Vol.2010-DD-74 No.5
2010/1/29

セマンティック Web 技術を用いた PC パーツの検索

長谷川明史† 西村紅美† 塚本享治†

Web 上に存在する様々な情報や知識を関連付け活用する技術としてセマンティック Web が提案され，関係を OWL で記述し推論を行うことで新しい知識を導くことができる．この OWL による推論を利用し，マザーボード，CPU，メインメモリの組み合わせを検索する例を作成した．
各パーツと規格，組み合わせの関係を OWL で記述し，Web 上から取得した実際の製品情報に対して，推論を行い，適切なパーツを調べた．

Search and Inference of Computer Parts by Semantic Web Technology

Akifumi Hasegawa† Kumi Nishimura† Michiharu Tsukamoto†

Semantic web is proposed as a technology to establish relations among various information and knowledge on the web. According to the technology, we are able to inference new knowledge from asserted relations in OWL. In this paper we describe a case study on modeling and reasoning on PC parts.
At first we describe schemas and relationships on abstract PC parts. Secondary we collect actual product data from the internet. Thirdly we actuate the pellet reasoner over the schemas and individuals. Finally, we search the induced triples using SPARQL queries.

1. はじめに

現在の Web 上にはさまざまな情報が存在しているが，それらは相互に関連づけられているわけではなく，独立して存在している．そこで，この情報を関連付け，活用するための技術として，セマンティック Web という技術が提案されている．セマンティック Web では Web Ontology Language[1]（OWL）いることで情報を相互に関連けることや新しい情報を導くことが可能になる．

† 東京工科大学大学院 バイオ・情報メディア研究科
Tokyo University of Technology Graduate School

本稿では OWL を用いて新しいトリプルを導く例題として，PC パーツの組み合わせについて取り上げる．基準となるマザーボードに対応している CPU，メインメモリを OWL の推論を使って求める．そのため，まずそれぞれのパーツとそのパーツが持っている規格との関係を OWL でモデリングする．次に，目的のパーツを求める推論用の OWL を記述する．Web 上から取得したパーツの情報と OWL を合わせることで，対応するパーツを求める．

2. 推論を目的とした OWL の記述

2.1 推論対象の設定

OWL は，セマンティック Web 技術の中で重要な位置を占め，語彙やデータ間の関係を定義することで，それらの関係から新しい関係を発見したり，語彙間の対応関係をとったりすることができる．この OWL を使った例として，W3C が公開している食品とワインのオントロジ[2][3]や，[4]中で例題として取り上げられているカクテルのオントロジなどがあげられる．しかし，それらで取り上げられている分野は技術系大学の講義で用いるには分野が不適切であったり，モデリングに偏っているものが多く，推論に関するものが少ない．

そこで，大学の講義でも利用できるテーマとして，適切な PC パーツを推論によって選択することを取り上げ，セマンティック Web の推論によって答えを見つける例題の 1 つとして OWL を記述した．

2.2 RDF 記述方法

セマンティック Web のデータ構造である RDF[5]は，ラベル付き有向グラフの構造をしているが，その記述方法はいくつか存在し，XML 形式で書く RDF/XML[6]，テキスト形式で記述する Notation3[7]（N3）などがあげられる．ここでは，RDF/XML より読みやすい N3 で表記する．

N3 形式ではトリプルは“:Subject :predicate :Object.”と表記し，ラベル付き有向グラフとして図示すると図 1 のようになる．

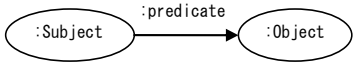

図 1 トリプルの例

このとき，ノードについている“:”では，URI が省略されており，本稿中では http://www.teu.ac.jp/g3109018/pc/を表している．RDF はノードやエッジを URI で識別する．したがって，:Subject は <http://www.teu.ac.jp/g3109018/pc/Subject> という URI を持つノードを表す．

2.3 セマンティック Web の推論

セマンティック Web での推論とは，RDFS[8]や OWL によって概念間の関係を表現し，その関係に基づいて新しいトリプルを導く処理のことである．ここで発見された新しいトリプルは RDF グラフに追加され, はじめから存在していたトリプルと区別することなく扱うことができる．たとえば， owl:equivalentClass は等価なクラスを定義することができるため，:InstanceA rdf:type :ClassA. （:InstanceA は :ClassA のインスタンスである）というトリプルに対して, :ClassA owl:equivalentClass :ClassB.（ :ClassA と :ClassB は等価である）というルールを用いて，新しく :InstanceA rdf:type :ClassB .（:InstanceA は :ClassB のインスタンスである）というトリプルを導くことができる (図 2).

