集約入出力条件に基づく ワークフロートランザクションの並行実行の等価性

徐　海燕† 古川　哲也††

† 福岡工業大学情報工学部　〒 811-0295 福岡市東区和白東 3-30-1
†† 九州大学大学院経済学研究院　〒 812-8581 福岡市東区箱崎 6-19-1
E-mail: †xu@cs.fit.ac.jp, ††furukawa@en.kyushu-u.ac.jp

あらまし　ビジネスの自動化を図るために、多くの企業がワークフローを導入している。ワークフローに従う実行間に共有データが存在する場合があるので、並行処理制御が必要となる。ワークフローにおいては、作業の基本単位であるタスクに入出力条件によって表されるアプリケーションの一貫性情報がある。基本操作単位の一貫性情報に基づく並行実行の等価性に関する研究は、以前から行われているが、本論文では、ワークフロートランザクションの並行実行に対して、実行順序の変換後の並行実行の正当性を保証できる等価変換の性質について分析する。さらに、一連のタスクに従う実行からまとめられた集約入出力条件に基づく並行実行の等価性についても検討する。集約入出力条件を利用することより、不必要な中間状態をカプセル化でき、正当な実行となるための等価交換がスムーズに行えることを示す。
キーワード　ワークフロー、並行実行、トランザクション、隔離性、一貫性、タスク、入出力条件

Equivalence between Workflow Transactions based on Aggregated Conditions

Haiyan XU† and Tetsuya FURUKAWA††

† Dept. Computer Science and Engineering, Wajirohigashi 3-30-1, Higashi-ku, Fukuoka, 811-0295 Japan
†† Dept. Economic Engineering, Kyushu University, Hakozaki 6-19-1, Higashi-ku, Fukuoka, 812-8581 Japan
E-mail: †xu@cs.fit.ac.jp, ††furukawa@en.kyushu-u.ac.jp

Abstract　To facilitate business process and information process reengineering and automation, workflow management systems have been widely used in the applications such as e-commerce and manufacturing. Conventional concurrency control approaches would overly restrict the concurrency of interleaved execution of workflow instances. In this paper, we discuss the equivalence between transactional workflow schedules from the view points of how to keep the correctness of the schedules. By integrating several tasks into an aggregate one, furthermore, we show that its equivalent schedules can be found smoothly.
Key words　Workflow, Concurrency control, Transaction, Isolation, Consistency, Task, Input and output conditions

1. ま え が き

ビジネスプロセスの自動化、および情報共有の効率化を図るため、多くの企業においてワークフロー管理システムが導入されている [1]。その下では、個々の仕事（タスク）と仕事の流れ（制御フロー）を事前に定義し、それに従って順次仕事を処理する。ワークフロー内のいくつかのタスク、または異なるワークフローのタスクの操作するデータに共有データが存在することがある。したがって、タスクの実行の正当性を保証するために並行処理制御が必要となる。ワークフローに従う実行は注文から入荷までのように数週間に渡ることがあるので、ワークフローの特徴を活用した並行処理制御が重要である。

ワークフローにおいては、作業の流れである制御フローやデータの流れであるデータフローに関する情報が利用でき、作業の基本単位であるタスクに関しても、入出力条件としてアプリケーションの一貫性情報がある。一貫性情報や述語に基づく競合操作の実行順序の変更に関する研究は、以前から行われているが [2], [5]、本論文では、ワークフロートランザクションの並行実行に対して、変換後の並行実行も正当性を保持できる等価交換について検討する。具体的には、ワークフロートランザクションの並行実行の正当性が入出力データの隔離性 [8] からなっていることに対して、等価交換もそれに応じて細分する。さらに、一連のタスクに従う実行からまとめられた集約入出力条件に基づく並行実行の等価性についても検討する。タスクを集約するすることによって、問題となる中間状態をカブセル化でき、等価交換もより適切に行えることを示す。

例えば、注文を受けた後、在庫が不足しているなら取寄せをしたのち同時に配達と支払のタスクを実行する注文業務を考える。顧客 A が注文した時は在庫があるので、図 1 (a) のように実行されることになり、希望配達日は 2 週間後とする。その直後注文した顧客 B は在庫不足となったため、図 1 (b) のような取寄せを含んだ実行となる。

注文 $\mathcal{T}_1^2 \rightarrow$ 支払 $\mathcal{T}_3^2 \rightarrow$ 配達 $\mathcal{T}_4^2$

(a)

注文 $\mathcal{T}_1^1 \rightarrow$ 入荷 $\mathcal{T}_2^1 \rightarrow$ 配達 $\mathcal{T}_4^1 \rightarrow$ 支払 $\mathcal{T}_3^1$

(b)

