社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2009－ＤＰＳ－138 (9)
2009－CSEC－ 44 (9)
2009／3／5

ペトリネットにおけるセマンティックグリッド サブオントロジ抽出ワークフローの妥当性の確認

内林 俊洋† アプドゥハン・ベーナディ‡
九州産業大学情報科学研究科† 九州産業大学情報科学部‡

自然言語による処理を目的としたグリッドにセマンティックグリッドがある。セマンティックグリッドはオントロジ技術を用いたグリッドであり、処理が複雑となるオントロジの処理ではワークフローの最適化が必要となる。本論文では OntoMove プロジェクトで提案されているオントロジの下位概念であるサブオントロジの生成を行う 2 つのスキーマを、ワークフローモデルであるペトリネットを使用して設計を行い、妥当性の確認を行った。

On Using Petri Net in Sub-ontology Extraction Workflow Validation for Semantic Grid

Toshihiro Uchibayashi† Bernady O. Apduhan‡
Graduate School of Information Science, Kyushu Sangyo University, Fukuoka†
Faculty of Information Science, Kyushu Sangyo University, Fukuoka‡

The Semantic Grid is a grid environment for natural language processing. It uses ontology technology, and workflow optimization is necessary in ontology processing. In this paper, we consider two optimization schemes in sub-ontology extraction and design its workflow models using Petri Net. The workflows for each model are described and validated.

I. はじめに

現在、グリッドに関するさまざまな研究が行われ、すでに十分実用可能なレベルとなっている。その用途として地球規模の気象予測や DNA の分析など膨大な処理能力を必要とする計算、１つのサーバでは収まりきれないような膨大な情報量を要するデータの分散保管などがあげられる。そのほかにもさまざまな企業でビジネスグリッドの提案、設置が行われている。このようにグリッドはすでに様々な場所に浸透しており、今後も注目される技術の１つである。そのようなグリッドをさらに拡張しセマンティックウェブの技術と融合したものがセマンティックグリッドである。セマンティックウェブではコンピュータに情報の意味を理解させる事で、情報機器間のやり取りに人間を仲介させずにコンピュータ自身が理解出来るようにすることを可能にしている。データの意味やデータ間の関連を定義するメタデータを持たせる事で、コンピュータにその内容を理解出来るようにし、情報を自動処理させようというものである。この技術により、検索結果の精度向上や、Web 上の情報の有効利用が実現出来るとされている。このメタデータを用いる事によって、意味的に同じ用語であるが表記の異なったものを同一のものと認識出来るようになる。セマンティックグリッドでも同様にデータグリッド内の膨大な量のデータを検索する際に、メタデータの集合体であるオントロジの検索をする事で、より効率よく目的のデータへとたどり着けるようになる。今までのグリッドコンピューティングではただ計算資源や記憶容量を提供するだけであったが、セマンティックグリッドへと拡張されることによりコンピュータがデータを自分で理解し、処理することが可能となる。本論文ではオントロジからサブオントロジを抽出するスキーマに着目し、そのワークフローモデルを作成、及び妥当性の確認を行う。