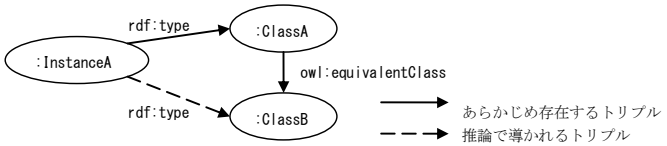

図 2　新しくトリプルを導く例

3. パーツと規格の OWL の記述と語彙の定義

PC のパーツはメーカーが異なるもの同士の組み合わせでも正しく動作するように，接続方式や伝送方式が，いくつかの規格として標準化されている．したがって，組み合わされるそれぞれのパーツが共通の規格に基づいていれば正しく動作すると考えられる．

ここでは，PC パーツそれぞれが，自身の対応している規格を持っているので，それを利用する．組み合わされるパーツ間でお互いの持っている規格が一致する場合を適切な組み合わせとし，そうでない場合を不適切な組み合わせとする．

ここでは，PC の最も主要なパーツである CPU，メインメモリ，マザーボードそれぞれに対して，パーツや規格の関係を OWL で表す.

3.1 各パーツとその規格クラス

パーツについてオントロジを記述するため，まず PC のパーツとそれらの持つ規格について OWL によるモデリングを行った.

一般的な PC パーツのクラスとして， :Parts を定義した．マザーボードのクラス（ :Motherboard ），メインメモリのクラス（ :Memory ），CPU のクラス（ :CPU ）は， rdfs:subClassOf プロパティを用いて :Parts のサブクラスとして定義する．これらパーツのクラスは，実際の製品を表しているのではなく，その製品の集合を表している．ある特定の製品，たとえば “Core 2 Duo E8400”という実際の製品は CPU クラスのインスタンスになる．これは，CPU の集合の中の要素として，“Core 2 Duo E8400” という製品があるとみることもできる.

マザーボードと CPU，マザーボードとメインメモリの適切な組み合わせは，共通規格を持っているかどうかで判断する．そこで，パーツ同様に一般的な規格クラスとして :Standard を定義する．:Standard のサブクラスには，マザーボードと CPU の間で関係のある規格として， :CPUSocket と:CPUBus の２つのクラスを，マザーボードとメインメモリとの間の規格として，:MemorySlot ，:MemoryChip， :MemoryModule の３つのクラスを定義した．それぞれのクラスの意味は表 1 に示す.

この規格クラスもパーツクラス同様に，特定の規格を表すのではなく，規格の集合を表し, 特定の規格は :Standard のインスタンスとして表す. たとえば,“DDR333” という実際の規格は :MemoryChip というメモリチップ規格クラスのインスタンスとして表すことができる.

表 1　定義したクラスの親子関係とその意味

クラス	親クラス	意味
:Parts		パーツ全般
:CPU	:Parts	CPU
:Memory	:Parts	メインメモリ
:Motherboard	:Parts	マザーボード
:Standard		規格全般
:CPUSocket	:Standard	CPUソケット規格
:CPUBus	:Standard	CPUバス規格
:MemorySlot	:Standard	メモリスロット規格
:MemoryChip	:Standard	メモリチップ規格
:MemoryModule	:Standard	メモリモジュール規格

さらに， :hasStandard というプロパティを定義し，これを用いて :Parts のインスタンスが :Standard のインスタンスを持っているということを表現する．たとえば，次のトリプルによって， ExampleMemory が DDR333 という規格を持っていることを表すことができる.

```
:ExampleMemory :hasStandard :DDR333.
```

3.2 パーツと規格との関係

実際には製品が規格を持っていることを :hasStandard プロパティで直接表わすことはせず， 関係する製品と規格に応じて :hasStandard のサブプロパティを定義して用いる．この :hasStandard と，そのサブプロパティは表 2 であり，それぞれのプロパティには定義域と値域を定める.

定義域は rdfs:domain によって定義することができ，このプロパティを述語にするトリプルの主語は rdfs:domain で示されたクラスのインスタンスになる．同じように，rdfs:range によってトリプルの目的語が所属するクラスを指定する．

表 2 中のプロパティは，パーツが規格を持っている，ということを表すためのプロパティであるため，定義域はパーツのサブクラス，値域は規格のサブクラスである．

表 2 規格を持つことを表すプロパティ

定義域(rdfs:domain)	プロパティ	値域(rdfs:range)
:Parts	:hasStandard	:Standard
:CPU	:hasCPUSocket	:CPUSocket
:CPU	:hasBus	:CPUBus
:Memory	:hasMemorySlot	:MemorySlot
:Memory	:hasMemoryChip	:MemoryChip
:Memory	:hasMemoryModule	:MemoryModule
:Motherboard	:acceptSocket	:CPUSocket
:Motherboard	:acceptBus	:CPUBus
:Motherboard	:acceptMemorySlot	:MemorySlot
:Motherboard	:acceptMemoryChip	:MemoryChip
:Motherboard	:acceptMemoryModule	:MemoryModule