図 1　顧客 A, B のワークフローに従う実行 WT_2 と WT_1

Fig. 1　Transaction WT_1 and WT_2 of Client B and A

B の希望配達日は 2 日以内であるが、入荷日は 10 日後となっているとする。A の予約した品を A に、B の予約した品を B に配達するなら、B への配達は希望した時間内に行えない。希望通りに配達してもらえなければ、顧客は注文をキャンセルすることがある。そのため、如何に既存の正当な並行実行に、新たなトランザクションの操作列を正当性を保持するように挿入するか、という問題が生じる。結果が等価となる実行順序の入れ換えである等価交換を通じてこの問題を考える。

等価交換の単位をどのようにすれば等価交換で得られる並行実行をより多く得られるかという課題も存在する。この例の場合では入荷された後 B への配達はできるので、入荷と配達をまとめて考える。A の注文数は B より多いとすると、B の入荷と配達をまとめた作業の内部の 2 つの作業の実行順序を入れ換えても配達作業は在庫数が配達数以上あるという入力条件を満たす。図 2 に示しているように B は A が予約した品を配達しているが、A も B の注文で入荷された品で希望される配達日には配達できるので、正当な並行実行である。

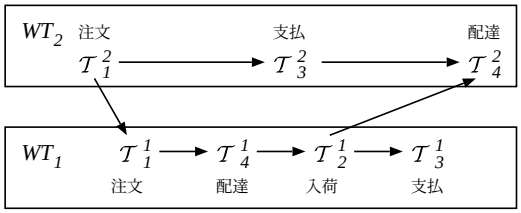

図 2　実行順序の入れ換えによる並行実行 WH_1

Fig. 2　WH_1 following delivery orders

与えられたワークフロートランザクションの並行実行に対して、それが正当かどうかを判定する方法は提案されている [8]。しかし、上記の例で示しているようにワークフロートランザクションの並行実行の正当性基準に基づく等価交換の性質の究明が重要である。また、等価交換の単位をどのようにすればより多くの並行実行を等価交換を通して得られるかという課題もある。例えば、上記の例を通してタスクを集約することで、タスクを個別に扱うことより広い範囲で並行実行の等価性を考えることができる可能性を示した。これはタスクを集約することによって中間入出力データや中間入出力データに対する入出力条件をカプセル化することができるためである。

本論文は、次のように構成される。2 節でワークフロートランザクションを定義し、正当なスケジュールの判定方法について述べる。スケジュール間の等価交換については 3 節で、スケジュールと集約後のスケジュールの性質については 4 節で検討する。5 節は本論文の結果についての議論と全体のまとめを行う。

2. ワークフロートランザクションの並行実行

本節では、ワークフロートランザクションを導入し、並行実行における正当性を定義する。

2.1 ワークフロートランザクション

ワークフローは、タスクとタスク間の実行順序を与える制御フローによって記述される [3]。タスク t はその特徴を表す 6 つのパラメータ、入力/出力データ項目の集合 I/O、入力/出力データに対する条件 IC/OC、関数 F、起動時間 ST からなる組 $t(I, O, F, IC, OC, ST)$ によって記述される。入力条件 IC は、入力データ項目に対する条件の集合である。関数 F は、出力データ項目の計算式の集合である。出力条件 OC は、出力データに関する条件の集合である。起動時間 ST は t の起動できる時間に関する指定を与えている。入出力データに含まれるデータ項目は肩字で区別する。

［**例 1**］ 商品販売のワークフロー管理システムには、次の 4 つのタスクがあるとする。簡単のため、単価が $tanka$ である単一の商品を販売するものとする。

- 注文 t_1：

I_1={ 顧客:x_1, 数量:x_2, 在庫数:x_3, 配達日:x_6,
注文総数:y},
O_1={ 取寄せ数:x_4, 注文総数:y, 配達日:x_6,
入荷日:z},
$F_1 = \{$if $x_3 - y^I \geqq x_2$ then $x_4 = 0$
else $\{x_4 = y^I + x_2 - x_3$, z=today+10$\}$,
$y^O = y^I + x_2\}$
$IC_1 = \{x_3 \geqq 0,\ x_2 > 0\}$,
$OC_1 = \{$ if $x_3 - y^I \geqq x_2$ then $x_6 > today$
else $x_6 \geqq z$,
$x_3 - y^O + x_4 \geqq 0$,$\}$,
$ST_1 = \{\}$.

- 入荷 t_2:

I_2 ={ 取寄せ数:x_4, 在庫数:x_3},
O_2 ={ 在庫数:x_3},
$F_2 = \{x_3^O = x_3^I + x_4\}$,
$IC_2 = \{x_3 \geqq 0,\ x_4 > 0\}$,
$OC_2 = \{x_3^O = x_3^I + x_4\}$,
ST_2 = { 入荷日 :z}

- 支払 t_3:

I_3={ 顧客 ID:x_1, 数量:x_2},
O_3={ 支払額:x_5},
$F_3 = \{x_5 = x_2 * tanka\}$,
$IC_3 = \{x_2 > 0\}$,

$OC_3 = \{x_5 = x_2 * tanka\}$,
$ST_3 = \{\}$.