II. 想定環境

セマンティックグリッド上の分散オントロジ

フレームワークの環境を図1に示す。

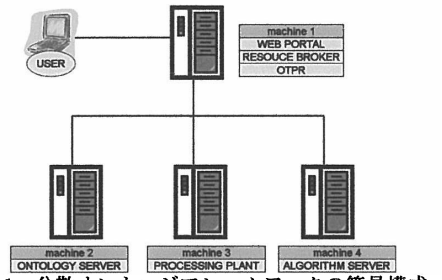

図1　分散オントロジフレームワークの簡易構成

これらは、RESOURCE BROKER, OTPR (Ontology Tailoring Processing Resource), ALGORITHM SERVER, ONTOLOGY SERVER, PROCESSING LANT で構成されている。RESOURCE BROKER はユーザに可能性、アクセス手段、セキュリティ問題や他のポリシーを気にすることなく透過的にヘテロジニアスな資源にアクセスさせるためのソフトウェアである。複数のユーザがいる場合、状況に合わせて接続先を仲介してジョブを配分する役割を持っており、最適なサービスを選択して実装する。OTPR は User から受け取った処理を ONTOLOGY SERVER にどのくらいの情報量なのかを見て PROCESSING PLANT を選別する。この選別が OTPR に返ってくると、今度はその処理に必要なアルゴリズムを ALGORITHM SERVER から受け取る。そして PROCESSING PLANT に処理内容とアルゴリズムを送り、結果を User に返す。ALGORITHM SERVER は膨大な情報量を検索するのにどのようにデータをソートすればよいかのアルゴリズムを格納している。ONTOLOGY SERVER は膨大な情報量のオントロジを格納している。PROCESSING PLANT は OTPR では処理できないような複雑な計算や処理を行う。本論文ではこの中で最も重要であり複雑な処理を行うオントロジサーバに着目し、オントロジの検索の際に作成されるサブオントロジの抽出に焦点を置く。

III.　関連研究

これまでワークフローの作成とその妥当性を調べる研究をしてきた。Workflow Scheduling Algorithms for Grid Computing [1]では、スケジューリングは分配されたリソースにおける互いに依存しているタスクの実行を写像して管理する過程であり、実行がユーザによって指定された目的関数を満たすために終了できるように、適当なリソースを割り当てるのをワークフロータスクに明記する。 適切なスケジューリングは重要な影響をシステムの性能に与えることができ、Grid プロジェクトによって開発されて、配備された既存のワークフロースケジューリングアルゴリズムを調査を行っている。これらの研究を踏まえ、ワークフローの作成及び妥当性の確認を行う。

IV. オントロジ最適化スキーマ

サブオントロジの抽出を行うスキーマに RCOS (Requirements Consistency Optimization Scheme) と 、 SCOS (Semantic Completeness Optimization Scheme)がある[2]。RCOS は様々なフォームの一貫性がないかどうかチェックする 4 つのサブスキーマの組み合わせであり、SCOS は様々なフォームの意味完全性がないかどうかチェックする 3 つのサブスキーマからなる。

A. Requirements Consistency Optimization Scheme (RCOS)

RCOS は、ユーザや他の最適化スキーマによる要件の一貫性を確実にする。ユーザによって開始されるのでラベリングは一貫している。要件の一貫性を確実にすることによって、サブオントロジが最初の場所に誘導できなくなる可能性をすべて排除する。RCOS は抽出の過程の間に適用されるべき最初の場所であり、総合的な抽出の過程の中のかなり簡単な規則の 1 つである。RCOS は 4 つの最適化スキーマの集合体であり、サブスキーマ RCOS1-RCOS4 として暗黙の命令や実行優先権なしで指示を行う。

1)　RCOS1

オントロジで示されている概念間の二項関係をユーザが選択できるなら、関係が結合している 2 つの概念が目標のオントロジから除外されることはない。

2)　RCOS2

RCOS2 の規則は RCOS1 と同様だが、概念のセットである二項関係の代わりに属性マッピングと呼ばれる属性間に存在する特別な関係の概念で適用されるということが異なる。もし属性マッピングが選択した呼び名を持っているなら、それがマッピングされる概念と同様に関連属性は目標のオントロジで結合しているものも選択されなければいけない。

3)　RCOS3

この規則は属性マッピングの特定の特性に要件を課す。それは属性マッピングに非選択された呼び名があるなら、関連属性もまた目標オントロジから除外されなければいけない。基本的に矛盾していない選択は属性マッピングと関係属性間で許容されている。RCOS2と結合して RCOS3はこの状態を課す。

4)　RCOS4

RCOS4 は RCOS1-RCOS3 のものより比較的複雑となる。 RCOS4 を例証するために経路の概念を非公式に利用する。 経路はオントロジ視点の仕様で非常に重要である。 これらは意味的に正しく、存在している外観上新しい関係をオントロジ定義に提供する。 経路は 1 つの関係の終わりの概念が連鎖における下位の関係の最初の概念と接続する関係の連鎖であると定義される。同じ関係が全体の経路に一度だけ現れることができる。属性が選択されたとき概念が非選択されるなら、属性から非選択されない別の概念まで経路があると想定する。