3.2.1 マザーボードと CPU 間の規格

CPU がマザーボード上で正しく動作するためにはまず, CPU がマザーボード上に物理的に取り付けられる，という条件を満たさなければならない．CPU メーカーや製品によってピンの数や形が異なっているため，マザーボードにあう CPU，CPU にあうマザーボードを選ばなければならない．

この CPU とマザーボードの接点は CPU ソケットである．これには，いくつかの規格が存在している．たとえば，CPU の大手メーカーである Intel では Socket478 や LGA775 といった CPU ソケットがあり，同じく大手メーカーの AMD には SocketAM3 などの CPU ソケットがある．

このソケットを規格として扱うため，:Standard クラスのサブクラスとして，CPU ソケットクラス（ :CPUSocket ）を定義する．また，CPU が CPU ソケット規格を持つ時のプロパティを :hasCPUSocket と，マザーボードが CPU ソケット規格を持つ時のプロパティを :acceptSocket と定義する．

CPU ソケットが一致していても, CPU を正しく動作させることができない場合がある. Intel の CPU の場合, CPU とマザーボードとのデータ伝送に FSB（Front Side Bus ）を用いているものがある．FSB にはクロック数が決められているため，このクロック数も，CPU 側とマザーボード側で一致させなければならない．そこで，このデータ転送用のバスの規格クラスを :CPUBus として定義し，CPU ならば :hasBus，マザーボードならば :acceptBus というプロパティによってパーツに規格を持たせる．

Intel と AMD では，メモリコントローラの位置なども異なる．AMD は CPU 側にメモリコントローラが置かれている．一方 Intel は CPU 側にある製品とマザーボード側にある製品が存在する．本来このような違いもパーツ選択の際には考慮しなければならないが，CPU ソケットや CPU バスが決定すればメモリコントローラがどちら側にあるかわかるため，CPU ソケットと CPU バスだけで判断することができる．

3.2.2 マザーボードとメインメモリ間の規格

CPU 同様，メインメモリもマザーボードに取り付けられるパーツである．したがって，マザーボード上に取り付けられる，ということが適切な組み合わせの第一の条件になる．

一般的な PC に用いられるメインメモリは，DIMM（ Dual In-line Memory Module ）と呼ばれ，複数のメモリチップが基盤に取り付けられた構造をしている．このメモリモジュールをマザーボードのメモリスロットに差し込むことで，メインメモリを取り付ける．DIMM に取り付けられているメモリチップには DDR，DDR2，DDR3 という３種類の規格が存在し，それぞれ動作方式が異なるために互換性はなく，メモリモジュールも，それぞれピンの数や切り欠きの位置を変えているため，マザーボード側の DDR2 用のメモリスロットに DDR3 のメインメモリを取り付けるということはできない．さらに，ノート PC に使われるメモリモジュールには，DIMM よりも小型な SO-DIMM （Small Outline DIMM ） というメモリモジュールが使われるが，DIMM 用のメモリスロットに差し込むことはできない．

このことから, まずメモリスロットの規格によって分類することができると考えた．そこで，メモリスロット規格クラスとして， :MemorySlot を :Standard クラスのサブクラスとして定義した．このクラスのインスタンスは，DDR, DDR2, DDR3 のうちの一つと， DIMM, SO-DIMM のどちらか一つとの組み合わせとする．すなわち，:DDR-DIMM, :DDR2-DIMM :DDR3-DIMM, :DDR-SODIMM, :DDR2-SODIMM, :DDR3-SODIMM のいずれかのメモリスロットの規格を持つことになる．メインメモリがメモリスロットクラスのインスタンスを持つ時は， :hasMemorySlot というプロパティを用いる．これは，メモリの形状に対応しているため，複数の規格を持つことはできない．一方で，マザーボードには，DDR2-DIMM 用のスロットと DDR3-DIMM 用のスロットを持つ製品もあり， :acceptMemorySlot を複数持っているマザーボードインスタンスもありえる．

メモリスロットの規格以外に，メインメモリにはチップ規格，モジュール規格という規格がある．チップ規格はメインメモリの動作周波数によってきめられている．一方，モジュール規格はメモリの伝送速度によってきめられている．この規格がマザーボードと一致しない場合，やはり正しく動作させることができない．

