

単方向性制約伝搬に基づく業務システム構築法の提案

桑山浩希 中西義尚 金田重郎

同志社大学大学院理工学研究科

1. はじめに

一般の業務システムでは、宣言的な業務規則を、手続的にプログラム化しているため、業務規則・プログラム間のセマンティックスギャップが大きい。この問題を解決するため、条例を制約として表現する手法が提案されている[1]。なお条例の集合を業務規則とする。しかし、データ伝搬が双方向性のため、ユーザとのインタラクションが増加する課題があった。一方、現実の業務規則では、(1)データ伝搬が一方方向性であり、(2)個々の処理事例の独立性が高く、他の処理事例との比較集計処理がない場合も多い。そこで、本稿では、データ伝搬が一方方向であることを利用し、インタラクションを削減できる、一方方向性の制約記述を用いた業務システム構築法を提案する。提案手法を税務処理に適用し、システム構築法の妥当性を確認した。

2. 業務処理の制約表現

業務処理の本質は業務規則であると考え、条例を制約表現することで、データ状態の整合性を確保する。制約は、静的なデータの状態及び関係を定義している。条例を制約として表現する際には、データの取りうる値をすべて網羅する。たとえば、条例「納税者が20日以内に軽自動車税に係る徴収金を納税しない場合、市町村の徴税役員は、納税期限後督促状を発行する」を例として考える。この場合、条例の許す4つの制約充足状態を表1に示す。また表2に制約違反状態の例を示す。

表1. 制約表現例1

	納税有無	20日以内	督促状
(A)	×	×	○
(B)	○	×	×
(C)	○	○	×
(D)	×	○	×

表2. 制約違反例

(E)	○	×	○
-----	---	---	---

表2における制約違反の状態は制約の条件に合わないデータである。処理途中で(A)の状態から(E)になったとする。この場合、制約表では判断できない。そこで従来研究では、最新のデータに重みをつける手法がとられている[1]。最新のデータである「納税の有無」が○の状態は(B)か(C)となる。どちらなのかシステムでは判断できないためユーザに提示しないと修正できない。

3. 提案手法

3.1. 制約充足の方向性

制約には、条件部から結果部を求める方向性が存在する。図1の例において、条件部を表1で示す「納税有無」と「20日以内」の部分とし、結果部を「督促状」の部分とする。この2つの条件部が揃って初めて結果部が決定する。

たとえば、表1の(A)の状態の場合を考える。納税者が納税した場合、「納税有無」が×から○に変更される。つまり(E)の状態になるが、制約表でチェックすることで、結果部を×に変更し、(B)の整合性のとれた状態になる。しかし、条例からわかるように、結果部である「督促状」を変更したとしても、条件部の「納税有無」や「20日以内」の制約状態が変更されることはない。よって、条件部から結果部への方向性が存在することがわかる。

制約チェック(データ変更時、確認、充足、変更)

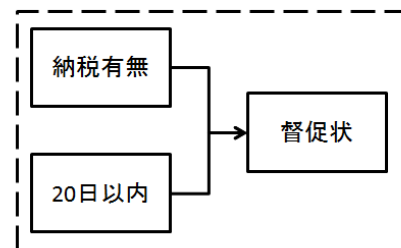


図1. 制約の方向性

3.2. 制約による連鎖的自動充足

ユーザによって、入力データが変更されると入力済みのデータ状態により、連鎖的に制約を充足することが可能である。ここで、2章で示した条例に加え、別の条例を基に作成した制約表を表3に示す。また、表1の制約表と、表3の制約表の制約伝搬のイメージ図を図2に示す。

A method of construction business system based on constraint of unidirectional process

Hiroki Kuwayama, Yoshinao Nakanishi, Shigeo Kaneda, Graduate School of Science and Engineering, Doshisha University

表 3. 制約表現例 2

総排気量(l)	最大積載量(t)	税額
1.0~1.5	~1.0	12000
1.5~2.0	1.0~2.0	15000
1.0~2.0	~2.0	20000

制約チェック(データ変更時, 確認, 充足, 変更)

