O-002

分散環境におけるトランザクション分割
Transactional Decomposition in Distributed Environments

片岡 慶二† 新川 芳行†
Keiji Kataoka Yoshiyuki Shinkawa

1．まえがき

クラウドコンピューティング[1]の普及は，ビジネスプロセスの自動化およびそれに関連するトランザクション処理の形態にも大きな影響を及ぼす．特に，トランザクション処理においては，クラウドにおける分散化と仮想化により，従来の ACID によるデータベース維持が困難となり，BASE と呼ばれるより弱い整合性基準が提案されている[2]．しかし，ビジネスアプリケーションの場合，中間状態の整合性が要求される場合もあり，BASE では要件を満たさないケースも出てくるものと思われる．もし，ACID ベースのトランザクションをクラウド環境で実行するなら，より小さい単位に分割したトランザクションにすることが望ましい．これにより，排他制御の範囲を小さくでき，トランザクション間の相互作用を減少させ，さらに平行性を向上できる．本論文では，カラーペトリネット(Colored Petri Nets - CPN)によるモデリングと分析により，トランザクションの整合性を維持した状態で分割する手法を提案する．

2．トランザクションシステムの概要と CPN によるモデル化

トランザクションとは，業務的もしくは論理的に関連する一連のデータベース操作で，一般に更新を伴うものをいう．この更新のため，排他制御が必要となり，デッドロック，ライブロック等による処理の遅延が発生しうる．特に，分散環境において，これらの遅延は致命的な問題となりうる．排他制御による影響は，トランザクションの粒度が小さいほど少なくなるが，処理の分割による機能的損失を防ぐことが課題となる．本節では，CPN によりトランザクション処理をモデル化することにより，トランザクション処理の機能と動作を明確化し，処理分割による損失を防止する手法につなげる．

2.1 CPN の概要

トランザクションの動的な振る舞いを表現するには，最低限

(1) 動作の基本要素であるアクションの種類
(2) 様々な環境下でのアクションの実行順序関係

を記述しなければならない．ペトリネット(Petri-nets - PN)[3]は「プレース」，「トランジション」，「アーク」，「トークン」という四つの要素でこれを表現する．ペトリネットはトランジションの発火によるトークンの移動で動作を表現するが，機能面の記述や決定性システムの表現にいくつか問題があるため，その拡張版である「カラーペトリネット(Colored Petri-nets - CPN)」[4]を使用する．CPN は，PN のトークンに「カラー」と呼ばれるデータ型を持たせ，トランジションの発火をトークンの値により制御できる「ガード関数」，トランジションによるトークンの値の変換を表す「アーク関数」を PN に付加することにより，ペトリネットによる決定性システムの表現と，具体的な機能の表現を可能にする．

2.2 CPN によるトランザクション処理のモデル化

CPN によりビジネスプロセスをモデル化する場合，アクティビティをトランジション，アクティビティに関連する資源を取り扱う組織もしくは施設をプレース，取り扱われる資源をトークンとし，アクティビティによる資源の変換機能をアーク関数で表現するのが自然な対応と考えられる[5]．一方，高度に自動化されたビジネスプロセスにおいては，アクティビティをトランザクション，組織もしくは施設をデータベース，トークンを処理されるデータと見なすこともできる．本論文は，トランザクション処理によるビジネスプロセスの自動化という視点に立つため，後者の解釈に基づき次のように CPN モデルを構築する．まず，各トランジションが一つの原子トランザクション（atomic transaction）を表す．原子トランザクションとは DBMS から見た最小の処理単位で ACID 性（Atomicity，Consistency，Isolation，Durability）が保証される単位となる．各プレースは原子トランザクションの入出力データ域もしくはデータベースを表す．プレース内トークンは処理されるべきデータ（入出力データ域用プレースに保持される場合）もしくはデータベースレコード（データベース用プレースに保持される場合）を表す．そして，各アーク関数が原子トランザクションの機能を表す．この機能はトランジションへの入力プレースのトークンを出力プレースに入れる際のデータ変換にあたる．

以上の CPN 構造に排他制御のメカニズムを追加することで，ACID ベースのトランザクション処理のモデル化が可能となる．排他制御は一般にロックにより行われる．ここでは，共有(S)と占有(X)の二種類のロックを使う．CPN では，更新に対しては，該当レコードがロックされていない時に限り，更新トランジションが発火し，参照に対しては，X ロックが取られていない限り発火可能となるよう制御する必要がある．このため，各データベースに対応するプレースに対し，ロック保持のためのプレースをそれぞれ設け，ここにロック用トークンをマーキングすることでロックの取得・リリースを表現する．CPN の場合，このトークンはカラーと呼ばれるデータ型を付与することができる．ロックをどのように実装するかは DBMS に依存する問題であるが，基本的には排他制御するデータベースレコードのキー値が特定できる情報を含む必要がある．また，ロックの総量を低減させるため，キーをグルーピングし，ロックの粒度を上げる場合がある．本論文では，このような状況をモデル化するため，

keyHash : {key 値} → [1, N]

なるハッシュ関数により，一定範囲の自然数にマッピングし，サイズ N の文字型配列を含むカラーLock を定義す

† 龍谷大学，Ryukoku University

る．各要素は取得されているロックの種類により，S, X, ,null をとるものとする．ロックを保持するプレースを使用した DB 排他制御は図 1 のように CPN 表現される．