B. Semantic Completeness Optimization Scheme (SCOS)

多くの方法でオントロジの意味完全性の考えを解釈できるが、サブオントロジ抽出のために要件仕様を通してユーザによって選択された要素のための定義要素に含まれる。定義要素は、オントロジの別の要素のセマンティックに不可欠な概念の関係または属性である。例えば、定義要素が同時に非選択されるなら、サブオントロジに存在している選択された概念は意味的に不完全となる。シナリオに複数の遺産、集合などの複雑な関係があるときは、より複雑になるが、SCOS は、そのような矛盾を回避するために存在している。SCOS は 3 つの最適化スキーマ SCOS1-SCOS3 の集合から成る。

1) SCOS1

概念が選択されるなら、すべての上位概念および概念とその上位概念との遺産関係を選択しなければいけない。

2) SCOS2

コンセプトが選択されるなら、集合関係と共にこの概念のすべての集合部分を選択しなければいけない。

3) SCOS3

概念が選択されるなら、ゼロ以外の最小の基数と共に持っている属性とそれらの属性マッピングのすべてが選択されなければいけない。

V. ペトリネットによるワークフロー設計

RCOS と SCOS の 2 つのスキーマのワークフローをペトリネットを用いて設計を行う。 ペトリネットとは離散分散システムを数学的に表現する手法であり、プレースとトランザクションの 2 つのノードをアークで接続する。プレース p が、非負整数 k に割り当てられたとき、プレース p は k 個のトークンでマーキングされていると言い、このときトークンはプレース p 内の k 個の点として図示される。ペトリネットの妥当性の評価は、到達可能性、活性、有界性、保存性、公平性についてそれぞれ検証する。到達可能性とは M0 からどのトランザクションにもトークンが到達できるかどうか、活性は M0 からどんなマーキングに到達していようとも、ネット内の任意のトランジションを、そのマーキングから何らかの発火系列を通して発火できるのか、有界性は各プレースのトークン数が有効な数 k を超えないか、保存性は重みつきのトークン数の和が一定かどうか、公平性は継続的に発火可能なトランザクションはいつか発火する、である。

A. RCOS の設計

1) RCOS1 の設計

図 2 は RCOS1 を設計したものである。仕様は以下のようになる。プレース 1 (p1)は 1 つのトークンを保持している。トランザクション 1 (t1)はユーザが選択したか否かをチェックする。t2 はもし概念が選択されていたら P3 へ移動する。概念が非選択ならば P4 へ。t3 では関係が結合している概念は非選択にしないという処理を行う。

P={p1,p2,p3,p4,p5}
T={t1,t2,t3,t4}
F={(p1,t1),(t1,p2),(p2,t2),(t2,p3),(t2,p4),(p3,t3),(p4,t4),
(t3,p5),(t4,p5)}
W((p1,t1)=1),W((t1,p2)=1),W((p2,t2)=1),W((t2,p3)=1),W((t2,p4)=1),W((p3,t3)=1),W((p4,t4)=1),W((t3,p5)=1),W((t4,p5)=1)
M0(p1)=1

a) *到達可能性*：M0 からは t3 か t4 のどちらかのみトークンが到達可能となる。

b) *活性*：M0 からどんなマーキングに到達していようとも、ネット内の任意のトランジションを、そのマーキングから何らかの発火系列を通して発火することはできない。

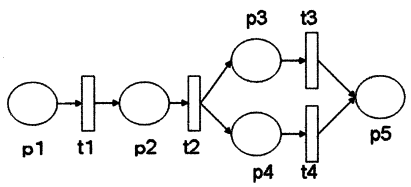

図 2 RCOS1 の設計

c) *有界性*：各プレースのトークン数が有効な数(k=1)を超えない。

d) *保存性*：重みつきのトークン数の和が常に 1 と一定。

e) *公平性*：継続的に発火可能なトランザクシ

ョンはいつか発火可能である。
f) *まとめ*：このサブスキーマは関係が結合している概念の選択矛盾を回避するために使用される。上記の検証よりこのサブスキーマの妥当性が確認された。