メモリチップ規格クラスは :MemoryChip とし，メインメモリがチップ規格を持つ

場合に :hasMemoryChip プロパティを，マザーボードがメモリチップ規格を持つ場合に :acceptMemoryChip プロパティを用いる．

メモリモジュール規格クラスは :MemoryModule とし，メインメモリがチップ規格を持つ場合に :hasMemoryModule プロパティを，マザーボードがメモリモジュール規格を持つ場合に :acceptMemoryModule プロパティを用いて表す．

この動作周波数と伝送速度は 1 対 1 で対応している．たとえば，チップ規格 DDR2-800 を持つメインメモリは，モジュール規格 PC2-6400 を持っている．このように，どちらか一方の規格がわかれば，対応するもう一方の規格は導くことができる．パーツのデータを用意するときから統一して扱うこともできるが，ここでは２つの規格クラスを定義し，OWL 推論によって対応関係をつけていく．

さらに，メインメモリは，メモリチップ規格とメモリモジュール規格に対して上位互換がある．たとえば，DDR2-800 規格に対応している製品は，それより性能の劣る規格 DDR2-667 として動作することができる．これも，OWL 推論によって上位互換の表現を行う．

3.3 規格クラス間の関係

3.3.1 メモリチップ規格とメモリモジュール規格の対応付け

メモリチップ規格とメモリモジュール規格は表 3 のような対応関係にあり，一方の規格がわかればもう一方の規格もわかる．そこで，一方が与えられたときに他方を推論で補うため，各規格インスタンスの対応関係を OWL で記述する．

たとえば :MemoryA というあるメインメモリの製品が，:PC2-6400 というモジュール規格を持っているとき， :MemoryA は :DDR2-800 をチップ規格として持っていることもわかる．このことを表したのが図 3 であり，実線矢印のトリプルしか与えられてなくても，チップ規格とモジュール規格の関係性から破線矢印を求めることができる．

表 3 メモリチップ規格とメモリモジュールの対応

チップ規格	モジュール規格	チップ規格	モジュール規格
DDR266	PC2100	DDR2-1200	PC2-9600
DDR333	PC2700	DDR3-800	PC3-6400
DDR400	PC3200	DDR3-1066	PC3-8500
DDR2-400	PC2-3200	DDR3-1333	PC3-10600
DDR2-533	PC2-4200	DDR3-1600	PC3-12800
DDR2-667	PC2-5300	DDR3-1800	PC3-14400
DDR2-800	PC2-6400	DDR3-2000	PC3-16000
DDR2-1066	PC2-8500	DDR3-2133	PC3-17066
DDR2-1150	PC2-9200		

この対応関係を実現するため，制約によってクラスを定義する owl:Restriction と，2 つのクラスが等価だと表す owl:equivalentClass を用いる．まず，“ :hasMemoryModule の値に :PC2-6400 を持つ”という制約クラスを作る．同様に，“ :hasMemoryChip の値に :DDR2-800 を持つ”という制約クラスを作る．最後にこの 2 つのクラスが等価であることを owl:equivalentClass を使って表す(図 4).

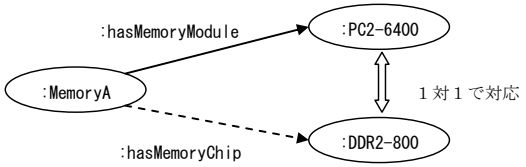

図 3 メモリモジュール規格からメモリチップ規格を推論

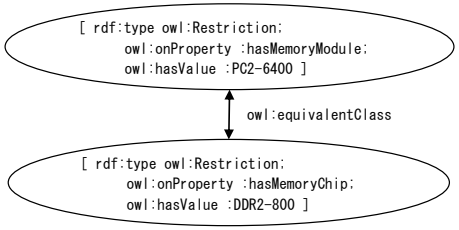

図 4 モジュール規格とチップ規格を対応させるための制約クラス

owl:Restriction によって定義される制約クラスは，その制約を満たすすべての個体をその制約クラスのインスタンスとみなす．また，そのクラスのインスタンスには制約条件を満足しているとみなされる．つまり， rdf:type を使って明示的にあるクラスのインスタンスであると表さなくても，制約を満たした時点で暗黙にそのクラスのインスタンスとなる．また同時に，明示的に制約の条件を満足していなくても，制約クラスのインスタンスになった時点で，暗黙に制約条件が満たされる．

:MemoryA :hasMemoryModule :PC2-6400. というトリプルがあった場合，:MemoryA は“ :hasMemoryModule の値に :PC2-6400 を持つ”という制約クラス条件を満たしているため，この制約クラスのインスタンスになる．さらに，この制約クラスと等価な，“ :hasMemoryChip の値に :DDR2-800 を持つ”という制約クラスのインスタンスでもあるため，結果として :MemoryA :hasMemoryChip :DDR2-800. というトリプルを導くことができる．このようにして，すべての対応関係について，OWL の制約クラスを記述した．