- 配達 t_4:

I_4={ 数量:x_2, 在庫数:x_3, 配達日:x_6, 注文総数:y},
O_4={ 在庫数:x_3, 注文総数:y},
$F_4 = \{x_3^O = x_3^I - x_2, y^O = y^I - x_2\}$,
$IC_4 = \{x_3^I \geqq x_2,\ y^I \geqq x_2,\ x_2 > 0\}$,
$OC_4 = \{x_3^O = x_3^I - x_2, y^O = y^I - x_2\}$,
ST_4 = { 配達日 :z} □

並行実行において、タスク $t(I, O, F, IC, OC, ST)$ は原子性の単位である。タスク t は入力データ I が入力条件 IC を満たすかどうかを検査する。満たしていれば、関数 F に従って出力データ O を生成し、生成された出力データは出力条件を満たす。

タスクの実行順序を、タスクを節点とする有向巡回グラフである制御フローによって記述する。順序には、順次、反復、条件分岐（XOR 分岐、XOR 結合）などがある。XOR 分岐にはいくつかの枝分かれがあり、それらの枝にはいずれか 1 つが真となる経路選択条件が記述される。例 1 の注文を受付け、在庫不足の場合のみ取寄せを行い、その後支払と配達を行う制御フロー CF_1 を図 3 に示している。

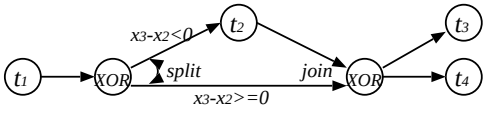

図 3　制御フロー $CF1$
Fig. 3　Control Flow $CF1$

トランザクションは、ワークフローに従うタスクの実行列である。形式的に記述するために、データ項目 x およびデータ項目集合 X のインスタンスを $\mathcal{I}_x$ および $\mathcal{I}_X$ とし、$\mathcal{T}(\mathcal{I}_I, \mathcal{I}_O, F, IC, OC, \mathcal{I}_{ST})$ でタスク $t(I, O, F, IC, OC, ST)$ のプロセスを記述する。トランザクションは、タスクのプロセスを表す節点集合とそれらの間の実行順序を表す枝集合からなる有向グラフ $WT(TN, TE)$ で定義する。$WT(TN, TE)$ において XOR 分岐節点は省略することがあるが、省略しない場合は、真となる経路選択条件がトランザクション内の XOR 分岐節点の出力条件となる。

トランザクションは、全順序グラフである。これは、制御フローが巡回の場合は複数回実行されるタスクは異なるプロセスに展開され、条件分岐は特定の経路のタスクに従って実行されるためである。制御フロー CF_j $(j = 1, 2, \cdots)$ 上のトランザクション $WT_i(TN_i, TE_i)$ $(i = 1, 2, \cdots)$ からなるスケジュールは、次のような有向非巡回グラフ $WH(HN, HE)$ である。

- $HN = \bigcup_i TN_i$, $HE \supseteq \bigcup_i TE_i$
- 異なる処理単位の節点 $\mathcal{T}_s$ と $\mathcal{T}_t$ が競合する $((O_s \cap (I_t \cup O_t) \neq \phi) \vee (O_t \cap (I_s \cup O_s) \neq \phi))$ なら、それらの間には実行順序を示す枝か推移的な枝が HE に含まれる。
- HE で示される実行順序は各節点の起動時間で示された順序と一致する。

2.2　正当なスケジュール

本節では、単独実行時に各トランザクションが満たすべき一貫性と並行実行時に同じ性質を満たすようにするための隔離性に関するこれまでの結果について述べ [8]、一貫性と隔離性を満たすトランザクションからなるスケジュールを正当なスケジュールとする。

ワークフロートランザクションの一貫性に対して、入力データの一貫性はタスク単位で、出力データの一貫性はトランザクション単位で管理している。まず、各節点の入力データについて、トランザクション内の祖先節点の出力データであることもあり、データベースのデータである場合もある。前者を内部データ、後者を外部データと呼ぶ。このため、入力データの一貫性は、入力データが入力条件を満たすことと外部入力データが一貫したデータベースから検索していることとなる。通常内部データが入力条件を満たすことはトランザクションの祖先節点の出力条件によって保証され、一貫したデータベースで実行されれば入力条件は満たされると仮定されている。一方、出力データの一貫性は、入力データの一貫性を満たすトランザクションが一貫したデータベースにおいて単独で実行されたとき、その結果のデータベースは一貫していることを意味する。