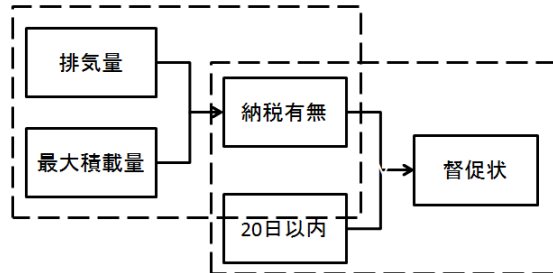


図 2. 制約による伝搬

表 3 より, 排気量と最大積載量を入力することで税額は決定する. その税額と納税金の入力情報により「納税有無」の状態も決定できる. つまり, 業務規則を制約毎に分けて表現することで, 整合性を取ることができる. このように, 半順序の関係にある処理の流れの場合, 連鎖的にデータ伝搬が行われる. また, 当初入力したデータが修正される(たとえば, 税の減免申告等)ことがあると, 一度制約充足状態だったものが, 制約違反状態になるが, 修正箇所から連続的に修正され, 再度整合性を保つ. つまり, 後戻りして修正するルーチンを書く必要がない.

3. 3. システム構成

システムは以下の 3 つのステップによってデータ伝搬を行う. 図 3 にシステム動作のイメージ図を示す.

- ①データが変更されたとき, コントローラ層の制御クラスに変更されたデータの種別を伝え, 制御クラスがすべての制約に, 変更データが関係しているか調べる.
- ②制約が満たされているかを確認する. 制約が満たされている場合は, 処理を終了する. 満たされていない場合, 充足可能かを調べるために, 条例との整合性を確認し, 自動的に書き換えを行う.
- ③充足が完了し, 制約と条件部が満たされた場合, 結果部を変更する.

また, 再度データが変更されたとき, ①から処理を繰り返し行う.

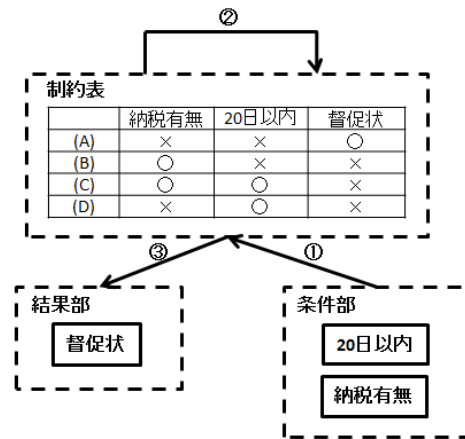


図 3. 提案システムのシステム構成

4. 評価実験

本実験では業務システムの一つである税務システムに適用した. 税務システムにおける業務規則は, 地方税法と条例である. 評価方法としては, 提案手法と従来手法(データに強度を付加する手法と一覧で表示する手法)で入力回数, 質問回数, 自動充足回数を基に比較する. 以下に結果を示す.

表 4 から従来手法より質問数が減っているため, システム構築法の妥当性が確認できる.

表 4. 評価実験結果

	提案手法	従来手法 (強度付加)	従来手法 (一覧表示)
入力回数	9	12	15
質問回数	1	2	8
自動充足回数	7	7	0

5. まとめ

本稿では現実の業務規則のデータ伝搬が制約1つ1つで単方向性であることを利用し, 制約を中心とした業務システム構築法を提案した.

業務規則には方向性が存在するため, 結果部から条件部を決定することはないと考える. その結果, ユーザへのインタラクションが削減された. また, この方向性を利用して制約表現することで, 制約から制約へと単方向にデータが伝搬し, 連鎖的にすべての制約状態の整合性を確保した.

参考文献

- [1]Megumi Ishii, Yutaka Sasaki, Shigeo Kaneda: A Constraint-Satisfaction Approach to Clerical Work. IEEE Intelligent Systems 15(1): pp.64-72 (2000, Jan/Feb.)