3.3.2 メモリチップ規格の上位互換表現

メモリチップには上位互換性があり，上位規格のメインメモリは下位規格のメインメモリとして扱うことができる．この互換性は，上位規格を持っているメモリは下位規格を持っているという制約を用いてこの互換性を表現する．

たとえば，MemoryA が :DDR2-800 という規格を持っているとき，同様に下位規格である :DDR2-667 を持ち，さらに下位規格である DDR2-533 を持つ(図 5)．このようなモジュール規格の関係を OWL で記述する．

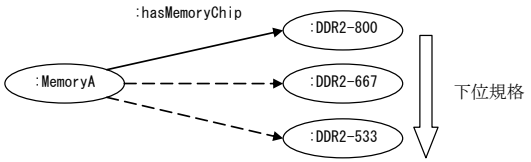

図 5　上位互換の推論例

メモリチップの互換性を OWL で書くには，制約クラスを定義する owl:Restriction と，クラスの親子関係を定義する owl:subClassOf を用いる．まず，owl:Restriction を用いて“ :hasMemoryChip の値に :DDR2-800 を持つ ”という制約クラスと，“ :hasMemoryChip の値に :DDR2-677 を持つ ”という 2 つの制約クラスも定義する．最後に 2 つの制約クラスのうち，下位規格を持つクラスが親クラス，上位規格を持つクラスが子クラスとなるように， owl:subClassOf を使って親子関係を定義する(図 6)．

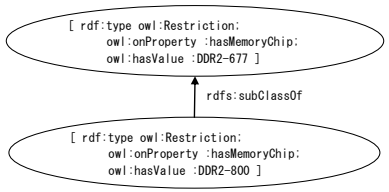

図 6　上位互換の関係を導くための制約クラス

owl:subClassOf によって作られた親子関係において，子クラスのインスタンスは親クラスのインスタンスでもあると推論される．制約クラスに対して親子関係を定義することで，子クラスの制約条件を満たしていれば，親クラスの条件を満たしているという推論が行える．

:MemoryA :hasMemoryChip :DDR2-800. というトリプルがあった場合，制約クラスの条件を満たしていることから，:MemoryA は“ :hasMemoryChip の値に :DDR2-800 を持つ ”という子クラスのインスタンスと推論される．また，子クラスのインスタンスは同時に親クラスのインスタンスであるため，“ :hasMemoryChip の値が :DDR2-677 である”という親クラスの制約を満たしていることになる．したがって， :MemoryA :hasMemoryChip :DDR2-677. というトリプルを推論によって導くことができる．

4. 推論によるパーツ検索のための OWL 記述

実際に PC の組み立てを行う場合，何か基準とする製品を選び，それに対応している製品を選んでいく．そこで，これまで定義してきたパーツクラスや規格クラスを基にして，どのような時にパーツの組み合わせが可能なのか OWL で記述する．

4.1 推論用クラスの用意

マザーボードの中から 1 つの製品を基準として選択し, それに対応する CPU とメインメモリを推論によって導くための OWL を記述する．

まず，基準となるマザーボードクラス（ :BaseMotherboard ） を :Motherboard のサブクラスとして定義する．また，対応するメモリのクラス（ :AcceptMemory ）を :Memory クラスのサブクラスとして定義し, 対応する CPU クラス（ :AcceptCPU ）を :CPU クラスのサブクラスとして定義する．

あるマザーボードの製品を :BaseMotherboard のインスタンスにしたとき, それに対応するメインメモリと CPU の製品が :AcceptMemory ， :AcceptCPU のインスタンスとして導かれるように，それぞれのクラスに対して制約の記述を行う．

4.2 :BaseMotherboard クラスの制約

マザーボードと他のパーツとのの組み合わせが正しいかどうかは，同じ規格を持っているかどうかで判断する. そこで, まず :BaseMotherboard の持っている規格を，選択の基準になる規格として用意した特別なクラスに属させる．そのために，:BaseMotherboard に次の 4 つの制約をつける．

1. この制約クラスのインスタンスが :acceptMemorySlot プロパティで参照している値はすべて :BaseMemorySlot クラスのインスタンスである
2. この制約クラスのインスタンスが :acceptMemoryModule プロパティで参照している値はすべて :BaseMemoryModule クラスのインスタンスである
3. この制約クラスのインスタンスが :acceptSocket プロパティで参照している値はすべて :BaseCPUSocket クラスのインスタンスである
4. この制約クラスのインスタンスが :acceptCPUBus プロパティで参照している値は :BaseCPUBus クラスのインスタンスである