一貫性が入出力データ別になっているため、隔離性も入出力データごとに扱う。入力データの隔離性と出力データの隔離性は、並行実行におけるトランザクションの入力データの一貫性と実行結果のデータベースの一貫性を保証するための性質である。例えば、WT_1 の実行で、A に配達する予定の在庫品を他の顧客に使用されたままでは、配達 $\mathcal{T}_4^1$ の入力条件を満たさなくなる。支払 $\mathcal{T}_3^1$ の出力条件である支払額に対する条件を満たさなくなる操作があれば、品物は届けたのに支払はなされてないことになりデータベースの一貫性を満たさない。

入力データの隔離性はさらに内部/外部入力データの隔離性に分けられる。外部入力データの隔離性は、他のトランザクションの中間結果を検索していないかどうかのグラフ判定方式が提案されている [8]。直列可能なスケジュールにおいて、各節点は外部入力データの隔離性を満たすが、各節点が外部入力データの隔離性を満たすスケジュールは直列可能とは限らない [8]。

内部入力データの隔離性と出力データの隔離性については、実行が終了した各節点の入出力データに対して、その満たしている入出力条件が並行実行によって満たさなくなってないことで保証される [8]。トランザクション WT の実行中のプロセス $\mathcal{T}_i$ に対し、それまでの各プロセスの入力データ項目と出力データ項目の和集合を $\mathcal{T}_i$ の使用データ項目 D_i といい、それらの最新値を $\mathcal{I}_{D_i}$ で表す。$\mathcal{T}_i$ までの入出力条件で最新値 $\mathcal{I}_{D_i}$ のみに関わるものからなる集合を、$\mathcal{T}_i$ の結果条件 TC_i という。また、トランザクション WT の最終プロセスの結果条件を、トランザクションの結果条件とする。

[**定理 1**]　[8] トランザクション $WT_i(TN_i, TE_i)$ $(i = 1, 2, \cdots)$ からなるスケジュール WH において、WT_i の各実行時点で使用データ項目のデータベースにおける値が直前に終了したプロセスの結果条件を満たせば、WT_i は出力データと内部入力データの隔離性を満たす。 □

例えば、WH_1 の配達 $\mathcal{T}_4^1$ と入荷 $\mathcal{T}_4^1$ の順序を入れ換えたスケジュール WH_2(図 4) は定理 1 を満たし、外部入力データの隔離性も満たす。一方、タスクの入力条件は与えられており、トランザクションの結果条件はその定義から求められる。内部入力データの隔離性と出力データの隔離性の必要十分条件をそれぞれ、

（1） トランザクション内の各プロセスが実行されるときに入力条件を満たす。

（2） トランザクションの終了時に、データベースにおけるトランザクションの使用データ項目の値が結果条件を満たす。

とすることができる [8]。例えば、WH_1 は上記の条件を満たすが、定理 1 を満たさない。これは、配達 $\mathcal{T}_4^1$ の実行が終了した時点で在庫数が配達 $\mathcal{T}_4^4$ の配達数より多いという条件が満たされなくなっているためである。したがって、定理 1 は出力データと内部入力データの隔離性を満たすための十分条件を与えている。

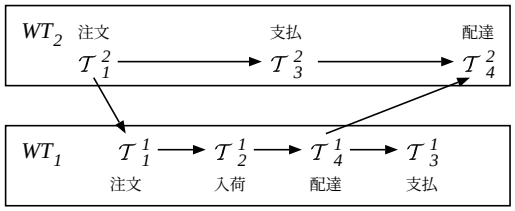

図 4　隔離性を満たすスケジュール WH_2
Fig. 4　WH_2 satisfying Theorem 1

3. ワークフロースケジュールにおける等価交換

本節では集約節点を含むスケジュールにおける等価交換について検討する。

3.1 集約タスクのパラメータ

トランザクションはワークフローに従う実行であり、トランザクションの節点はタスクの実行であるため、タスクをまとめることによる入出力データ、出力関数、入出力条件といったパラメータの変化から集約後のスケジュールの性質について考察する。

［**定義 1**］ スケジュール $WH(HN, HE)$ において、隣接する 2 つの節点の属するタスク t_i, t_j から集約されたタスク t のパラメータ (I, O, F, IC, OC, ST) は、次のようになる (図 5)。ここで t_i,t_j は集約されたタスクまたは通常のタスクである。

$I = I_i \cup (I_j - O_i)$

$O = O_j \cup (O_i - O_j)$

$F = f_i(X)$　if $f_i(X) \in O_i - O_j$

$\quad = f_j(Y)$　if $y_k \in Y \in O_i \cap I_j$ then $y_k = f_i(X)$

if $y_k \in Y \notin O_i$ then $y_k = y_k$

$IC = IC_i \cup$ (IC_j 中の $(I_j - O_i)$ に係わる条件(注1))

