電子商取引における商品ルールの動的制約代数による検索実験

木村　　塁† 　志賀　隆之†† 　岩井原瑞穂††

† 京都大学工学部情報学科，〒 606-8501 京都市左京区吉田本町
†† 京都大学大学院情報学研究科，〒 606-8501 京都市左京区吉田本町
E-mail: †{rui,tshiga}@db.soc.i.kyoto-u.ac.jp, ††iwaihara@i.kyoto-u.ac.jp

あらまし　インターネット上では電子商取引が盛んに行われているが，そこで用いられている利用条件や割引条件などのルールは自然言語で記述されている事が多い．これらのルールは従来の関係データベースで扱うのは困難であり，プログラムと切り離して記述する事ができないため，複雑なルールを含めた検索が実現されず，利用者がルールを一つ一つ確認する必要がある場合が多い．本稿では，商品ルールの例として格安航空券販売サイトを取り上げ，格安航空券やホテル予約のルールを動的制約条件で記述し，それに対する質問を動的制約代数で記述する．そして処理系で質問処理を行い，利用者に指定すべき条件の提案や，条件による価格変化を含めた検索支援を行える事を示す．

キーワード　DB 言語，e-commerce，問合せ処理

Search Experiment of Business-Rule by Dynamic Constraint Algebra

Rui KIMURA†, Takayuki SHIGA††, and Mizuho IWAIHARA††

† School of Informatics and Mathematical Science, Faculty of Engineering, Kyoto University, Yoshida-Honmachi, Sakyo-ku, Kyoto, 606–8501 Japan
†† Department of Social Informatics Graduate School of Informatics, Kyoto University, Yoshida-Honmachi, Sakyo-ku, Kyoto, 606–8501 Japan
E-mail: †{rui,tshiga}@db.soc.i.kyoto-u.ac.jp, ††iwaihara@i.kyoto-u.ac.jp

Abstract　Although e-commerce is becoming popular, product rules, which describe various conditions such as prices, applicability, and discount conditions are still expressed in natural languages. These rules are difficult to deal with in relational databases because logics are separated from data. There is no system to search on multiple rules, so buyers have to check business rules one by one. In this paper, we take the case of airline tickets and hotel bookings and others as an example, and describe them in dynamic constraint conditions. We also show several queries written in Dynamic Constraint Algebra (DCA), and the result of the Dynamic Constraint Algebra Query Processing Engine can be utilized to present buyers available options, and to check how variables affect prices.

Key words　DB languages，E-commerce，query processing

1. はじめに

電子商取引においても，実世界の取引と同様に，価格以外の複雑な条件が絡み合い価値が決まる商品もある．例えば，会員限定や女性限定などの優遇サービスや，変更やキャンセルの禁止や高い手数料を設定する事で，販売条件を様々な形で設定する事で価格の割引を行い，販売促進が行われている．

また，電子商取引市場の拡大に伴い，利用者が得る事ができる情報量も増加し，一つの商品に対して数十種類もの販売条件・価格条件が提示されることも頻繁に生じるようになってきている．

しかしながら，取引に用いられている利用条件や割引条件などのルールは自然言語で記述されている事が多く，プログラムが理解できるメタな情報で記述される事はまだ数少ない．このため，利用者が一つ一つ価格とルールを照らし合わせて自分にあった商品を見つけることが必要となっており，希望の商品を見つけるための検索支援が必要となってきている．

そこで我々は，ビジネスルールの自動処理に着目し，ルール付商品，つまり商品ごとに多種多様な形式のルールが付され，その適用によって価格などの商品属性が変化するという特徴を持つ商品の電子商取引に対して，そのルールの処理の支援を行う仕組みを研究してきた．

自然言語で書かれたルールを扱うには，機械が処理できる形に変換する必要がある．我々が開発を進められている動的制約

データベースシステム（DCDB: Dynamic Constraint Databases）は，このような問題を扱うためのデータベースモデルであり，動的制約代数（DCA: Dynamic Constraint Algebra）はその質問言語である [1] [2].

従来の関係データベース（RDB: Relational Databases）および Web アプリケーションを用いた方法では，商品の販売条件やビジネスルールなどの複雑なルールはプログラムの中で表現され，あらかじめプログラム内で設定した方法でしか，条件を指定できなかった．このため，複数のアプリケーションでビジネスルールを共有し，利用する事は簡単ではなかった．しかし，DCDB を用いることで，商品ごとに異なるルールを記述する事が可能であり，自然言語で書かれた商品ルールを動的制約表現で記述し，そのルールをデータベースのタプル内に組み込むことが出来る．これによって，複雑な販売条件やルールを，プログラムから切り離して記述することが可能となり，また質問言語である DCA を用いて，動的制約による様々な検索や演算が可能となる．

Web 技術の標準化団体である World Wide Web Consortium（W3C）が標準化を進めている，セマンティック Web のオントロジ言語（OWL: Web Ontology Language [3]）によっても，ビジネスルールなどの制約を表現する試みが行なわれているが，OWL は表現力が非常に高く実用的な処理系が存在していないという問題点がある．

本報告書は以下のような構成になっている．第２章では理論的背景として，CDB，DCDB，DCA について概説する．第３章では，実際に販売されている格安航空券のデータや，それに関係するホテルやレンタカーのビジネスルールの DCDB による表現方法を示す．第４章では，その表現されたデータを利用したアプリケーションを想定し，その実現に必要な質問を DCA で記述する．また，その質問を処理系で実際に実行し，その結果の考察を行う．

