを表す述語をKIFの枠組みで記述できる。また，セマンティックWebのオントロジーによりオブジェクト x と等価なオブジェクトを記述することができる。これらのことから，述語 $\mathcal{K}_{[x]}$ を状態 x であること，$K_{[x]}$ を x の同値類であることを表すものとする。このとき，述語とモデル M 上でその述語を満たす状態の集合を同一視できることにする。同様にKIF上の意味関数 s_{iv} はKIFの項を与えるとKIFの概念領域のオブジェクト，つまり，状態を返す関数であったことに注意する。$\mathcal{K}$をKIFの閉じた項の集合とすると，$K \in \mathcal{K}$，$\{K_x\}$，$\{K_{[x]}\}$ はそれぞれ$\mathcal{K}$に含まれることに注意する。このとき，それぞれ意味関数 s^*_{iv} は次のようになる。

$$||K||^{\mathcal{T}^*}_{\mathcal{V}^*} = s^*_{iv}(K)$$

$$s_{iv}^{*}(K_x) = \{x\}$$

$$s_{iv}^{*}(K_{[x]}) = \{s \mid s \in [x]\}$$

ここで，mu計算の文法もKIFの閉じた式を受理するように変更する必要がある。

5.3.2 参照関係と遷移関係

前述のように，モデル構築を行うにあたって状態遷移システムの状態とXML文書を同一視することができ，さらに，同様に状態間の遷移関係を定めることができる。この遷移関係の候補としてはXMLのリンクをあげることができる。ここで，リンクとはAタグ（anchor tag）を解釈したもののことである。しかし，XMLのリンクをそのまま遷移関係に導入したのでは以下の意味で不十分である。

- Aタグは遷移の元（source）の状態からの一方向のみの遷移に限られる
- Aタグで指定した先（destination）の状態に遷移できるとは限らない
- ユーザオブジェクトからは任意の状態を指定し得るために，依存関係の記述ができない

この問題に対処するために遷移関係 $\xrightarrow{a}$ を定める前段階の関係 R_T* を導入する。

関係 R_T* は (s_s, l, s_d) の三つ組みであり，s_s, l, s_d はそれぞれ元状態（source state），ラベル（label），先状態（destination state）とする。このとき，それぞれに特別な記号“★”と⊗を導入し，それらに関する計算規則を導入する。

$$s \otimes s' = \begin{cases} s & s = s' \\ \emptyset & \text{otherwise} \end{cases}$$

$$\bigstar \otimes s = s \quad s \otimes \bigstar = s$$

$$\bigstar \otimes \bigstar = \bigstar$$

$$(s_s^1, l^1, s_d^1) \otimes (s_s^2, l^2, s_d^2) = (s_s^1 \otimes s_s^2, l^1 \otimes l^2, s_d^1 \otimes s_d^2)$$

さらに，状態遷移システム T^* を次のように定義する。

$$T^* = \{(s_s, l, s_d) \mid (s_s, l, s_d) = x \otimes y \textbf{ where } x, y \in R_T \textbf{ and } s_s \neq \emptyset \wedge l \neq \emptyset \wedge s_s \neq \emptyset\}$$

ここで，★は任意を表している。例えば一般のユーザ（u）は無条件ですべての状態に遷移可能であるため，$(u, \bigstar, \bigstar)$ で記述される。反面，任意のユーザから参照可能なオブジェクトは $(\bigstar, \bigstar, o)$ と記述することができる。このため，$(u, \bigstar, \bigstar) \otimes (\bigstar, \bigstar, o) = (u, \bigstar, o)$ となり，ユーザ u はオブジェクト o を参照可能に