$OC = OC_j \cup$ (OC 中の $(O_i - O_j)$ に係わる条件)

$ST = ST_i$ if $ST_i \neq \phi$ else ST_j　□

図 5 で示しているように入出力データ項目はそれぞれ、t_i, t_j の入出力データ項目の和集合である。入出力条件と同様に、出力関数は、t_j の内部入力データに対しては再帰的に計算している。集約する節点は同じトランザクションに属することもあり、異なるトランザクションに属することもある。例 2 は、前者の例である。

(注1)：$O_i \cap I_j$ 中のデータ項目 y は $y = f_i(x) \in F_i$ によって置き換える。

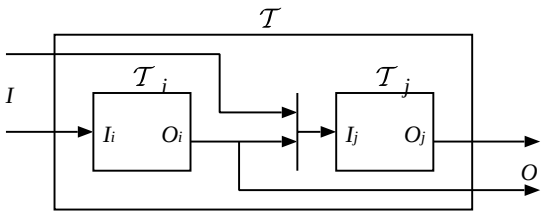

図 5　集約タスクのイメージ
Fig. 5　Image of Aggregate Tasks

［**例 2**］ 例 1 の配達と入荷タスクの集約タスクは、それぞれ次のようになる。

- 入荷・配達 t_{24}

I_{24} ={ 数量:x_2, 在庫数:x_3, 取寄せ数:x_4,
　配達日:x_6, 注文総数:y, 入荷日:z},

O_{24}={ 在庫数:x_3, 注文総数:y},

$F_{24} = \{x_3^O = x_3^I + x_4 - x_2, y^O = y^I - x_2\}$,

$IC_{24} = \{x_2 > 0, x_4 > 0, x_3 + x_4 \geqq x_2, y^I \geqq x_2\}$,

$OC_{24} = \{x_3^O \geqq 0, y^O \geqq 0\}$,

ST_{24}={ 入荷日:z}.

- 配達・入荷 t_{42}

I_{42}={ 数量:x_2, 在庫数:x_3, 取寄せ数:x_4,
　配達日:x_6, 注文総数:y, 入荷日:z},

O_{42}={ 在庫数:x_3, 注文総数:y},

$F_{42} = \{x_3^O = x_3^I + x_4 - x_2, y^O = y^I - x_2\}$,

$IC_{42} = \{x_2 > 0, x_4 > 0, x_3 \geqq x_2, y^I \geqq x_2\}$,

$OC_{42} = \{x_3^O \geqq x_4, y^O \geqq 0\}$,

ST_{42}={ 配達日:x_6}.

3.2 実行結果に基づく等価交換

各節点 $\mathcal{T}$ は、入出力データ項目とその値、出力関数、入出力条件、起動時間といったパラメータを有する。本節では変換前後の結果の集約タスクの入出力データ、出力関数に焦点を当てた等価交換について考察する。

［**定義 2**］ 節点 $\mathcal{T}_i$ と $\mathcal{T}_j$ に対して、$\mathcal{T}_i, \mathcal{T}_j$ の順で集約された節点の入出力データ項目、出力関数と $\mathcal{T}_j, \mathcal{T}_i$ の順で集約された節点の入出力データ項目、出力関数が同一であるとき、$\mathcal{T}_i$ と $\mathcal{T}_j$ を入れ換えたスケジュールは、実行結果に基づく等価交換であるという。　□

明かに実行結果に基づく等価交換は推移律を満たす。

例えば、WT_1 中の入荷 $\mathcal{T}_2^1$ と配達 $\mathcal{T}_4^1$ の交換は実行結果に基づく等価交換である。これは集約節点入荷・配達 $\mathcal{T}_{24}^1$ と配達・入荷 $\mathcal{T}_{42}^1$ の入出力データ項目と出力関数が同一のためである。したがって、WH_1 と WH_2 は実行結果に基づく等価交換前後のスケジュールである。

［**例 3**］ 配達 $\mathcal{T}_4^2$ の配達日は 2 週間後ではなく 1 日後とすると、

図 2 のスケジュール WH_1 は図 6 のようになる。 □

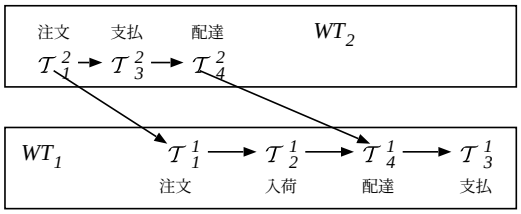

図 6 スケジュール WH_5
Fig. 6 Schedule WH_5

この場合に、配達 $\mathcal{T}_4^2$ と集約節点入荷・配達 $\mathcal{T}_{24}^1$ の交換も実行結果に基づく等価交換である。これは、2 つの順序で集約した後の入出力データと出力関数が、それぞれ次のようになるためである。