この， :acceptMemorySlot プロパティの値すべてが :BaseMemorySlot であるという

制約は， owl:allValuesFrom を使うことで定義できる．

:BaseMotherboard クラスは，owl:intersectionOf を用いて 4 つの制約クラスの積集合として定義する．これによって，:BaseMoathoerboard のインスタンスは，4 つすべての制約を満たさなければならない．

図 7 は :BoardA という製品をパーツ選択の選択基準に選んだときの例である．図中の:BoardA は持っている規格それぞれが選択基準規格のクラスのインスタンスとして推論される．

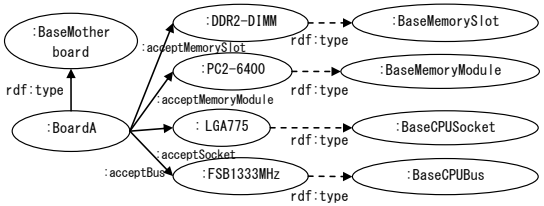

図 7　各基準規格クラスへのトリプルを導く

たとえば，この :BoardA は，:acceptMemoryModule プロパティの値に :DDR2-DIMM を持つ．したがって， :DDR2-DIMM は :BaseMemorySlot のインスタンスであると推論され，次のトリプルが導かれる．

:DDR2-DIMM　rdf:type　:BaseMemorySlot.

メモリスロットの規格だけでなく，他の 3 つの規格も同様に推論される．これら，パーツ選択の基準となる規格を決めることで，どのメインメモリを選べばいいか，どの CPU を選べばいいかがわかる．

4.3 :AcceptMemory クラスの制約

パーツ選択の基準となる規格は :BaseMotherboard の制約条件によって推論することができた．ここではさらに，実際にそれらの規格を持つメインメモリも推論を使って求める．

推論によって :BaseMotherboard に取り付けることのできるメモリスロットの規格は :BaseMemorySlot のインスタンスになっている．同様に，取り付けられるメモリモジュールの規格は :BaseMemoryModule のインスタンスになっている．そのため，:BaseMemorySlot の規格と :BaseMemoryModule の規格の両方を持っているメインメモリを :AcceptMemory のインスタンスとして導けばよい．そこで，次の 2 つの制約を :AcceptMemory につける．

1. この制約クラスのインスタンスが :hasMemoryModule プロパティで参照している値のうち，少なくとも 1 つは :BaseMemoryModule のインスタンスである
2. この制約クラスのインスタンスが :hasMemorySlot プロパティで参照している値のうち，少なくとも 1 つは :BaseMemorySlot のインスタンスである

少なくとも 1 つは指定したクラスのインスタンスを持っている，ということは，owl:someValuesFrom を使って定義することができる．:AcceptMemory は，この owl:someValuesFrom によって作られた 2 つの制約クラスの積集合として定義する．図 8 は :MemoryA が :AcceptMemory クラスのインスタンスとして推論されるときの例である． :DDR2-DIMM は :BaseMemorySlot のインスタンスであり， :PC2-6400 は :BaseMemoryModule のインスタンスである．ここで， :MemoryA は 1，2 の制約を満たしているため， :AcceptMemory のインスタンスとして推論される．

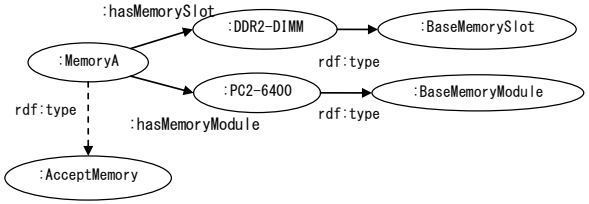

図 8　:AcceptMemory クラスのインスタンスを導く

4.4 :AcceptCPU クラスの制約

マザーボードに取り付けられる CPU のインスタンスは，推論によって:AcceptCPU のインスタンスとなる．これを求めるためには，:AcceptMemory のインスタンスを求めたとき同様に考えればよい．

:BaseMotherboard に取り付けることのできる CPU は，ソケットの規格として :BaseCPUSocket のインスタンスを持ち，CPU バスの規格として :BaseCPUBus のインスタンスを持っているものである．したがって，:AcceptCPU が満たさなければならない制約は，次の 2 つである．

1. この制約クラスのインスタンスが :hasSocket プロパティで参照している値のうち，少なくとも 1 つは :BaseCPUSocket のインスタンスである．
2. この制約クラスのインスタンスが :hasBus プロパティで参照している値のうち，少なくとも 1 つは :BaseCPUBus のインスタンスである

1，2 それぞれの制約クラスを定義し，owl:intersection によって制約クラスの積集合部分を :AcceptCPU とする．これによって， :BaseCPUSocket の規格と :BaseCPUBus の規格をもつ CPU が :AcceptCPU のインスタンスとして推論で導かれる(図 9).