なる。

もうひとつ例を挙げよう。

$$R_T* = \{(u, \star, \star), (\star, l, b), (b, m, c)\}$$

というR_T*が与えられたとする。このとき$\mathcal{T}^*$は次のようになる。

$$\mathcal{T}^* = \{(u, l, b), (b, m, c)\}$$

このため，$\mathcal{T}^*$ではuからの遷移は$u \xrightarrow{t} b \xrightarrow{m} c$の遷移に限られ，$u$から$\mathcal{T}^*$直接状態$c$には遷移できないことがわかる。これは$R_T*$から$\mathcal{T}^*$を作る方法が依存関係を生成していることを意味している[6]。

5.4 Next State探査

以上の議論で，XMLおよびセマンティックWebの空間からモデル$\mathcal{M}(\mathcal{T}^*, \mathcal{V}^*)$を作ることができた。あとは「今の状態から目標の状態に到達するために次に選択する状態のリスト」もしくは「今の状態から目標の状態に到達するためのこれからnステップの間にある状態のリスト」が表現できればよい。以下，開始を表す状態s_{start}，目標を表す状態をs_{target}，とする。このとき，s_{start}，s_{target}のそれぞれの状態を表す述語は$K_{s_{\text{start}}}$，$K_{s_{\text{target}}}$であったことを思い出そう。

まず，「いつかs_{target}に到達する状態（eventually $K_{s_{\text{target}}}$）」をmu式で表すと次のようになる。

$$\psi \overset{\text{def}}{=} \mu Z.\, K_{s_{\text{target}}} \vee [-]\, Z \wedge [-]\, \mathbf{tt}$$

しかし，この式は$\mathcal{T}^* = \{(s_1, a, s_2), (s_2, a, s_1), (s_2, a, s_3)\}$，$s_1 = s_{\text{start}}$，$s_3 = s_{\text{target}}$とすると$s_1 \not\models \psi$となることがわかる。これは$\psi$はループが必ず終了することを強制していることが原因である。これでは自己参照，相互参照を自由に行えるWebでは制約が強すぎるといえる。このため，ここでは「いつかs_{target}に到達する状態が存在する」というように条件を緩めたΦを採用することにする。

$$\Phi \overset{\text{def}}{=} \mu Z.\, K_{s_{\text{target}}} \vee \langle - \rangle Z$$

次に開始状態s_{start}からnステップ遷移した状態の集合を記述することを考える。0ステップは$K_{s_{\text{start}}}$，1ステップ以下は$K_{s_{\text{start}}} \vee [-]\, K_{s_{\text{start}}}$，2ステップ以下は$K_{s_{\text{start}}} \vee$

(6) 逆に言うと、bは外部から隠蔽されている（hidden）ということができる。

$[-]K_{s_{\text{start}}} \vee [-][-]K_{s_{\text{start}}}$ というようになり，一般には $K_{s_{\text{start}}} \vee_{0 \le i \le n} [-]^i Z$ となる。ここで，近似記述子（approximant）を持つmu式で記述すると次のようになる。

$$\mu^n Z.\, K_{s_{\text{start}}} \vee [-] Z$$

近似記述子を持つmu式の意味は，意味関数 $\| \Phi \|^{\tau *}_{\nu *}$ の $\emptyset$ からの適応回数をn回に限ったものとして定義することができる。

これまでの議論から，「今の状態から目標の状態に到達するためのこれからnステップの間にある状態のリスト」（Γ）は次のように記述することができる。

$$\Phi \wedge (\mu^n Z.\, K_{s_{\text{start}}} \vee [-] Z)$$

ここで，$\Phi \stackrel{\text{def}}{=} \mu Z.P \vee \langle - \rangle Z$ であったので，すべて記述すると次のようになる。

$$\Gamma \stackrel{\text{def}}{=} (\mu Y.\, K_{s_{\text{target}}} \vee \langle - \rangle Y) \wedge (\mu^n Z.\, K_{s_{\text{start}}} \vee [-] Z)$$