2. 動的制約データベースと動的制約代数

2.1 制約データベース

制約データベース (CDB: Constraint Databases) [4] とは，文献 [5] [6] で提唱された RDB の拡張モデルである．RDB を Datalog に基づく述語論理式として見なした時に，この論理式のクラスを，より高い物で置き換えた物に相当する．テーブルの表現に立ち返ると，この拡張によってタプルが“ 属性と属性値 ”という形式に限らず，制約 (constraint) と呼ばれる論理式によってもデータを持てるという特徴で表現される．そして，この制約式を充足する値の組み合わせが，通常の関係のタプルに相当する．CDB では関係のスキーマは RDB と同じ [A_1 , A_2 , $\cdots$, A_k] で与えられるが，そのタプルは [a_1 , a_2 , $\cdots$, a_k]/ ϕ で与えられる．各 a_i が属性 A_i の値になっており，この値は定数あるいは制約変数である．ϕ がタプルにつけられた制約式である．

RDB のタプルは一つの事実を表すのに対し，CDB では充足可能性問題が可解である制約式を満たす定数の組み合わせの集合により，可能な事実の集合を表す．このため，CDB では制約式を満たす組み合わせが無限である場合でも，有限の長さの制約式で表現できることもあり，通常の関係よりも表現力に勝っている．

制約式 ϕ が真であれば，そのタプルが表している情報が正しいとみなす．偽であれば，その情報が存在しないということになる．また，制約式内に存在するいくつかの変数に値が割り当てられていない場合は，タプルの情報が正しいかどうかは不明である．

2.2 動的制約データベース

DCDB とは，前項で述べた CDB の一種であり，動的制約代数 DCA を質問言語として持つ．DCA は，これまでに知られている制約関係代数について変数によるスキーマ定義を行わず，任意の変数が混在する制約式も取り扱えるようにしたものであり，これにより，多様な電子商取引の制約条件を柔軟に扱うことを可能にしている．

現在の実装では制約式は，”制約変数同士 ”，または ”制約変数と定数 ”を等号 (=) とその否定 (≒) で結んだ項 (e.g. x=y,x≒5) をさらに論理演算子である連言 ∧ (and)，選言 ∨ (or)，否定 ¬ (not) ，含意 → (implication) で結合した制約式を用いる．また制約式は全て和積標準形で書くものとする．

DCA の処理系は C++で実装されており，制約関係の入出力は XML 形式を用いている．このため，テーブルで表現された表 1 の制約タプルは，実際の処理では表 2 のような XML 形式にして扱われる．

この XML 形式での表記について簡単に述べる．eq(x,”string”); は等号制約$x=”str” を表わす．eq の代わりに ne と書けば等号の否定を表す事ができる．choice(y,”a”,”b”); は次の和項 ($y=”a”OR$y=”b”) を表す．また，imply(x,y,”a1”->”b1”); は $x=”a1”->$y=”b1” という含意を表わす．

またセミコロン ”; ”により区切られた式の集まりで１つの和項を表わし，セミコロンを AND に置き換えた和積形の制約式が１つの<cond> タグに含まれる．<cond> タグ同志は AND で結合されるものとする．

Model	Price	Buy	Condition
$m1	$p1	$b1	($m1 = 50 ∨ $m1 = 75) ∧ ($m1 = 50 → $p1 = 25)

表 1　制約関係の例

```
<tuple>
<value num="0" atr="Model">$m1</value>
<value num="1" atr="Price">$p1</value>
<value num="2" atr="Buy">$b1</value>
<cond>
eq(m1,"500") eq(m1,"750");
imply(m1,p1,"500","249");
</cond>
</tuple>
```

表 2　XML 形式での動的制約表現

2.3 動的制約代数

DCA では，任意の変数が混在する制約式を用いる事ができ

る．また，あらかじめ出現する変数を限定しないために，変数を陽に指定せずに限量子を与える演算として，動的制約演算という，DCA 固有の演算を新たに導入している．ここでは，関係演算と動的制約演算に分けて，演算の種類ごとに簡単な説明を行う．

2.3.1 集 合 演 算

関係データベースの集合演算を，制約を扱えるように拡張したものである．

和集合演算 二つのリレーションの和集合を取る．

差集合演算 差集合を取る．これは未実装である．

直積集合演算 直積集合を取る．制約式は，元の二つのタプルがもつ制約式の論理積である．

2.3.2 関 係 演 算

選択演算 RDB での選択演算は，テーブルの中から条件に合ったタプルを取り出す演算である．DCDB では和積標準形で書かれている制約に，“ 属性又は変数==属性又は変数又は文字列 ”という等号演算か，同じ形の等号の否定の和項を追加する演算である．