I ={ 数量:x_2^1,x_2^2, 在庫数:x_3, 取寄せ数:x_4,
配達日:x_6^1,x_6^2, 注文総数:y, 入荷日:z^1,z^2},
O={ 在庫数:x_3, 注文総数:y},
$F = \{x_3^O = x_3^I + x_4 - x_2^1 - x_2^2, y^O = y^I - x_2^1 - x_2^2\}$,

3.3 入出力データの隔離性を満たす等価交換

実行結果に基づく等価交換では、交換前後の集約節点の入出力データ項目と出力関数は同じであるが、入力条件が同じであるとは限らないし、同じ外部入力データに対して実行される交換前後のスケジュールにおいて同じ節点の入力データの値が同一とも限らない。例えば、例 2 で示しているように配達・入荷の集約タスク t_{42} の入力条件には在庫数が配達数以上を意味する条件「$x_3 \geqq x_2$」を含むが、入荷・配達の集約タスク t_{24} の入力条件にはそれが含まれていない。また、WH_5 から配達 $\mathcal{T}_4^4$ 節点と集約節点入荷・配達 $\mathcal{T}_{24}^1$ を交換した後、集約節点 $\mathcal{T}_{24}^1$ の入力データである在庫数の値は異る。したがって、交換後の節点が入力データの隔離性を満たすとは限らない。このため、交換後のスケジュールも入力条件を満たすことと外部データの隔離性を満たすことをそれぞれ定義する。

［**定義 3**］ 与えられたデータベースにおいて、交換前のスケジュール中の各節点 $\mathcal{T}$ の入力データ項目が入力条件を満たせば、交換後のスケジュールにおいても同様な性質を満たす交換を実行可能な等価交換という。 □

等価交換の実行可能性は実行されるデータベースの初期値にも依存する。顧客 A の配達日は 2 週間後、顧客 B の配達日は 2 日後、入荷日は 10 日後、在庫数は顧客 A の注文数より多い、顧客 A の注文数は顧客 B より多いという場合に、WH_1^a は WH_2^a の実行可能な等価交換の結果である。

交換前後の集約節点の出力関数が同じであることは、同じ入力データに対して交換前後の集約出力データの値は同一であることを意味する。すなわち、実行可能かつ実行結果に基づく等価交換であれば、交換前後のスケジュールの実行結果は同じである。

次に、外部データの隔離性を満たす等価交換について検討する。例えば、トランザクション WT の口座 X からお金を降ろすタスク t_i のプロセスと、口座 Y に入金するタスク t_{i+1} のプロセスの間に、他のトランザクションの口座 X、Y の残高を参照するタスク t_r のプロセス $\mathcal{T}$ を移動することを考える。その交換は実行結果に基づく等価交換であるが、$\mathcal{T}$ が WT の中間結果を検索することになる。すなわち、$\mathcal{T}$ は外部データの隔離性を満たさない。

［**定義 4**］ 外部入力データの隔離性を満たす隣接する節点 $\mathcal{T}_i$ と節点 $\mathcal{T}_j$ に対して、$\mathcal{T}_i, \mathcal{T}_j$ の順を $\mathcal{T}_j, \mathcal{T}_i$ の順へ交換したとき、交換後のスケジュール WH の $\mathcal{T}_i$ までの部分スケジュールが外部入力データの隔離性を満たせば、外部入力データの隔離性を満たす等価交換であるという。 □

$\mathcal{T}_i, \mathcal{T}_j$ の順で隣接する操作を $\mathcal{T}_j, \mathcal{T}_i$ の順への交換によって、$\mathcal{T}_j$ が $\mathcal{T}_j$ の属する WT の中間結果を検索することにつながる可能性がある。外部入力データの隔離性を満たす等価交換では、等価交換によってこのような新たな問題を生じないように制限している。

実行結果に基づく等価交換と実行可能で外部入力データの隔離性を満たす等価交換をまとめて、入出力データの隔離性を満たす等価交換という。

［**定理 2**］ 正当なスケジュール WH と入出力データの隔離性を満たす等価交換で得られたスケジュール WH' も正当である。 □

証明. 等価交換によって外部入力データの隔離性上で新たに問題が生じないことは、外部入力データの隔離性を満たす等価交換で保証している。

正当なスケジュール WH と実行可能な等価交換によって得られる WH' 中の各節点の入力データ項目は入力条件を満たす。入力条件を満たせば、各節点の出力データは出力条件を満たす。一方、WH' は内部入力データの隔離性を満たす、すなわち WH' 内の各節点の内部データが入力条件を満たすことは、トランザクションの祖先節点の出力データが出力条件を満たすことによって保証される。さらに、実行可能かつ実行結果に基づく等価交換であれば、交換前後のスケジュールの実行結果は同じである。したがって、正当なスケジュール WH と実行可能かつ実行結果に基づく等価交換で得られる WH' も出力データの隔離性を満たす。 □