このmu式Γに対応する状態の集合 $\| \Gamma \|^{\tau *}_{\nu *}$ は前述の一般的なモデル検証アルゴリズム（図3）を利用して機械的に計算可能であり，またタブローに基づいたモデルチェックアルゴリズム（図4）を使用した場合にはmu式の充足可能性をも検証することができる。

5.5 モデルの抽象化

今までの議論で，本論文の目的であるXML上の参照関係に基づいた依存関係をもつ状態遷移システム，すなわちコースの構築が可能になった。また，コース上で目的に通じるパス上の次の状態表現式Γを示した。このmu式Γにモデル構築アルゴリズムを適応することにより，具体的な「次の」状態の集合を求めることが可能である。

このモデルチェックアルゴリズムに基づいた計算量はmu式が不動点演算子（μ, ν）の交代（alternation）が存在しない場合にはモデルの状態数|S|の大きさに比例することが知られている。実際にここで構築したはalternationのない不動点演算子を2つ使うため $2 \times |S|$（$\cong |S|$）に比例した計算量に抑えられ，非常に効率のよい検索を行うことができるといえる。

反面，インターネット上に接続されるコンピュータ数，ユーザ数は急激に増大し，それにつれインターネット上に置かれるWebドキュメントは爆発的に増加している。

このことは，ドキュメントを検索するアルゴリズムが線形の計算量で抑えられるとしても全体の計算量は年々爆発的に増加することを意味している。もちろんCPUのスピード，利用可能な記憶装置の量が状態数の増加と相殺し，計算に必要な時間の総量をおさえることは期待できる。しかし，コンピュータネットワーク上で提供されるサービスが現状の単純なものから，セマンティックWebを用いた複雑なものに進化していくことを考えるとさらなる工夫を行う必要がある。これらのことを考慮し，ここではモデル全体の状態数を減少させるための方法を検討する。

R_T*から$\mathcal{T}^*$を作る過程で依存関係を表現したことを思い出そう。

前の例では$R_T* = \{(u,\star,\star),(\star,l,b),(b,m,c)\}$ から$\mathcal{T}^* = \{(u,l,b),(b,m,c)\}$を作った。このとき，状態$c$は$b$から，$b$は$u$からの遷移しかないことに注目する。このとき，$o \notin \{u,b,c\}$ なる状態oから見た場合に意味を持つのは $\{u,c\}$ のみであることに注目しよう。逆に状態bはあるシステムのコースの内部を表現しており，外部からの参照を意図的に禁止している。このような内部に隠蔽された状態を外部から検索するアルゴリズムが考慮する必要はない。

ここで，モデル$\mathcal{M}(\mathcal{T}^*,\mathcal{V}^*)$ 中の状態の集合S ($S \subseteq S$) を考える。S中の状態でsに関して$t \in S-S$間の状態遷移$s \longrightarrow t$ **or** $t \longrightarrow s$がない状態の集合を隠蔽された状態（hidden states）と呼び$\mathcal{H}_S$と記すことにしよう。さらに$S-\mathcal{H}_S$を公開された状態（exhibited states）と呼び$\mathcal{E}_S$と記述する。ここでつぎの関係を定義する。

$$\forall s,t \in S.\ s \models \mu Z.(K_t \vee \langle - \rangle Z) \Rightarrow s \gg t$$

ここで，新たに定義した関係$\gg$は推移律を満たすことに注意したい。すなわち，$s \gg t$ **and** $t \gg u$ **implies** $s \gg u$が成り立つ。

ここでモデル$\mathcal{M}$から$\mathcal{H}_S$を消去し，Sに関連する状態間の関係を取り去ったのち，関係$\gg$を加えたモデル$\mathcal{M}^-$を作る。このとき，$\mathcal{M}^-$上でsからtへ到達可能なパスが存在する場合にはは定義から$\mathcal{M}$中でも同様である。このため，$\mathcal{M}$上で公開された状態間での到達可能性は抽象化されたモデル$\mathcal{M}^-$での調査で十分である。