射影演算 RDB と同様に，DCDB の射影も必要な属性を取り出す演算であり，制約は全て残る．

代入演算 assign という文を用いて，属性や変数に文字列を代入することが出来る．

2.3.3 動的制約演算

bool bool(1) で制約が恒真のタプル，bool(0) で制約が恒真でないタプルを選ぶことができる

dependency 変数や属性の依存関係を調べる dep という演算と，２変数間の依存関係が直積かどうかを調べる dep2 という演算が用意されている．

cardinary cardinary 演算は，属性と範囲（有限か無限か，１個か０個か）を指定し，その属性が充足する値の個数に対する制約を記述することが出来る．

vconjunct 要素の値ごとにまとめることで，テーブル内のタプルを全て連結することができる．

3. ビジネスルールの動的制約表現

商品に付随する追加のオプションや，有効期限や納期，購入条件，そして割引や追加料金の設定などを合わせて，ビジネスルールと呼んでいる．

本章では，このビジネスルールを動的制約表現で記述する手法について述べる．ビジネスルールが付加されている商品の例として，レンタカー・格安航空券・ホテル予約などを取り上げ，具体例をあげて説明する．以後は atr で属性名，$var で変数，string で文字列を表す事とする．

また，商品ルールを自動的に抽出し，動的制約表現に自動変換する手法 [7] [8] についても研究が行われているが，本論文では制約の記述における正確性を重視し，手動での変換手法について述べる．

3.1 レンタカー契約条件の動的制約表現

レンタカーの契約条件のデータについては，トヨタレンタカー（rent.toyota.co.jp）や日産レンタカー（nissan-rentacar.com）などの，日本の大手レンタカー会社数社のウェブサイトを参考に作成した．レンタカーに関するビジネスルールは，車種・レンタル時間による価格変化，メンバーカードやクレジットカードによる割引，乗り捨て料金，離島料金や北海道の夏季料金，オプション料金，予約可能期間，キャンセル料金などである．これらを制約の表現方法別に数種類に分類し，制約式に変換する手法について述べる．

カード割引・乗り捨て料金・オプション料金 これらの制約を制約式で表現するために，その価格変化が発生する選択肢を変数におき，さらにその変数による割引率を表す変数も作成する．例えば，カードの種類の変数 $card を作成し，カードの種類による割引率を表す $cutrate という変数を作る．この二つの変数の関係を制約式として表現すればよい．この二つの変数を用いて例をあげると，

($card = ”カード A”)->($cutrate = ”10%”)

といったような含意関係を表す項を，全ての割引・価格変化の組み合わせについて作ることで実現できる．DCDB では制約式を和積標準形で作成する必要があるため，

($card != ”カード A”) OR ($cutrate = ”10%”)

のように和項に直す必要があるが，DCDB では

imply($card , $cutrate , ”カード A”->”10%”);

と書く事によっても，同じ意味を持つ制約式を作ることができる．

車種・レンタル時間による価格変化 各レンタカー会社のサイトには，車種（クラス）とレンタル時間による料金表が用意されている．車のクラスの変数を $class , レンタル時間を $time , それらによって決まる価格を $price とおくと，価格の制約は，

($class=”P1” ∧ $time=”6h”)->($price=”5250”)

といった形で記述できる．このままでは imply を利用しても記述できないので，論理式の等価変換を行い，

($class≠”P1”) ∨ ($time≠”6h”) ∨ ($price=”5250”)

という形に変換できる．これを DCDB で扱える以下のような表記に書き直すことで，制約式として扱う．

ne($class,”P1”) ne($time,”6h”) eq($price,”5250”);

北海道の夏季料金 北海道では夏季は繁忙期のため，夏季料金表として割高な料金を設定し，料金表を別に用意している．つまり，車種とレンタル時間による価格が通常料金と全く異なるので，これを表すためには別にタプルを用意する事で実現する．制約式で記述するときは，“ 利用地が北海道かつ利用始期間が夏季料金適用期間内 ”という制約を持つタプルと，“ 利用地が北海道以外または利用期間が夏季料金適用期間外 ”という制約をもつタプルをつくり，それぞれ別々に価格表の制約を作るという方法を取ればよい．

3.2 格安航空券に付随する条件の動的制約表現

格安航空券のビジネスルールを DCDB で表現する．航空券のデータは格安航空券販売の大手サイトである，アルキカタ・ドット・コム（http://www.arukikata.com/）からデータを得た．表 3 は，実際にアルキカタ・ドット・コムに掲載されている格安航空券データの一例である．

格安航空券の制約条件は，記述方法で以下のように分類さ

利用航空会社	エア・カナダ航空
出発地	成田
訪問可能都市	サンパウロ、サンチャゴ、ブエノスアイレス、 カラカス、サンタフェボゴタ＝ボゴタ、リマ
訪問可能都市数	1
有効期限	6 日〜2ヶ月間 FIX（帰路変更不可）
座席クラス	エコノミー
追加料金	復路カナダ発 1/2-1/3 は成田出発日に関わらず ￥245,000 となります。
ストップオーバー	トロントで一回可、無料。ブエノスアイレス行きは トロント、またはサンチャゴで一回可、無料。
日本国内線	全日空利用．日本航空はお問い合わせください。 * 福岡、伊丹、関西片道￥6,000UP、 * 沖縄片道￥8,000UP
帰国可能空港	関西　￥0UP
リコンファーム	おすすめします
子供料金	（大人料金の 90 %）
コメント	*カナダ乗継の南米行きです。 *有効期間：1 年間有効、片道航空券もございます。 お問い合わせください。