Vol.2010-DD-74 No.5
2010/1/29

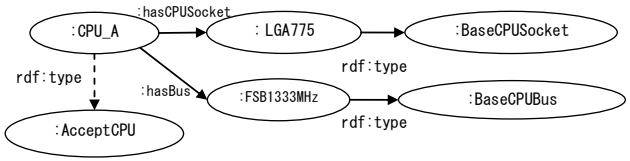

図 9　:AcceptCPU クラスのインスタンスを導く

5. 実際のＰＣパーツに対しての実験

5.1 PC パーツ情報の準備

これまで述べてきた PC パーツのオントロジを用いて, 実際に PC パーツの検索を行った．このとき PC パーツの情報は，PC パーツの小売店であるパソコン工房の Web ページ[9]から 2009 年 7 月 10 日に取得した．各パーツの情報は以下のように，パーツ 1 つ 1 つがどのような規格を持っているか N3 形式の RDF で記述した(リスト 1).

リスト 1 Web から取得した情報の RDF 化

```
board:224215    rdf:type                :Motherboard;
                :acceptMemorySlot       :DDR2-SDRAM;
                :acceptMemoryModule     :PC2-6400;
                :acceptSocket           :LGA775;
                :acceptBus              :FSB1333MHz, :FSB1066MHz,
                                        :FSB800MHz;
                :name                   "DG45FC".
```

このとき， board:224215 というのがマザーボードの製品をあらわす URI であり，省略せずに記述すると<http://www.teu.ac.jp/g3109018/pc/board/224215> を表す．このときの 224215 という番号は，パーツの情報が載っていた Web ページに振ってあった番号である. 実際に検索を行うことを考えた場合, こちらで決めた製品の URI だけでなく, 製品名もわかったほうがいいと考え，:name プロパティを使って各製品の製品名を表した．最終的 RDF の規格数，パーツ数は表 4 のようになった.

表４　各パーツ，規格クラスのインスタンス数

パーツの種類	インスタンス数
:Memory	51
:CPU	22
:Motherboard	24

規格の種類	インスタンス数
:MemorySlot	6
:MemoryModule	17
:MemoryChip	17
:CPUSocket	8
:CPUBus	7

5.2 推論によるトリプルの増加

実際に目的のパーツを推論するためには，まず組み合わせの基準となるマザーボードを決めなければならない．ここでは， board:224215 の URI をもつマザーボードを検索の基準とする．このとき，次のトリプルを用意する.

board:224215　rdf:type　:BaseMotherboard.

このトリプルと，Web 上から取得したパーツの RDF，パーツ推論用オントロジを合わせて推論することで, board:224215 に取り付けられるメインメモリと CPU を推論で求めることができる．このとき，推論を行う推論エンジンに Pellet 2.0[10]を用いる．このときの各トリプル数を整理したのが表 5 である. 推論前の RDF グラフは 979 トリプルで構成されていたが, Pellet2.0 によって推論を行ったところ, トリプルに増えた．この増加した 2353 トリプルが推論によって新たに RDF グラフに追加されたトリプルであり, この中に board:224215 に対応しているパーツを表すトリプルも含まれている.

表 5　トリプル数の比較

RDFグラフ		トリプル数	
推論前のRDFグラフ	パーツのオントロジ	320	979
	推論のためのOWL	58	
	マザーボード	227	
	CPU	112	
	メインメモリ	261	
	基準マザーボード指定	1	
推論後RDFグラフ			3332

5.3 推論結果の検索

セマンティック Web の推論は RDF グラフに新しいトリプルを追加する処理であるため，推論後の RDF グラフから必要とする推論結果を取り出す必要がある．そこで，RDF クエリ言語である SPARQL[11]を用いる．SPARQL は RDF グラフ中からクエリ文で指定されたグラフパターンを探す．リスト 2 は推論結果から :AcceptMemory のインスタンスと，その名前を求めるための SPARQL であり，推論結果として表 6 にある :AcceptMemory のインスタンスが得られた．同様に，:AcceptCPU のインスタンスをリスト３のクエリによって求めたところ，表 7 のような結果が得られた.

