対話的制約充足手法による制約プログラミングの実現*

3N－1

石井 恵　佐々木 裕　金田 重郎
NTT情報通信網研究所

1 はじめに

世の中で行なわれている事務処理は、コンピュータ化により、コストや時間の削減の面で大きな効果が期待できる。なぜなら、健康保険規則や社員給与規則等、事務処理が従うべき事務規則が明確に規定されているからである。そして、事務処理で最も要求されることは、処理要求に対して常に事務規則と整合した処理結果を生成することである。よって、事務規則をデータ間の関係を規定する制約としてとらえ、事務処理は事務処理上のデータ状態を、常に事務規則に関して整合した状態に維持管理する整合性管理である、という立場で事務処理を実現することを考える。

従来、事務処理システム構築にあたっては、多くの場合、手続きフロー、ルール、トリガといった状態を遷移させる操作として、事務規則を記述していた。しかし、規則と整合した処理結果を生成する操作のみを可不足なく記述することは困難と考えられる。

そこで、我々は整合性の仕様である事務規則そのものを制約として表現し、制約の記述のみで処理を実現する手法を提案する。本手法では制約の充足により、処理を実現するので、事務規則と整合した処理結果の生成が保証される。但し、制約とデータの状態から制約の充足ができない場合、ユーザとの対話により、データを制約充足状態に推移させる。このとき、ユーザとの対話、すなわち、充足に必要なデータの値の質問がユーザに負担にならないように、質問数を抑えるための幾つかのヒューリスティックスを導入する。

以下、2章では、提案する整合性管理システムを説明し、3章ではシステムの評価結果を述べる。4章はまとめである。

2 整合性管理システム

2.1 制約表現

事務処理を規定する事務規則の制約表現について考える。事務処理では、処理要求に対する全ての規則の正しい適用により、規則と整合した処理結果が得られる。よって、規則を制約としてとらえ、規則が許すデータ状態として表現する。通常、規則が許すデータ状態は規則を構成する条文に対応する。なぜなら、正しく規則が適用された場合、処理結果は、規則中の条文のいずれかを満足するからである。以下に制約表現のシンタックスを説明する。

規則が許すデータ状態を表したものを制約素と呼ぶ。各制約は、1つ以上の制約素から構成される。制約素中には制約変数と呼ばれる変数を記述でき、制約変数は各制約素内でのみ有効な変数である。

事務処理で扱うデータ項目名をユニットとし、以下、ユニット、制約、制約素、パタン、テスト節の定義を示す。

表1 真偽値

種別	ユニフィケーション，関数評価結果	真偽値
パタン	ユニフィケーション成功	真
	ユニフィケーション失敗	偽
テスト節	関数の返却値が真	真
	関数の返却値が偽	偽
	関数内に定数とユニフィケーションしていない制約変数が存在	未定
制約素	パタン、テスト節が全て真，かつ，制約素が参照するユニットの値は全て定数	真
	少なくとも1つのパタン，または，テスト節が偽	偽
	その他	未定
制約	制約内で唯一の制約素が真であり，他の制約素は偽	真
	制約内の制約素が全て偽	偽
	その他	未定

1．ユニット

unit(ユニット名，　値).

各ユニットの値を定義する。ユニット名はユニークであり、値は定数、定数のリスト、または、変数を書く。値が変数の場合、ユニットは値がアンバインドであることを示す。

2．制約

constraint(制約名，　(制約素名1，　制約素名2，　…).

制約名はユニークであり、制約素名は次の element で定義される制約素の名前である。制約条件を重複なく記述するため、3章で述べる制約充足エンジンによって計算される真偽値を用いて、制約素相互に以下の関係を設ける。

[制約素の排他性]

同一制約内の制約素は、1つの制約素が真となる時は、残りの制約素は偽となる。

3．制約素

element(制約素名，　(pattern(ユニット名1, 値1)，　…，
TEST(Boolean 関数1)，　…).

制約素名はユニークである。pattern は各ユニットのとるべき値を表し、パタンと呼ばる。値には、定数、定数のリスト、または、制約変数を書く。TEST は、テスト節と呼ばれ，ユニットが満たすべき Boolean 関数条件を表す。テスト節で評価される関数の引数は、定数、定数のリスト、または、そのテスト節と同じ制約素内の制約変数である。

2.2 対話的制約充足エンジン

本エンジンは充足されていない全ての制約を、ユーザとの対話を交えながら、充足することにより事務処理を実現する。制約が充足されているか否かを判定するマッチングの結果は、ユニフィケーションと関数評価によって決定され、表

*A Constraint Solver with an Interactive Satisfaction Mechanism
Megumi ISHII, Yutaka SASAKI and Shigeo KANEDA
NTT Network Information Systems Laboratories

1に示す真偽値で表される。ユニットには、充足過程でユーザ処理要求が反映されていることを表す *decided* と、ユーザ処理要求を反映するために、値の変更が行なわれる可能性のあることを表す *default* の2値をもつ強度を設定する。本エンジンでは、対話によるユーザへの負荷を抑えるために、ユーザへの質問回数を抑えるヒューリスティックスを用いている。以下、エンジンの動作を説明する。但し、下線が引いてあるステップは、ヒューリスティックスを使用している。