表 3　航空券データの例

```
<tuple>
<cond>
eq(al,”エア・カナダ航空”);
eq(fdp,”成田”);
choice(vc,”サンパウロ”,”サンチャゴ”,”ブエノスアイレス”,”カラカス”,”サンタフェボゴタ＝ボゴタ”,”リマ”);
eq(vcn,”1”);
eq(fof,”FIX”);
eq(vdmin,”6”);
eq(vdmax,”60”);
eq(sc,”エコノミー”);
eq(of,”1”);
imply(bdt,bdtpc,”050102”->”245000”,”050103”->”245000”);
eq(soapc,”0”);
eq(vc,”ブエノスアイレス”) eq(soa,”トロント”);
~eq(vc,”ブエノスアイレス”) choice(soa,”トロント”,”サンチャゴ”);
imply(df,dfpc,”福岡”->”+6000”,”伊丹”->”+6000”,”関西”->”+6000”,”沖縄”->”+8000”);
imply(baa,baapc,”関西”->”+0”);
eq(cppc,”*0.9”);
</cond>
</tuple>
```

表 4　航空券データを DCDB データに変換した例

れる．

出発地・利用航空会社など　これらは，一つの格安航空券につき，それぞれ一つの値が定まるため，変数と等号制約を用いて制約式にする．

日本国内線　規定の出発地と異なる空港から出発し国内線利用を利用する場合には，空港ごとに異なる追加料金が発生する．これを imply を用いて記述する．

訪問可能都市・都市数　訪問都市によって追加料金が変化したり, 訪問都市数によって追加料金が発生したりする.「訪問都市数が何都市以上で追加料金が何円必要である」などのルールを制約式で記述する．

日程・曜日　週末料金や正月料金など，出発日・出発曜日・復路到着日・復路到着曜日によって追加料金が必要な場合がある．また，出発地との組み合わせで追加料金が変わる事が多く，これらを制約式で表す．

ストップオーバー　訪問する都市によってストップオーバーが可能な都市が変化したり，可能な期間などが制約条件としてあるので，これらを制約式で記述する．

購入条件　子供料金の適用条件や割引率，学生料金の適用年齢や必要物，発券期限などの制約がある．

変換する際の基本的な方針としては，前節で詳しく述べたように，項目とそれに伴う価格変化の値をそれぞれ変数におき，その変数間の関係を動的制約表現で記述した．こうして表 3 のデータを DCDB の制約に変換し，制約関係を XML 表記にしたものが表 4 である．

3.3　ホテル予約における契約条件の動的制約表現

ホテルのウェブ予約におけるビジネスルールの動的制約表現について考える．ホテルの種類によって制約の種類も変化するが，大きく分けてビジネスホテルの予約と，シティホテル・リゾートホテル・旅館の予約に二分される．

ビジネスホテルの予約サイト　ホテルとその場所，食事の有無，部屋の設備，宿泊日とそれによって変動する価格などが制約条件として考えられる．しかしビジネスホテルには，部屋（ベッド）の種類もシングルとツインだけなど，種類が少ない所が多く，また，設備や食事などは全て有無で済まされてしまうので，既存のデータベースシステムでも十分扱う事ができる．

その他のホテルの予約サイト　ビジネスホテル以外の予約サイトでは，いくつかの宿泊プランを用意する形をとっており，利用者はそのプランから自分の条件にあったものを選ぶ形をとっている．この条件を構成するものは，部屋のタイプ（何人まで宿泊可能・ベッドの種類・スイートルームなど），朝夕の食事の有無，食事のメニュー，部屋の露天風呂の有無，レディースプラン，連泊プランなどである．

これらを動的制約表現に直すのは比較的簡単である．プランによって部屋のタイプ・食事・利用者数・利用者の性別などを変数におく事で制約を作成し，後は宿泊日による価格表を元に imply を使って指定してやればよい．食事のメニューも制約として表すことで，プランを選ぶために利用者が何を指定する必要があるかを示すことができる．

3.4　動的制約表現に変換不能な条件に関する考察

３節にわたり，レンタカー・格安航空券・ホテル予約のビジネスルールを動的制約表現に変換する手法について述べたが，ビジネスルールの中には変換できなかったルールも存在した．第４節では，変換できなかったルールの例をあげ，動的制約表現で記述できなかった理由と，その制約を記述するために必要な DCDB 処理系の機能について考察する．

範囲を含んだ制約条件　格安航空券やホテル予約の購入条件に

含まれる学生料金の適用年齢については,「学生（13 歳〜27 歳）」といったように年齢の範囲で記述されている．しかし，DCDB の現在の実装では不等号を用いた制約を記述する事ができないため，プログラム側で制約を記述する必要がある．
制約を充足するかどうかについては，できる限りプログラム側で判断することを減らし，DCDB の処理系で扱えるようにすべきである．こういった範囲に含んだ制約条件は，航空券の購入条件に限らず，他のビジネスルールでも多く見られる条件であるため，今後実装していく必要がある．