2) RCOS2 の設計

図 3 は RCOS2 を設計したものである。仕様は以下のようになる。プレース 1 (p1)は 1 つのトークンを保持している。トランザクション 1 (t1)はユーザが選択したか否かをチェックする。t2 はもし属性が選択されていたら P3 へ移動する。属性が非選択ならば P4 へ。t3 では関係が結合している属性は非選択にしないという処理を行う。

P={p1,p2,p3,p4,p5}
T={t1,t2,t3,t4}
F={(p1,t1),(t1,p2),(p2,t2),(t2,p3),(t2,p4),(p3,t3),(p4,t4),
(t3,p5),(t4,p5)}
W((p1,t1)=1),W((t1,p2)=1),W((p2,t2)=1),W((t2,p3)=1),W((t2,p4)=1),W((p3,t3)=1),W((p4,t4)=1),W((t3,p5)=1),W((t4,p5)=1)
M0(p1)=1

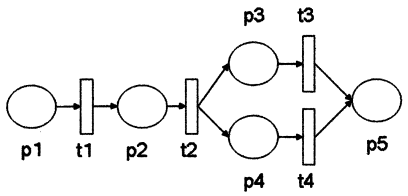

図 3 RCOS2 の設計

a) *到達可能性*：M0 からは t3 か t4 のどちらかのみトークンが到達可能となる。
b) *活性*：M0 からどんなマーキングに到達していようとも、ネット内の任意のトランジションを、そのマーキングから何らかの発火系列を通して発火することはできない。
c) *有界性*：各プレースのトークン数が有効な数(k=1)を超えない
d) *保存性*：重みつきのトークン数の和が 1 と一定。
e) *公平性*：継続的に発火可能なトランザクションはいつか発火可能である。
f) *まとめ*：RCOS1 と非常によく似ているがこのサブスキーマは概念ではなく属性を対象としている。同様にこのサブスキーマの妥当性が確認された。

3) RCOS3 の設計

図 4 は RCOS1 を設計したものである。仕様は以下のようになる。プレース 1 (p1)は 1 つのトークンを保持している。トランザクション 1 (t1)はユーザが非選択か否かをチェックする。非選択ならば p2 へ移動する。t2 では関連属性も非選択にする。

P={p1,p2,p3,p4,p5}
T={t1,t2,t3,t4}
F={(p1,t1),(t1,p2),(t1,p3),(p2,t2),(p3,t4),(t2,p4),(t3,p5)}
W((p1,t1)=1),W((t1,p2)=1),W((t1,p3)=1),W((p2,t2)=1),W((p3,t4)=1),W((t2,p4)=1),W((t3,p5)=1)
M0(p1)=1

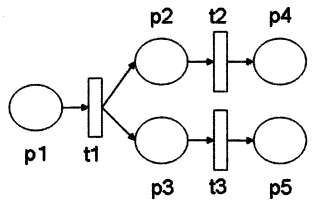

図 4 RCOS3 の設計

a) *到達可能性*：M0 から t2 か t3 のどちらかのみトークンが到達可能
b) *活性*：M0 からどんなマーキングに到達していようとも、ネット内の任意のトランジションを、そのマーキングから何らかの発火系列を通して発火できない。
c) *有界性*：各プレースのトークン数が有効な数(k=1)を超えない
d) *保存性*：重みつきのトークン数の和が 1 と一定。
e) *公平性*：継続的に発火可能なトランザクションはいつか発火可能である。
f) *まとめ*：属性マッピングにおいて非選択された際の関連属性の選択矛盾を回避するために使用される。終了プレースが 2 つあるが、非選択された場合とそうでない場合の終了プレースであるため、上記の検証より活性は満たされなかったが処理は条件と合っているため妥当性が確認される。

4) RCOS4 の設計

図 5 は RCOS2 を設計したものである。仕様は以下のようになる。プレース 1 (p1)は 1 つのトークンを保持している。トランザクション 1 (t1)はユーザが概念を選択したか否かをチェックする。選択しているなら p2 へ、選択していないなら p3 へ移動する。t2 はもし属性が非選択