処理系の動作

(1)STEP1: 処理要求の入力 (処理開始)

全てのユニットの強度を *default* にする。入力されたユニット値を変更し、そのユニットの強度を *decided* とする。

(2)STEP2: 処理終了チェック

全ての制約の真偽値が真であれば、処理を終了する。

(3)STEP3: 非充足な制約の選択

真偽値が偽、または未定の制約を1つ選択する。但し、強度が *default* のユニットの値を変数と仮定した場合、1つの制約素のみが未定となり、その制約素が参照する強度が *default* のユニットの値がユニフィケーションにより全て定数に決定される非充足の制約が存在する場合、その制約を優先する。

(4)STEP4: default 強度のユニットの値を削除

強度が *deafult* のユニットの値は、処理要求を必ずしも反映していないため、ユニフィケーションによりユニットの値を決定するには、値を変数化する必要がある。しかし、値を変数化すると、ユニフィケーションによって値が決定できない場合、質問候補となる。そこで、変数化するユニット数を抑えるため、真となりうる制約素の決定に影響するユニットのみを変数化する。具体的には、選択された制約中の制約素で参照され、強度が *default* のユニットのうち、以下の条件を満たすユニットを変数化する。

(条件) 選択された制約中の制約素で参照される強度が *default* のユニットの値を全て変数と仮定した場合、(a) 未定となる制約素のテスト節で参照される制約変数とユニフィケーションされるユニット、及び (b) 制約素間で異なる値でユニフィケーションされるユニット。

(5)STEP5: 制約が充足されたかをチェック

選択した制約中の制約素が、1つのみ未定、残りは偽となり、かつユニフィケーションによって、未定の制約素が参照するユニットの値が全て定数に決定されるなら、STEP7へ。そうでなければ、STEP6へ。

(6)STEP6: ユーザへユニット値を質問

質問数を抑えるため、選択した制約中の真偽値「未定」の制約素から参照されている変数状態のユニットの中から、充足状態を決定するのに影響力の大きい順に、すなわち、テスト節で参照されるユニット、制約素間で異なる値が宣言されているユニット、それ以外のユニットの順で選択し、そのユニットの値をユーザに質問する。質問したユニットに入力された値を割り付け、強度を *decided* にする。STEP5へ。

(7)STEP7: ユニット値の変更

選択された制約に関して，真偽値が「未定」である制約素とユニット状態のユニフィケーションを行い、ユニフィケーション前後で値が変数から定数に変化するユニットの値を、ユニフィケーションによって決定された値に変更し、強度を *decided* にする。STEP2へ。

本エンジンでは、変更すべきユニットの値を、処理要求と制約にもとづく各制約の局所的な充足により決定する。この時の最大の特徴は、ユーザ処理要求と制約から変更すべきユニットの値が決定できない場合、ユーザへユニットの値を質問することにより、局所的な充足を実現している点である。これは事務処理では処理要求に対する処理結果が、必ず一意に決定され、また、ユーザに質問されたユニットの値は、処理要求の情報の不足を補う情報であり、ユーザが容易に質問に答えられることを反映している。制約を局所的に充足することにより、制約処理で問題となる多重世界の管理が不要となり、実用的なメモリ量・処理時間で処理が可能となる。

表2 質問回数

処理例／手法	既存システム	本手法
独身者の転入	59	57
単身赴任者の転入	122	130
結婚	46	40
出産	27	26

3 評価

本処理系の有用性を実証するため、事務処理の代表的なタスクである帳票処理を行なう総務業務支援システムＫＯＡ[中野 92]の主要部である帳票作成部を制約を使って記述し、その動作の評価を行なった。ＫＯＡは一部ルールにより記述されているが、基本的には業務担当者からのヒアリングに基き、手続き型記述により表現されている。対象帳票は30種類、対象データ項目は実に900項目に及ぶ。

評価結果

[制約による記述容易性]

帳票作成部分は、制約数約８０個、制約素数約２７０個で記述でき、記述量(行数)は従来に対し２分の１程度に削減された。９０％以上の制約が４個以下の制約素で表された。高々４つ程度の排他関係のチェックは、人間にとって容易である。また、参照される制約数が３個以下のユニットは全体の８０％以上を占めた。このことは、事務処理をモジュラリティ高く記述出来たことを表す。よって、制約表現した結果、事務規則の変更による保守・開発の容易化が期待できる。

[処理系の動作性能]

表２は主要４処理におけるユーザへの質問回数の比較結果を表す。従来の手続き型システムにおいても，業務規則から決定できない場合は，ユーザへの質問を必要とした。本手法によってユーザに質問された質問数は従来システムと同等であり、手続き記述を無くしたことによるユーザへの負担の増加はなかった。

4 まとめ

事務処理を整合性管理ととらえ、その制約表現と対話的制約充足エンジンを開発した。さらに、我々の手法の有用性を実証するため、実用システムＫＯＡの帳票作成部分を制約表現により記述し、記述量の削減、モジュラリティの高さ、手続き記述なしで従来と同等の質問数で処理が実現できることを確認した。

参考文献

[中野 92] 中野: 容易に知識修正ができるオフィス業務エキスパートシステム, 電子情報通信学会, 人工知能と知識処理研究会, AI91-80,pp.49-56(1992).