四則演算を含んだ制約条件　格安航空券のチケットの発券期限に関する制約には,「発券期限：出発から 35 日前」といったように，購入可能な期間に関する制約条件が存在する．これは，今日の日付を $today，往路出発日を $fdt と変数においたとき，$today + 35 ≦ $fdt という制約条件が成り立つことになる．しかし，現段階では変数に値を保持することができるだけで，この制約条件はプログラム内に記述する必要がある．こういった制約条件を記述することができれば，制約条件の表現の幅が広がり，制約条件に自動処理を進めることができる．
また，この例では日付の計算も発生するため，その対応も必要であるが，既存の関係データベースの多くで日付計算のサポートがされており，ビジネスルールで日時の制約条件が頻繁に利用されることを考慮すると，実装をする必要もある．価格体系の制約条件なども,
合計＝ (基本価格＋追加料金 1 ＋…)× 割引率
といったような四則演算を用いた式で表現されているが，これをプログラムから切り離して記述するためにも，四則演算を処理できるエンジンにする必要がある．

4.　動的制約代数を用いた検索実験

電子商取引で実際に利用されるアプリケーションを想定し，DCDB を用いることで可能となる検索支援の例を示す．また，航空券・ホテル・レンタカーの予約にまつわるデータに対して，我々が開発した DCA の処理系で質問を実行し，その結果の考察を行う．

4.1　買い手と売り手の制約の照合による検索手法

この節では，制約条件を自動的に処理することで可能となる検索手法について，格安航空券販売アプリケーションを例に用いて，実験を行う．

4.1.1　格安航空券販売の現状

インターネット上で販売される航空券は，大きく分けると正規割引航空券と格安航空券の二種類に分類される．前者は，各航空会社の公式サイトで販売を行っており，その中で各社が料金形態を明示し，航空会社でも旅行会社でも同じ条件・価格で購入可能である．一方で後者は，同じチケットに対してもそれぞれの旅行会社が異なる制約条件を設定し，その制約の量や種類によって価格が変化している．また，その価格変化も旅行会社の独断で決定されるため，同じ制約条件でも価格が異なっているチケットや，異なる文章で書いているが指し示す意味は同じであるチケットが存在し，正規割引航空券と比較して，より複雑な価格体系となっている．

このため，希望の格安航空券を見つけるためには，利用者が予め決めている条件と販売されるチケットの制約を照合し，マッチングする必要がある．しかし，既存の格安航空券検索サイトでは，こういった機能を備えた検索は実現されておらず，以下のような流れでチケットの検索を行うのが一般的である．

（1）　出発月（出発日）や帰国日，そして出発地と目的地の指定を行う．

（2）　価格順にチケットの名称一覧と価格帯が表示される．ただ，ここに表示される価格では各種の価格変更が考慮されていない．

（3）　利用者はチケットの詳細ページを見て，自然言語で記述された制約を一つ一つ見て確認し，自分の条件にあったチケットを探す．

この中で３番目の作業は非常に煩雑な作業であり，チケットの候補が多ければ多いほど，またチケットに付随するビジネスルールの量が多ければ多いほど，絞り込むために時間がかかってしまう．このビジネスルールを動的制約表現に置き換えることで，ルールを論理的に処理することを可能にし，利用者が選択肢の中から最適なチケットを検索するのを支援できるようにする．

4.1.2　ビジネスルールの検索実験

今回用いた格安航空券のデータは，第３章で紹介したアルキカタ・ドット・コムのデータである．作成した航空券データの数は，日本からロサンゼルスへの往復便を中心とした 40 件である．このデータを変換し，動的制約データベースの制約式についてまとめたのが表 5 である．１つのタプルで１つの航空券を表現している．

タプル数	和項の数	変数の数	定数の数
40	1666	734	2878

表 5　航空券データ

条件の代入による絞込み　まず，従来の格安航空券販売サイトと同じように，出発日や目的地，出発地の入力を行い絞込みを行う．DCA では，条件を入力し変数又は属性に指定した値を代入する場合，assign 文を用いる．作成した制約データでは $vc を訪問可能都市が入る変数にしているので，目的地をロサンゼルスに設定する場合は,
assign(vc,”ロサンゼルス”);
という質問文を用いて代入する．出発地や出発日なども同様に代入し，候補を絞る．

また，ストップオーバーをどの空港でしたいかどうか決まっている場合や，二カ国以上訪問都市がある場合など，それ以外に明らかな条件がある場合も，同じように変数に代入する事で検索を行う事ができる．

選択肢が複数ある商品の検索　利用者が商品を探す際に，選択肢が複数あるものに対して価値を見出すことがある．例えば，支払い方法が指定されている場合，選択肢が複数ある方が便利で価値があると考える．こうした時に，選択肢が複数ある商品だけ検索で見つけることができれば便利である．
DCDB では，cardinary 演算を用いることにより，変数と範囲を

指定し，その変数が充足する値の個数に対する制約を記述することが出来る．つまり，充足する値の個数の制約を用いた質問文を作成することができる．これを用いる事で,「目的地を複数都市から選べる航空券（訪問可能都市が複数ある航空券）」や,「復路運行曜日が複数ある航空券」などを検索する事が可能となる．前者の方は,

card($vc,gt1);

という問い合わせ（|$vc|>1）により検索を行う．後者の質問も同様に，復路運行曜日の変数を用いることで検索を行うことができる．

価格変化の依存関係をもとにした検索　価格に関する制約条件は，利用者が特に意識するルールである．利用者は価格変化，特に追加料金が発生する条件を考える必要があり，自分が選んだ選択肢に対して追加料金が発生しないものだけを探しだす事で，検索支援が行えると言える．