この抽象モデルは分割統治（devide and rule）可能である。このため，それぞれのコースを管理しているマシンで抽象化のプロセスを分担することができ，全体のコース探査に必要な状態数を減少させること，すなわち効率のよいネットワーク空間の探査が可能になることを示している。

6 まとめ

本論文では生涯学習，およびそれを支援するe-Learningの現状と問題点について報告した。ここで挙げた問題点には主に，教材作成にかかる費用が膨大であること，教材を社会的に効率よく共有する方法がないこと，カリキュラムを実現する方法などがあった。また，就学者を大学に就学している学生と同様のカリキュラムを提供するのは時間的，金銭的に困難であることも指摘されている。

筆者は，いつでも，どこでも，だれでも就学可能な大学，すなわちユビキタス・ユニバーシティを提案している。このユビキタス・ユニバーシティの実現に向け，本論文ではインターネット上でカリキュラムを提供すること，その具現化の方法について述べた。具体的には，XMLおよびセマンティックWebの技術で提供される教材間の参照関係をセマンティックWebの意味体系であるKIF，および不動点理論を使った時制論理の枠組みの中で記述する方法を示した。さらに，既存のモデルチェック技法がコース内での指針を自動的に与える手段として有効であること示した。そして最後に，コース全体を示すモデルの抽象化を議論し，分割統治を行うことにより検索に必要な全体の状態を減らすことができることを示した。

本論文では細部の証明は省いて紹介してある。また，いくつかの仕組みのうち細部が決まっていないために，動作が不確定になっている部分もある。これらの問題は適宜証明の詳細化とプログラムの実装を行う過程においてさらに洗練していく必要がある。

本論文では形式的なモデルに基づき，かつ動作可能なモデルをアルゴリズムと共に示した。この意味で十分に実用可能である。しかし，生涯学習やユビキタス・ユニバーシティに関するパラダイムは実践の中にその価値が存在する。今後，本システムをさらによいものにし，多数の就学者に実際に役に立つよう実装と普及そして運用へと努力して行きたい。

謝辞

本研究は財団法人生涯学習開発財団の助成を受けています。松田妙子同財団理事長，故角田公正東京大学名誉教授ならびに中村行秀千葉短期大学教授は本研究に対してご理解をいただき，研究計画において適切な指針を示してくださいました。ここに感謝申し上げます。

参考文献

[1]『ユビキタスユニバーシティ』，CUC View & Vision No. 11, 2001 Mar，大矢野潤
[2] http://www.w3c.org/
[3]『eラーニングの国際動向』，IDE現代の高等教育 No. 440　eラーニングの可能性，2002/7，坂元昴
[4]『アメリカのeラーニングの市場』，IDE現代の高等教育 No. 440　eラーニングの可能性, 2002/7，ロバート・ゼムスキー（訳：苑復傑）
[5]『アメリカのeラーニングの事情』，IDE現代の高等教育 No. 440　eラーニングの可能性，2002/7，吉田文
[6]『A Semantics for the Knowledge Interchange Format』, Proceedings of 2001 Workshop on the IEEE Standard Upper Ontology, Aug/2001, Patric Hayes & Christopher Menzel
[7] http://logic.stanford.edu/kif/kif.html
[8]『RDF Model Theory』, http://www.w3.org/TR/rdf-mt/, Sep 2001 (Latest Version), Patrick Hayes
[9]『論理プログラムの基礎』，産業図書，J. W. Lloyd（訳：佐藤雅彦，森下真一）
[10]『Model Checking』, MIT Press, 2000, E. M. Clarke, Orna Grumberg, and Doron Peled
[11]『Modal and Temporal Properties of Processes』, Springer, 2001, Colin Stirling
[12]『Verifying Temporal Properties of Systems』, Birkhäuser, 1992, Julian Charles Bradfield
[13]『Tableau-Based Model Checking in the Propositional Mu-Calculus』, Acta Informatica, 1990, Rance Cleaveland