リスト 2　メモリに関する推論結果を検索するための SPARQL

```
SELECT   ?memory     ?name
WHERE{
        ?memory      rdf:type    :AcceptMemory;
                     :name       ?name.
}
```

表 6 :AcceptMemory　のインスタンス

?memory	?name
216533	"W2U800CQ-2GL5J (DDR2 PC2-6400 2GB 2枚組)"
247881	"FSH800D2C-K4G"
176994	"D2U800CQ-1GLZJ (DDR2 PC2-6400 1GB)"
220953	"GB24GB6400C5DC (DDR2 PC2-6400 2GB 2枚組)"
240512	"Pulsar DCSSDDR2-2GB-1066OC (DDR2 PC2-8500 1GB 2枚組)"
210610	"TEDD4096M800HC5DC (DDR2 PC2-6400 2GB 2枚組)"
211523	"JM4GDDR2-8K (DDR2 PC2-6400 2GB 2枚組)"
232960	"W2U800CF-2GHZJ"
230793	"Pulsar DCDDR2-4GB-1066OC (DDR2 PC2-8500 2GB 2枚組)"
211524	"Castor LoDDR2-2GB-800-R1 (DDR2 PC2-6400 2GB)"
212971	"Pulsar DCDDR2-4GB-800 (DDR2 PC2-6400 2GB 2枚組)"
212969	"Castor SSDDR2-1G-800 (DDR2 PC2-6400 1GB)"
203265	"JM2GDDR2-8K (DDR2 PC2-6400 1GB 2枚組)"
232959	"W2U800CF-S1GHZJ"
227815	"TEDD4096M800C5DC"
214036	"Pulsar DCSSDDR2-2GB-800 (DDR2 PC2-6400 1GB 2枚組)"
225615	"W2U1066DQ-1GLZJ (DDR2 PC2-8500 1GB 2枚組)"

リスト 3 メモリに関する推論結果を検索するための SPARQL

```
SELECT ?cpu ?name
WHERE{
        ?cpu            rdf:type    :AcceptCPU;
                        :name       ?name
}
```

表 7　:AcceptCPU のインスタンス

?cpu	?name
217337	"Core 2 Quad Q9550 BOX"
213270	"Core 2 Duo E8500 BOX"
245674	"Core 2 Duo E7600 BOX"
223962	"Core 2 Quad Q9650 BOX"
213269	"Core 2 Duo E8400 BOX"
242630	"Core 2 Quad Q8400 BOX"
227185	"Core 2 Duo E7400 BOX"
223957	"Core 2 Duo E8600 BOX"
225020	"Core 2 Quad Q8200 BOX"
243024	"Pentium Dual-Core E6300 BOX"
203304	"Celeron 430 BOX"

6. OWL 記述と検索についての評価・考察

セマンティック Web 技術である OWL の推論を用いることで，マザーボードからメインメモリと CPU を推論で求めることができ，さらに，上位互換や規格の対応関係も適切に記述することができた．とくに，互換性の問題は，単純な RDF 検索だけではクエリが複雑化してしまうことが考えられるため，OWL を効果的に用いて互換性を表せたために，非常にシンプルなクエリで柔軟な検索が行える．

しかし，情報が欠けているために本来ならば動作する組み合わせが検索結果から漏れるということもあった．これは，パーツの情報を取得してきた Web サイトでは，マザーボードが対応しているメモリのモジュール規格のうち，もっとも上位のものだけが書かれていたためである．記載されているもの以外の対応しているモジュール規格の情報は，別のパーツ小売店の Web サイトやマザーボードのメーカーで発見することができた．

セマンティック Web 技術は様々な情報を合わせて扱うことができるため，1 つの Web サイトからの情報だけでなく，複数の情報源にまたがった推論ができると，より有効な PC パーツの検索になると考えられる．

7. おわりに

本稿ではマザーボードと CPU，マザーボードとメインメモリの適切な組み合わせを OWL の推論を用いて検索する方法とその結果について述べた．しかし，PC のパーツには電源や外部記憶装置など，扱わなかったパーツが数多くある．また，規格としては正しく動作するはずでも，実際に組み立ててみるとうまく動かないという相性問題も，PC を組み立てているときに起こりうる．これらのことにつて，OWL で表現し，より推論の有用性を示していかなければならない．

参考文献

1) OWL Web Ontology Language Overview, http://www.w3.org/TR/2004/REC-owl-features-20040210/
2) Food Ontology , http://www.w3.org/TR/2004/REC-owl-guide-20040210/food.rdf
3) Wine Ontology , http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine.rdf
4) 小町祐史, 大野邦夫, 須栗裕樹, 山田篤 ：オントロジ技術入門―ウェブオントロジと OWL, 東京電機大学出版局, 2005
5) Resource Description Framework, http://www.w3.org/RDF/
6) RDF/XML Syntax Specification, http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/
7) Notation 3, http://www.w3.org/DesignIssues/Notation3
8) RDF Schema, http://www.w3.org/TR/2004/REC-rdf-schema-20040210/
9) パソコン工房, http://www.pc-koubou.jp/
10) Pellet: OWL 2 Reasoner for Java, http://clarkparsia.com/pellet/
11) SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query/