DCA には，変数や属性の依存関係を調べる dep という演算と，２変数間の依存関係が直積かどうかを調べる dep2 という演算が用意されている．例えば,「帰国便の出発曜日が日曜日の時に価格変化のない物のうち，ストップオーバーが可能なもの」という複雑な質問を考えたときには，次の質問文で検索することができる．

assign($bdw=”日”);

dep($bdwpc,0);

dep($soa,1);

ここで用いた$bdw は帰国便出発曜日の属性，$bdwpc は帰国便の出発曜日による価格変化の変数，$soa はストップオーバーが可能な都市が入る変数である．

利用者の定めた条件で制約が恒真となるものを検索　商品に非常に多くの制約式が与えられている場合，利用者がこれまで決めた条件で，全ての制約条件を満たしているかどうかを知りたい場合がある．また，条件を全て満たした商品だけピックアップしたい場合がある．

Boolean 演算を用いる事で，制約の充足性を判定できるので，現在与えた条件で制約が恒真かどうかを判断することができる．bool(1) で制約が恒真のタプル，bool(0) で制約が恒真でないタプルを選ぶことができる．

例えば,「ユーザーがこれまで指定した条件で，制約条件を全て満たすことが確定した商品」を見つけるためには bool(1) を用いる．また,「何らかの制約がまだ与えられている商品」を見つけるためには bool(0) を用いる．これを使い，ユーザーインターフェースの方で，上部に確定したチケット一覧を，その下にまだ確定していないチケット一覧を同時に表示し，利用者の支援を行うこともできる．

複雑な制約条件の記述を用いた検索　旅行のプランなどを立てる際に，利用者は航空券を調べながら，徐々に自分の条件を厳密なものにしていく．チケットを探す初期段階では，目的地を複数の候補から決めかねていたり，出発日や帰国日についても複数の候補を考えたりしていることも大いにありうる．

DCA の結合演算を用いる事で，簡単にこの複雑な条件を含む質問を作成することができる．ユーザーの制約を別に用意したテーブル内に記述し，そのテーブルと航空券データが入ったテーブルとの直積をとることで，制約の論理積を取り，条件を満たすタプル，つまり所望の航空券データだけを取り出すことができる．

例えば,「目的地はロサンゼルスかニューヨークで，1 月 10 日か 11 日に出発」という条件を満たす航空券は，表 6 の制約を持つテーブルと航空券データのテーブルとの直積を取る事で簡単に検索する事ができる．

```
<cond>
choice(vc,"ロサンゼルス","ニューヨーク");
choice(fdt,"050110","050111");
</cond>
```

表 6　利用者の条件を制約にしたもの

4.1.3　検索結果に対する考察

前項で作成した質問文を実際に実行し，その結果を考察する．以下の４種類の質問文に対し処理を行った．実行環境は，CPU:PentiumM 1.2GHz，Memory:504MB で cygwin 上という環境で実験を行った．表 7 は，実行時間（３回実行した平均）と出力結果のタプル数を示している．

（1）　assign(vc,”ロサンゼルス”);

（2）　assign(vc,”ロサンゼルス”);
　　card($vc,gt1);

（3）　assign(vc,”ロサンゼルス”);
　　assign($bdw=”日”);
　　dep($bdwpc,0);
　　dep($soa,1);

（4）　表 6 の制約を持つテーブルを結合

質問	実行時間（秒）	タプル数	和項の数	変数の数	定数の数
1	0.223	26	391	481	919
2	0.384	18	285	354	669
3	0.329	3	37	45	64
4	0.306	31	538	659	1348

表 7　質問文を処理した結果とその実行時間

まず，実行時間については，全ての質問が 0.3 秒程度の実行時間内で計算されているため，実際にアプリケーションで用いる場合を考えても，十分早い時間で結果を得る事ができると言える．

次に，出力されたタプル数についてであるが，従来の格安航空券検索サイトでは，目的地を設定する質問１を処理し，結果に残されたタプル数と同じだけの航空券の候補について，自分にあったものを一つ一つ探す必要があった．これに対し動的制約データベースでは，さらなる条件を設定する質問２や質問３を行うことで，航空券を効率的に絞り込むことができると，実行結果から言える．

また，和項の数と定数の数については，処理系に適用することで制約式が最適化されるため，大幅に減っていることがわかる．

質問４については，曖昧な条件しか思いつかない利用者に

とっても，柔軟な制約の記述により航空券を検索する事で，航空券の候補を減らす事ができるという結果が出た．これによって，従来は検索条件を絞り込むことさえできなかった段階でも，利用者が自由に制約条件を与えることで，検索支援が行えるようになると言える．

4.2 対話的なビジネスルールの検索手法

インターネット上での格安航空券検索は手間がかかる一方，旅行会社の店頭での格安航空券販売では，旅行会社の販売員と対話しながら条件を絞り込んでいく事ができる．利用者はこの対話を通して，質問に答える事で自分の条件を確定する．そして，多くのチケットの候補の中から，利用者の条件にマッチするものを販売員が提示してくれる．

そこで，インターネット上での取引でも，店頭でのやり取りと同じように対話的に質問することで，目的の航空券を探せるようにするという利用者支援の方法を考え，このアプリケーションを実現するために DCDB を用いた例を示す．