ならば P4 へ移動する。t4 では他の概念の検索を行う。
P={p1,p2,p3,p4,p5,p6,p7,p8}
T={t1,t2,t3,t4,t5}
F={(p1,t1),(t1,p2),(p2,t2),(t1,p3),(t2,p4),(p3,t3),(p4,t4),
(t3,p5),(t4,p7),(t3,p6),(t5,p8)}
W((p1,t1)=1),W((t1,p2)=1),W((p2,t2)=1),W((t1,p3)=1),W((t2,p4)=1),W((p3,t3)=1),W((p4,t4)=1),W((t3,p5)=1),W((t4,p7)=1),W((t3,p6)=1),W((t5,p8)=1)
M0(p1)=1

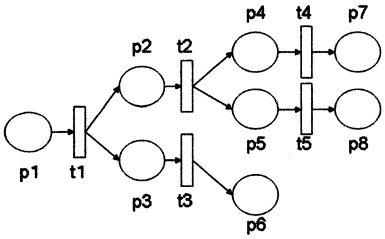

図5　RCOS4 の設計

a)　*到達可能性*：M0 からは t3, t4, t5 のいずれかのみトークンが到達可能。
b)　*活性*：M0 からどんなマーキングに到達していようとも、ネット内の任意のトランジションを、そのマーキングから何らかの発火系列を通して発火できない。
c)　*有界性*：各プレースのトークン数が有効な数(k=1)を超えない。
d)　*保存性*：重みつきのトークン数の和が 1 と一定
e)　*公平性*：継続的に発火可能なトランザクションはいつか発火可能である。
f)　*まとめ*：属性が選択されたか、された場合に概念は非選択であるかの違いで終了プレースが異なる。活性は満たされていないが問題と思われる。よって妥当性が確認される。

B.　SCOS の設計
1)　SCOS1 の設計
図 6 は SCOS1 を設計したものである。仕様は以下のようになる。プレース 1 (p1)は 1 つのトークンを保持している。トランザクション 1 (t1)はユーザが選択したか否かをチェックする。選択されていれば p2 へ移動する。t2 では上位概念及び遺産関係の概念も選択する。
P={p1,p2,p3,p4,p5}
T={t1,t2,t3,t4}
F={(p1,t1),(t1,p2),(p2,t2),(t2,p4),(t1,p3),(p3,t3),(t3,p5)}
W((p1,t1)=1),W((t1,p2)=1),W((p2,t2)=1),W((t2,p4)=1),W((t1,p3)=1),W((p3,t3)=1),W((t3,p5)=1)
M0(p1)=1

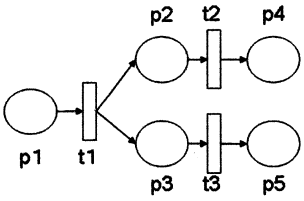

図6　SCOS1 の設計

a)　*到達可能性*：M0 から t2 か t3 のいずれかのみトークンが到達可能。
b)　*活性*：M0 からどんなマーキングに到達していようとも、ネット内の任意のトランジションを、そのマーキングから何らかの発火系列を通して発火可能ではない。
c)　*有界性*：各プレースのトークン数が有効な数(k=1)を超えない
d)　*保存性*：重みつきのトークン数の和が 1 と一定
e)　*公平性*：継続的に発火可能なトランザクションはいつか発火可能である。
f)　*まとめ*：概念が選択された場合とそうでない場合で終了プレースが異なる。活性が満たされていないが問題ないと思われる。よって、妥当性が確認される。

2)　SCOS2 の設計
図 7 は SCOS1 を設計したものである。仕様は以下のようになる。プレース 1 (p1)は 1 つのトークンを保持している。トランザクション 1 (t1)はユーザが選択したか否かをチェックする。選択されていれば p2 へ移動する。t2 では集合関係と共に概念の集合部分も選択する。
P={p1,p2,p3,p4,p5}
T={t1,t2,t3,t4}
F={(p1,t1),(t1,p2),(p2,t2),(t2,p4),(t1,p3),(p3,t3),(t3,p5)}
W((p1,t1)=1),W((t1,p2)=1),W((p2,t2)=1),W((t2,p4)=1),W((t1,p3)=1),W((p3,t3)=1),W((t3,p5)=1)
M0(p1)=1