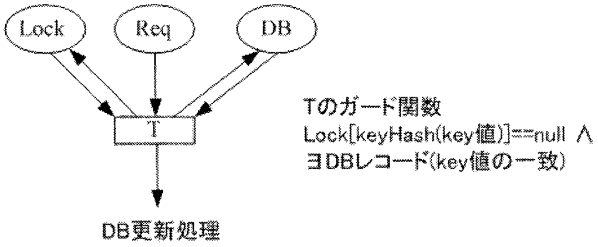

図 1. CPN による DB 排他制御表現

参照系の排他制御や，ロックリリース処理も同様にして行うことができる．

3. CPN によるトランザクション分割と評価

3.1 トランザクション分割

CPN での機能分解は，階層化ペトリネットを用いて行うことができる．分解は，トランジションに対して行う代入トランジションとプレースに対して行う代入プレースという二つの方法がある．今回は機能面での分割のため，代入トランジションを用いる．前節で導入した，トランザクション処理の CPN モデルはビジネスロジック，DB 排他制御のような DBMS 制御メカニズムおよび純粋なトランザクションが混在した形でフラットなモデルを構成するため，分割に際してはまず，これらを識別し，純粋なトランザクションを表す部分を見出す必要がある．これをもとに図 2 に示すような階層化を行い，ランザクション分割を行う．

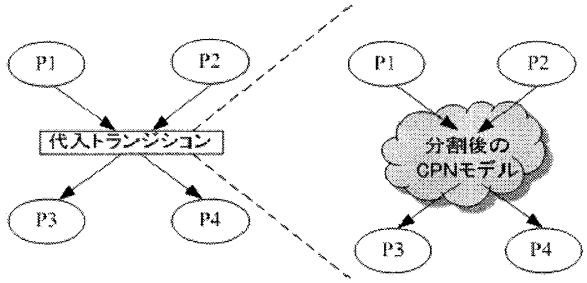

図 2. CPN によるトランザクション分割

図 2 の例では、入力プレース P1, P2 および出力プレース P3, P4 を持つトランジションがサブページに分解されている。なお，階層化は，任意回数可能である．

3.2 分割の評価

CPN 表現されたトランザクション処理モデルは，抽象レベルの違いにより，処理のテンプレートを表す場合と，処理のインスタンスを表す場合がある．テンプレートレベルでは，機能の連続性が分割評価の基準になるのに対し，インスタンスレベルでは競合や平行性など実行時の品質特性が基準となる．CPN ではアーク関数やガード関数などの機能的側面記述には関数型言語の一種である Standard ML を拡張した CPN ML を使用する．このため，分割による機能の連続性は，分割前後の入力プレースのカラートークン状態，ガード関数および出力アークのアーク関数の関数的な等価性を示すことで比較的容易に評価することが可能である．一方，インスタンスレベルにおいては，競合やデッドロック，ライブロックおよび並行度などの動作的側面を評価することになるため，シミュレーションによる検証が必要となる．CPN の場合，cpn tools [6]などのいくつかのシミュレーションツールが利用できる．インスタンスレベルのモデルはテンプレートレベルより導出することになる．この際，トランザクションのインスタンス数と入力プレースのマーキングにより多数のインスタンスレベルのモデルが存在する．これらの中から，ユースケースやシナリオ分析等を用いて，できる限り現実に近いインスタンス・モデルを選択することで検証の精度を上げる．

一方，並行度の評価では，最終的なデータベースの整合性を保証する必要があるため，直列化可能性(最終状態直列化可能性，ビュー直列化可能性，相反直列化可能性など)を検証しなければならない[7]．CPN モデルの場合，最終状態での DB プレースのマーキングおよび個々のトークンの値を比較することで，この評価が可能となる．この検証も CPN Tools により行うことができる．

4. 結び

ビジネスプロセスおよびトランザクション処理を CPN によりモデリングすることで，クラウドコンピューティングのように高度に分散化，仮想化された環境に適した，より小さな粒度を持つトランザクション単位を導出し，その機能的等価性，整合性，安全性などを CPN Tools などのツールにより半自動的に評価することが可能となる．シミュレーションを行うことで，並行度の最適化や，デッドロック，ライブロックの回避が可能となる．

今後の課題としては，機能のみでなくデータベースを含めた分解とその評価方法を確立することが挙げられる．この場合，代入トランジションだけでなく，代入プレースによる階層化が必要となる．また，最近のシステムは時間制約を含むものが多くなっており，この視点での評価も重要となる．CPN の拡張として時間カラーペトリネット(Timed Colored Petri Nets)があり，カラーセットやマーキングなどに時間概念を含めることが可能となっている．CPN Tools もこの拡張を実装しており，これらを利用することで，評価が可能になると考えられる．

参考文献

[1]John W. Rittinghouse, James F. Ransome, “Cloud Computing: Implementation, Management, and Security”, CRC Press (2009).
[2]Dan Pritchett, “BASE: An Acid Alternative”, Queue, Vol.6, No.3, pp48-55 (2007).
[3]村田 忠夫, “ペトリネットの解析と応用 (アルゴリズムシリーズ)”, 近代科学社 (1992).
[4]Kurt Jensen, Lars M. Kristensen, “Coloured Petri Nets: Modelling and Validation of Concurrent Systems”, Springer (2009).
[5]Yoshiyuki Shinkawa, Masao J. Matsumoto, “An Information System View of Consistency and Integrity in Enterprise Operations”, Kluwer Academic Publishers (2002).
[6]http://wiki.daimi.au.dk/cpntools/cpntools.wiki.
[7]増永 良文, “リレーショナルデータベース入門—データモデル・SQL・管理システム”, サイエンス社 (2003).