出力された XML を元に指定すべき項目を読み取る　DCDB の制約式は<cond>と</cond>の二つのタグで囲まれた範囲に記述される必要があるが，DCA の処理系で処理された制約は，以下の表 8 ように cond タグ内の schema 属性に，条件式を構成する変数の集合が入るように整形される．このため，cond タグを解析することで，制約式内にどの変数が使われているかを，ユーザーインタフェースのプログラム側だけでも判断する事が可能である．つまり，利用者が指定すべき条件を判断し，それを指定するように利用者に促す事が可能となる．

チケットを確定するために必要な条件をプログラム側が提示してやることにより，スムーズに航空券の検索を進めることができる．

```
<cond schema="bdw bdwpc">
~choice(bdw,"金","土") eq(bdwpc,"+5000");
</cond>
<cond schema="vdmin">
eq(vdmin,"5");
</cond>
```

表 8　DCA の処理系から出力された制約条件

価格変化に関連する変数　利用者は，航空券チケット購入の際に価格を判断基準にする事が多い．従来の航空券検索でも指定した出発日の基準価格順に並ぶが，これは各種追加料金などを考慮しておらず，帰国日や日本国内線などを設定するだけで価格が大幅に変化する場合もありうる．

本実験では，航空券データを制約に直す際に，価格変化量を示す変数を全て，最後が ”pc ”で終わる変数名にした．例えば，ストップオーバー可能な都市名が入る変数を$soa とし，ストップオーバー都市による価格変化量を示す変数は$soapc とした．

上記の方法で cond タグを解析し，pc で終わる名前の変数が入っている制約文を探し出し，候補となっているチケットのタプルの中でどの変数が一番多く使われているかを調べる．そして，その項目に対する条件の入力を促す．これを繰り返すことで，チケットの価格の決定を対話的に進めていくことができる．

4.3 複数サイトのビジネスルールの結合手法

商取引においては，複数の異なる商品を売ることで割引をしたり，景品をつけたりして販売促進を行う．アメリカでは，Expedia Travel(www.expedia.com) や Travelocity(www.travelocity.com) などの航空券販売サイトで，航空券とホテルやレンタカーなどを同時に予約・販売し，割引やホテルの部屋のグレードアップなどの特典を使って販売促進を行っている．

日本では，インターネットを使ったホテルの宿泊予約も航空券チケット予約も，それぞれ共に普及し，利用されることも多くなってきているが，ホテルと航空券を同時に予約することでインセンティブを出し，割引を行っているサイトはほとんどなく，大手サイトと言われるような大規模なサイトではまだ導入されていない．

関係データベースのシステムを使ったサイトでは，新たに同時予約のシステムを追加することは，プログラムの全面的な修正が必要となるため，導入することはなかなか難しい．DCDB を用いることで，それぞれ個別の予約・販売データを利用する事ができ，同時予約による割引を追加するアプリケーションが，容易に構築可能である事を示していく．

4.3.1 ビジネスルールの結合実験

DCDB では，変数や属性の意味を共有する事で，複数のアプリケーションで同じデータを共有し，異なる用途で利用できるという特徴がある．これにより，航空券販売サイトで利用されている航空券のデータと，ホテルの宿泊予約サイトで利用されているホテルのデータをそのまま用いて，同時予約のシステムを構築することができる．この事を，Expedia Travel で行われている事例を参考に，データと質問をいくつか作成し，実際に処理系で処理させてみる．結合実験に用いたデータに関しては表 9 にまとめる．

商品	タプル数	和項の数	変数の数	定数の数
格安航空券	40	1666	734	2878
ホテル	10	91	83	193
同時予約割引	15	95	218	269

表 9　結合実験に用いたデータ

航空券のデータ　Expedia Travel で用いられている航空券データを利用せず，前節で作成した格安航空券のデータを用いる．

ホテル予約の制約データ　ホテル予約の制約は，Expedia Travel のホテル予約で実際に用いられているデータを参考に作成した．ホテルの価格は部屋の種類と宿泊日の日付によって決定される．

同時予約による割引の制約データ　割引きの制約データは，Expedia Travel の割引方法を参考に作成した．表 10 のような制約である．割引名・割引価格・ホテル名を属性とした．Expedia Travel での割引価格に関する制約は，航空会社・ホテル名・宿泊期間によって決定されていたので，それを元に制約式を作成している．この表 10 で使用されている $days は宿泊期間の変数， $bdwpc は復路運行曜日，つまりチェックアウトの曜日による航空券の価格変化の変数である．この例では，二日以上の宿泊日数で，ロサンゼルスが目的地の一つ，またはロサンゼルスでストップオーバーする場合に，宿泊期間とチェックアウト

の曜日に応じて割引価格を決めている．
この例の他にも，航空会社によって割引率が異なるホテルや，部屋の種類によって割引の有無があるホテルなどの制約を作成した．

```
<tuple>
<value num="0" atr="bt_name">book1</value>
<value num="1" atr="bt_price_change">$btpc</value>
<value num="2" atr="book_hotel">LAX Plaza</value>
<cond>
eq(al,”エア・カナダ航空”);
eq(vc,”ロサンゼルス”) eq(soa,”ロサンゼルス”);
ne(days,”1”);
~choice(days,”2”,”3”) eq(bdwpc,”+0”) eq(btpc,”-150”);
~choice(days,”2”,”3”)  eq(bdwpc,”+0”) eq(btpc,”-100”);
~choice(days,”4”,”5”) eq(bdwpc,”+0”) eq(btpc,”-140”);
~choice(days,”4”,”5”)  eq(bdwpc,”+0”) eq(btpc,”-90”);
</cond>
</tuple>
```