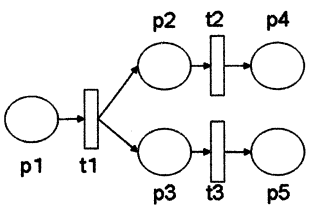

図7　SCOS2の設計

a)　*到達可能性*：M0 から t2 か t3 のいずれかのみトークンが到達可能。
b)　*活性*：M0 からどんなマーキングに到達していようとも、ネット内の任意のトランジションを、そのマーキングから何らかの発火系列を通して発火可能ではない。
c)　*有界性*：各プレースのトークン数が有効な数 k を超えない
d)　*保存性*：重みつきのトークン数の和が 1 と一定
e)　*公平性*：継続的に発火可能なトランザクションはいつか発火可能である。
f)　*まとめ*：SCOS1 と非常によく似ており、t2 での処理が異なるだけでワークフローは同様のものになる。よって、妥当性が確認される。

3)　SCOS3 の設計

図 8 は SCOS1 を設計したものである。仕様は以下のようになる。プレース 1 (p1)は 1 つのトークンを保持している。トランザクション 1 (t1)はユーザが選択したか否かをチェックする。選択されていれば p2 へ移動する。t2 ではゼロ以外の最小の基数と共に持っている属性とそれらの属性マッピングのすべてを選択する。

P={p1,p2,p3 }
T={t1,t2}
F={(p1,t1),(t1,p2),(p2,t2),(t2,p3) }
W((p1,t1)=1),W((t1,p2)=1),W((p2,t2)=1),W((t2,p3)=1)
M0(p1)=1

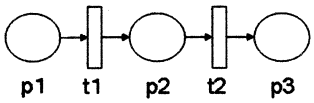

図8　SCOS3 の設計

a)　*到達可能性*：M0 からどのトランザクションにもトークンが到達可能。
b)　*活性*：M0 からどんなマーキングに到達していようとも、ネット内の任意のトランジションを、そのマーキングから何らかの発火系列を通して発火可能。
c)　*有界性*：各プレースのトークン数が有効な数 k を超えない。
d)　*保存性*：重みつきのトークン数の和が 1 と一定。
e)　*公平性*：継続的に発火可能なトランザクションはいつか発火可能である。
f)　*まとめ*：すべての検証が満たされており、妥当性を確認できる。

VI. まとめ

サブオントロジ抽出を行う RCOS と SCOS の各サブスキーマのワークフローの設計をペトリネットで行い妥当性の検証を行った。SCOS3 を除いて活性を満たすことができなかったが、それぞれに妥当性が確認できた。検証により妥当性の確認が行えたことで、複数のオントロジが存在するような大規模なセマンティックグリッド環境においてもサブオントロジ抽出は問題ないと思われる。今後は複数のオントロジを使用した際のサブオントロジ抽出の検証や実環境での検証を行いたい。

VII. 参考文献

[1] Jia Yu , and Rajkumar Buyya, “Workflow Schdeduling Algorithms f-or Grid Computing”, Technical Repo-rt, GRIDS-TR-2007-10, Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Aus-tralia, May 31, 2007.

[2] Mehul Bhatt, Andrew Flahive, Carlo Wouters, J. Wenny Rahayu and David Taniar, “MOVE: A Distributed Framework for Materialized Ontology View Extraction”, Algorithmica 45(3), pp. 457-481, 2006.

[3] Feilong Tang, Minglu Li, and Minyi Guo, “Transactional grid workflow service for ShanghaiGrid”, Int. J. Web and Grid Services, Vol. 3, No. 4, 2007.

[4] Andrew Flahive, J. Wenny Rahayu, Bernady O. Apduhan and David Taniar, “Simulating the Distributed Ontolo-gy Framework in the Semantic Grid Environment with GridSim”, In Proceedings of PDPTA 2006, pp. 717-723.