4. 集約スケジュール

等価交換に対するもう 1 つの課題は、交換の単位である。例 3 の場合においては、顧客 B の要望に答えたスケジュールは WH_6(図 7) となる。

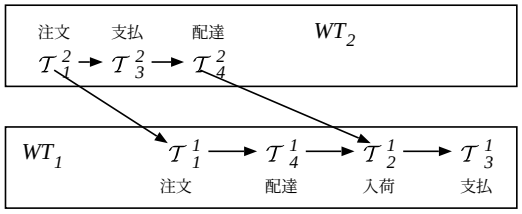

図 7 スケジュール WH_6
Fig. 7 Schedule WH_6

売り上げ総額を向上するため顧客 A の配達日をある程度ずらし

て WH_6 と実行可能な等価交換を探る。ここで、配達 $\mathcal{T}_4^1$ と配達 $\mathcal{T}_4^2$ の実行結果に基づく等価交換は実行可能な等価交換ではなく、配達 $\mathcal{T}_4^2$ と配達・入荷の集約節点 $\mathcal{T}_{42}^1$ の実行結果に基づく等価交換は実行可能な等価交換となっている。

すなわち、個別の操作を交換の単位とするより集約した節点単位で行った方が有効である場合がある。本節では、トランザクションのいくつかの節点をまとめたスケジュールと集約前のスケジュールの正当性の関係について検討する。

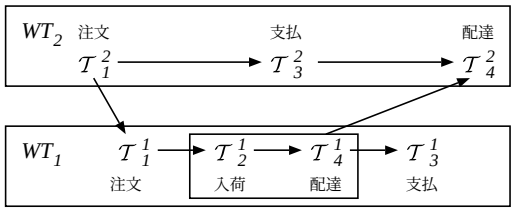

図 8　集約スケジュール WH_2^a
Fig. 8　Aggregate Schedule WH_2^a

トランザクション $WT_i(i = 1, 2, \cdots, n)$ からなるスケジュール WH において、ある $WT_j(j \in 1, 2, \cdots, n)$ のいくつかの節点を集約節点としてまとめた後のトランザクションを WT_j^a、まとめた後のスケジュールを WH^a と記する。WH_2 に対して入荷節点 $\mathcal{T}_2^1$ と配達節点 $\mathcal{T}_4^1$ を集約したスケジュール WH_2^a と WH_1 の集約スケジュール WH_1^a を、それぞれ図 8 と図 9 に示す。

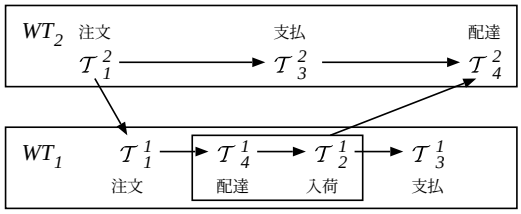

図 9　集約スケジュール WH_1^a
Fig. 9　Aggregate Schedule WH_1^a

本論文の例の実行環境においては WH_2^a は定理 1 を満たすため、正当なスケジュールである。WH_1^a は WH_2^a の入出力データの隔離性を満たす等価交換であるため、定理 2 より WH_1^a も正当なスケジュールである。

ワークフロートランザクションの隔離性は、外部入力データの隔離性、内部入力データの隔離性、出力データの隔離性に分けられるので、まず外部入力データの隔離性に対して集約前後のスケジュールの関係について検討する。

［**補題 1**］　外部入力データの隔離性を満たすスケジュール WH の集約スケジュール WH^a も外部入力データの隔離性を満たす。□

証明. 定義 1 より t_i, t_{i+1} の順で実行される 2 つのタスクから集約されたタスク t の入力データ項目 I は $I_i \cup (I_{i+1} - O_i)$ となる。一方、I_{i+1} 中の外部入力データ項目は I_{i+1} から $\mathcal{T}_{i+1}$ の祖先の出力データ項目を除いたものである。すなわち、I_{i+1} 中の外部入力データ項目と $I_{j+1} - O_i$ 中の外部入力データ項目は等しく、WH^a が外部入力データの隔離性を満たす必要十分条件は、WH が外部入力データの隔離性を満たすことである。□

次に、内部入力データの隔離性と出力データの隔離性に対して、集約前後のスケジュールの関係について検討する。集約後のスケジュールの各実行時点において、使用データ項目の値は集約前のスケジュールの同時点での値と同じなので、次の結果が成り立つ。