表 10　同時予約による割引の制約

割引制約を結合する　航空券に対する条件として，assign 文を用いて出発地を関西空港に，目的地をロサンゼルスに設定する．さらに，割引を得るために，航空会社が未定の航空券を除外する．そして，航空券のテーブルと同時予約割引のテーブルを join 文で結合する．次に，宿泊の条件として，assign 文を用いて木曜から土曜まで二日間の宿泊し，ベッドはキングベッドという条件を設定する．これらの条件をまとめて結合する．

4.3.2 考　　察

目的地と出発地と航空会社の指定をし，同時予約割引の制約を結合した段階でのタプル数は 34 となった．さらにホテルのテーブルを結合し，宿泊に対する条件を指定した段階でのタプル数は 12 となった．結合の質問を処理する時間は 1.063 秒であった．多くの制約文を結合した場合でも，有用な実行時間内で動的制約の処理を終えることができた．また，出力される結果にも不都合もなく，満足のいく結果を得る事ができた．

今回の実験では，自然結合によるテーブルの巨大化を防ぐために，事前に航空券に対する制約条件を与えている．この制約条件を先に与えずに結合を行うと，航空券とホテルのテーブルの結合だけで，最大で 600 ものタプルを生成することになる．DCDB の処理系では，タプル間の制約文がない場合，タプルの数と実行時間が比例することが知られているので，実行時間を短く抑えるためにも，結合の前に制約条件を設定し，タプル数を減らす必要がある．この事は，実際にアプリケーションとして利用するときには必ず考慮すべき点である．

結合文の有用性としては，それぞれの商品の制約が残ることにより，複数の商品の制約条件を合わせて表示したり，キャンセルポリシーや割引条件だけを表に出力することも可能となる．また，様々な商品の制約条件を用意しておき，それらを組み合わせることで，可能な組み合わせを求めるなど，プラニングが行なえる．

ところで日本では，旅行産業全体で企業間取引を電子化するときに利用する標準規格として，TravelXML という標準規格の開発を進められている．DCDB では，TravelXML のように全ての用途に関して属性を決めておく必要がなく，変数を自由に設定することができ，変数同士の関係も制約として記述できるので，制約条件の記述に関する自由度が高く，システムの構築も簡単であるというという点で優れている．

5.　まとめと今後の課題

本稿では，実際に電子商取引を行っているウェブサイトを対象にし，そこで用いられているビジネスルールを動的制約表現に変換する事で，複雑なビジネスルールの記述における動的制約データベースの有効性を示した．そして，その中でも最も複雑なビジネスルールをもつ格安航空券販売に対し，データベースシステム用のデータを作成し，動的制約代数を用いることで可能となる航空券検索支援の例やその質問例を示した．さらに，格安航空券販売とホテル予約のビジネスルールを結合する事で，簡単に複合商品のビジネスルールを作成し，そこに割引の制約条件を結合させることで，動的制約データベースを用いた複合商品の販売システムの構築方法を示せた．

また，ビジネスルールの記述において動的制約表現に変換できないルールをまとめ，それを表現するために実装が必要な，動的制約データベースの機能について考察することができた．

本研究では旅行分野に関する電子商取引を主に扱ったが，今後はさらに多くの分野のビジネスルールを実験として扱うことで，動的制約データベースに適用することで検索の支援が行えるかどうかを調べる必要がある．また，動的制約表現の記述力に関して，データを取ることでさらに詳しく分析し，今後，動的制約データベースに実装すべき機能と仕様について検討していく必要がある．

文　　献

[1] Iwaihara,M:Supporting Dynamic Constraints for Commerce Negotiations，2nd Int. Workshop in Advanced Issues of E-commerce and Web-Information Systems (WECWIS)，IEEE Press，pp.12-20, June (2000).

[2] Iwaihara,M:Matching and Deriving Dynamic Constraints for E-Commerce Negotiations, Workshop on Technologies for EServices,San Francisco, California，2000.

[3] Web Ontology Language OWL / W3C Semantic Web Activity:http://www.w3.org/2004/OWL/

[4] Kuper, G. , Libkin L. and Paredaens, J. (Eds.):Constraint Databases (2000)

[5] Kanellakis, P. C., Kuper, G. M. and Revesz, P. Z.: Constraint query languages, Proceedings of the 9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS ’90), ACM Press, pp.299–313 (1990).

[6] Kanellakis, P. C., Kuper, G. M. and Revesz, P. Z.: Constraint query languages, Journal of Computer and System Sciences, Vol. 51(1), pp. 26–52(1995).

[7] Iwaihara, M., Shiga, T. and Kozawa, M.: Extracting Business Rules from Web Product Descriptions, Proc. 5th Int. Conf. on Web Information Systems Engineering (WISE2004), LNCS3306, pp. 135–146 (2004).

[8] 小澤正幸, 岩井原瑞穂, 上林彌彦: 電子商取引における商品ルール抽出支援手法, 第 15 回データ工学ワークショップ (DEWS2004) 論文集, ISSN 1347–4413.