［**定理 3**］　トランザクション $WT_i(i = 1, 2, \cdots)$ からなるスケジュール WH が定理 1 を満たすなら、その集約スケジュールも定理 1 を満たす。□

例えば、WH_2 が定理 1 を満たすので、WH_2^a も定理 1 を満たす。ただし、定理 3 の逆は成り立つとは限らない。例えば、WH_1 は定理 1 を満たさないが、WH_1^a は定理 1 を満たす。

一方、集約後のスケジュールの各実行時点において、使用データ項目の値と内部入力データの値も集約前のスケジュールの同時点での値と同じである。集約後のスケジュールの判定時点は集約前より少ないので、出力データの隔離性と内部入力データの隔離性の必要十分条件を満たす WH の集約スケジュール WH^a も出力データの隔離性と内部入力データの隔離性の必要十分条件を満たす。逆に、外部入力データの隔離性を満たす場合での内部入力データの隔離性は、入力データが入力条件を満たすかどうかによって検査できる。このため、正当な WH^a に対して WH 内の各節点が入力条件を満たせば内部入力データの隔離性を満たす。補題 1 とまとめると、次の結果が成り立つ。

［**定理 4**］　正当なスケジュール WH の集約スケジュール WH^a も正当なスケジュールである。集約スケジュール WH^a が正当なら、WH 内の各節点が入力条件を満たせば正当である。□

例えば、本論文の例の実行環境においては、WH_1^a が正当でありかつ WH_1 内の各節点が入力条件を満たすため、定理 4 より WH_1 も正当なスケジュールである。

5. ま と め

等価交換には、従来の競合関係のみに基づくものがある [4]。そのような交換は、入出力条件で表される意味論を用いていない。意味論に基づく等価交換には、スケジュール WH が正当なら、交換後のスケジュール WH' も正当というものがある [2]。例えば、同一の口座に対する引き出し操作、入金操作と言う順序を含むスケジュール WH と、入金、引き出しという交換後の順序で実行するスケジュール WH' がその例である。しかし、それらは直列可能性に基づく等価である。本論文で提案した等価交換はワークフロートランザクションの並行実行の正当性に基づくものである。さらに、集約タスクによって一部の中間状態をカプセル化することができる。

本論文の考え方に従って、与えられた正当なスケジュールから利用者の要望を満たす等価なスケジュールを、次のような手順で求めることが考えられる。まず利用者の要望に沿ってトランザクション内の実行結果に基づく等価交換を行う。1 章の例の場合では入荷節点 $\mathcal{T}_2^1$ と配達 $\mathcal{T}_4^1$ 節点の交換に当たる。次にそれは実行可能な等価交換であるかどうかを検査する。実行可能な等価交換であればよいが、そうでない場合には、実行結果

に基づく等価交換が行われた節点を集約した上で、スケジュールにおいて入力条件を満たす範囲内で再度実行結果に基づく等価変換を行う。1 章の例の場合では集約節点入荷・配達 $\mathcal{T}_{24}^{1}$ と配達節点 $\mathcal{T}_{4}^{2}$ を実行結果に基づく等価交換を行うことに当たる。実行可能な等価交換となるまたは交換の候補がなくなるまでこれを繰り返す。

定性的な評価と定量的な評価が今後の課題である。

文　　献

[1] Alonso, G., Agrawal, D., Abbadi, A. E., and Mohan, C., “Functionality and Limitations of Current Workflow Management Systems,” IEEE Expert: Special Issue on Cooperative Information Systems, 1997.

[2] Agrawal, D., Abbadi, A. E., and Singh, A. K., “Consistency and Orderability: Semantics-Based Correctness Criteria for Databases,” ACM Trans. Database Syst., Vol. 18, No. 3, pp. 460-486, 1993.

[3] Arpinar,L. B., Halici, J., Arpinar, S., and Dogac, A., “Formalization of Workflows and Correctness Issues in the Presence of Concurrency, ” Distributed and Parallel Databases, Vol. 7, No. 2, pp. 199-248, 1999.

[4] Bernstein, P. A., Hadzilacos, V., and Goodman, N., “Concurrency Control and Recovery in Database Systems,” Addison-Wesley, 1987.

[5] Gray, J. and Reuter, A., “Transaction Processing: Concepts and Techniques,” Morgan Kaufmann, 1994.

[6] Puustjarvi, J., Tirri, H., and Veijalainen, J., “Concurrency Control for Overlapping and Cooperative Workflows,” IEEE Transactions on Computer Systems Bulletin, pp. 24-30, 1996.

[7] Puustjarvi, J., “Workflow Concurrency Control,” Computer Journal, Issue 1, pp. 42-53, 2001.

[8] 徐 海燕、古川哲也, “ワークフロートランザクションの隔離性、” データベースと Web 情報システムに関するシンポジウム、pp. 145-152, 2002.